Distributed Agile Development: Using Scrum in a Large Project

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

While seemingly incompatible, combining large-scale global software development and agile practices is a challenge undertaken by many companies. Case study reports on the successful use of agile practices in small distributed projects already exist. How these practices could be applied to larger projects, however, remains unstudied. This paper reports a case study on agile practices in a 40-person development organization distributed between Norway and Malaysia. Based on seven interviews in the development organization, we describe how Scrum practices were successfully applied, e.g., using teleconference and web cameras for daily scrum meetings, synchronized 4-week sprints and weekly scrum-of-scrums. Additional agility supporting practices for distributed projects were identified, e.g., frequent visits, unofficial distributed meetings and annual gatherings are described.

1. Introduction

Many software projects are faced with global software development (GSD), including outsourcing, subcontracting and partnerships. A lot of challenges related to GSD have been identified [12,19] and solutions to these have been proposed, e.g., dividing work into separate modules and making them independent in order to minimize communication between sites [10]. However, for software projects developing genuinely novel products, making a clear modular structure and minimizing communication might be difficult or even impossible. Projects with uncertain requirements and implementation technologies cannot provide clear requirement specifications to all parties up front [23].

In collocated software development this kind of project scenario led to the introduction of agile methods. At the bottom of these methods lies the assumption that software development is an empirical rather than a defined process and needs a different way of working than described in the waterfall model [27,29].

Cockburn proposes that agile methods are particularly suitable for development projects facing high uncertainty [5]. Nevertheless, agile methods cannot be simply taken into use in GSD as they rely heavily on face-to-face communication [6], which is very seldom possible in distributed projects. To gain benefits from agile development the practices need to be modified in order to apply to distributed settings.

Some case studies [8,9,21,31] have shown that agile methods such as Scrum [29] and XP [1] can be successfully customized to distributed projects. Even distributed versions of agile methods like distributed Scrum [32] and DXP [16] have been developed.

Nevertheless, advice 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., [33,34]. However, it seems that quite many companies are interested in taking DAD into use, or have already started to use it.

In particular, advice for large projects is missing in the literature. Existing publications are mostly limited to the use of XP in small and middle-sized GSD projects [3,11,20]. Although there are some experience reports discussing the aspects of Scrum in GSD [e.g., 2,14], no case studies have been performed on a distributed Scrum project with the aim of thoroughly describing the applied practices and their benefits and related challenges. The aim of this paper is to report on such a case study.

This paper presents a single-case study of a large distributed software development project that started to apply agile practices from Scrum. First, we present experiences reported in the literature on using agile practices in distributed projects. Then, we describe our case study method and our findings. The results include experiences on how Scrum practices were applied in our case project, what kind of challenges were faced and what kind of other support practices were taken into use. Finally, we discuss our findings and suggest future research topics.

978-0-7695-3280-6/08 $25.00 © 2008 IEEE
DOI 10.1109/ICGSE.2008.38

2. Related Work

Distributed software projects with volatile requirements and uncertain implementation technologies need effective practices to be organized and managed successfully. We performed a systematic literature review on practices used in distributed agile software development projects. Based on our findings from the literature we divided the practices into two groups: 1) agile practices, and 2) agility supporting distributed practices. Agile practices include practices that come from some agile method and are either used as such or applied to distributed projects. Agility supporting distributed practices include practices that help to tackle some of the new challenges arising from DAD. They can be also known GSD practices that can help the project to be more agile.

2.1 Agile practices

The literature offers some advice especially on how to use Scrum and XP practices in distributed agile projects. The most frequently reported practice is the daily scrum meeting, as it provides at least a partial solution to one of the biggest challenges in GSD [19] and in DAD – communication [25,26,31]. The daily scrum meeting normally takes 15 minutes during which each team member answers the three scrum questions [28]: What did you do since last scrum meeting? Do you have any obstacles? What will you do before next meeting? Sutherland et al. recommends answering these three scrum questions before the scrum meeting via email to shorten the time needed for a teleconference [32]. This approach also helps to overcome language barriers. Berczuk reports that Skype with speaker and microphone, provides sufficient voice quality and is easy to set up for the scrum meetings [2]. The use of Web conferencing for these meetings has also been described [7,14]. According to [7], daily scrum meetings both enhance communication and provide a coordination mechanism for everyone in the project. Jensen and Zilmer [14] add that these meetings greatly improve cross-team project management. Burndown charts displayed on the Wiki pages for cross-location viewing [7] provide visibility for the project progress.

Sprint planning and review meetings improve communication, coordination and team cohesion [12]. Usually the whole team participates in a planning meeting [1,28]. Due to time-zone differences it can be difficult to find a sufficient block of time that suits all parties [2]. Thus, Layman et al. [18] suggest that only lead-developers take part in the planning meetings. Simons [31] on the other hand emphasizes the importance of the participation of the whole team to get the viewpoints of all team members. He suggests that pre-work should be done before the actual planning meeting: customers and developers should clarify as many issues as possible before the meetings to keep the actual planning meeting short since these kind of remote meetings held via web conference [18] or phone [31] are arduous. Berczuk [2] reports that sprint review meetings can be centered on the local team and the product owner to minimize the amount of remote meetings.

Demonstrations of working functionalities can be arranged for customers at the end of every sprint [7,9]. They might be held using videoconferencing with desktop sharing [23]. Jain [13] reports that a customer could try out a new feature using VNC to connect to the developer's computer.

It might be difficult for an offshore team to have a business customer working physically on their development site [31]. Instead, a proxy or a remote customer may help transfer business domain knowledge from the customer to the developers, and thus enhance communication [21]. The remote customer communicates with the development team either through videoconferencing or email [16]. A further possibility is to send to the customer prototypes that he can test and comment [18]. A proxy customer is helpful when the real customer is unavailable to answer questions. The proxy customer can make decisions on the behalf of the real customer [18] and interface with the real customer when questions arise that he cannot answer [31].

2.2 Agility supporting distributed 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. Also known GSD practices can help a team to be more agile. Practices that have been created to solve problems caused by the distributedness of team members and that also help projects in their attempt to develop in an agile way are here referred to as "agility supporting distributed practices".

The challenges caused by the distribution include, for example, communication problems [2,16,25,26,31,32], lack of close physical proximity [12,22,34], lack of team cohesion [26], lack of shared context and knowledge [31,34], and unavailability of team members [34].

The supporting distributed practices strive to provide solutions to some of the challenges. Frequent visits are used to build and maintain collaboration relationships between distributed team members [9]. A good collaboration relationship needs to be created for the remote communication to work effectively. There are two kinds of visits: seeding visits and maintaining visits. Seeding visits occur early in the project. Their aim is to build a relationship [9,17] and they should be intense during the early development cycles [26]. Maintaining visits are shorter and they aim to maintain the collaboration relationship [9]. Ramesh et al. [26] recommend that people from both onshore and offshore sites should travel. Customers and product managers should visit the development team. Developer representatives should travel to the customer site. According to Braithwaite and Joyce [4] and Danait [7], team members should conti-

nually rotate between the sites and at least one team member at any time should be away from his or her home location. This helps distant teams get along better and develop peer-to-peer relationships [7]. It is easier to maintain trust with remote team members if you have worked collocated with them in the past [4].

A practice that particularly enhances communication between distributed teams is multiple communication modes [4]. Team members should be provided with several different kinds of communication media that can also be applied in parallel, i.e., individual and conference telephone, teleconference, videoconference, email, instant messaging, Wiki and desktop sharing. A major part of the missing face-to-face communication is substituted by these tools to alleviate distance [4].

Mirroring [11] and balanced sites [4] can be used to reduce the dependency on the other site. These practices mean that each role in a team at one site has a counterpart on the other site. These persons may work closely together with their counterparts. For example, special knowledge on architecture is good to have on both sites.

For successful collaboration, team members need to gain a common understanding on how the teams at different sites work, think, communicate, and in general deal with the various issues and problems that arise from the development effort. Therefore, it is useful to send an experienced engineer for a longer period of time to the other site to facilitate cultural exchange [25].

The ambassador [11,21] or rotating guru [34] are similar practices that aim at more than just cultural exchange. The ambassadors report lessons learned and set future directions for the project. They participate in daily meetings and retrospectives of the visited team [11]. Business-oriented ambassadors provide business context information to the offshore team [4]. Rotating gurus, who usually are senior team members, provide initial training and mentoring to the other site. As pointed out by Nisar and Hameed [21], this is workable only for a bit larger budget projects.

The synchronization of working hours that is a widespread practice in GSD is important also for DAD. Constant communication is possible only through the maximization of overlapping work hours. For example, early morning shifts for onshore site and late evening shifts for the offshore site let the teams "to share the pain" of synchronizing the work schedule [26].

3. Research method

The research presented in this paper is a single-case study [35] of a large distributed product program using an agile process. We used purposeful sampling [24] when looking for a suitable case. Our aim was to find a globally distributed project that already for some time had used an agile method or at least a collection of agile practices. The large globally distributed IT company that we contacted searched for suitable projects for our study, this chosen case being one of them.

The product program that we will from now on call EnergySoftware, develops a software product for oil and energy companies. The product is already in use in several customer companies and the current development aims for new versions. Every new version is a new "project" and the development of the whole product is called here as program. The development work is distributed between two countries, Norway and Malaysia, where our case company has its own offices. We will describe the case program in more detail in Section 4.1.

We collected data using semi-structured, open-ended interviews that were recorded. We asked our interviewees to tell in their own words about the project and the practices they use. Altogether we performed seven interviews each lasting 1-2 hours. From Norway we interviewed 4 persons, each in a face-to-face interview with one researcher asking questions and the other one taking notes. From Malaysia we were able to interview only one person face-to-face while this person was visiting Norway. In this interview we had one researcher interviewing, but no note taking. Since we could not visit Malaysia for cost reasons, we interviewed two additional persons from Malaysia over the phone. We used SkypeOut calls that were recorded. From Norway we interviewed a product development director, a scrum master and two developers. From Malaysia we interviewed three developers. One of them had left the project recently.

We sent all recordings to an outside professional transcription company. One researcher that had participated in the interviews checked each transcription by listening to the tape at the same time correcting possible transcription mistakes.

The qualitative data analysis was done by one researcher, who coded all the interviews, developing separate codes for all the identified practices and problems.

4. Results

4.1 Case description

Our case study is a product development program from an IT company that has a globally distributed organization. The company's main activities are in Europe, but it also has offices in other parts of the world, e.g., in Asia. The company is quite experienced in carrying out distributed projects, but using agile methods in distributed projects is a new experience.

The case chosen for our study is a large product development program that has been going on for ten years. Many large customers all over the world are using the product. New product versions are released approximately twice a year and between the main releases, service packs are

shipped. Currently, six old versions are maintained, since not all the customers upgrade with every new release.

The product organization has grown over the years from 18 persons to a current size of 190 employees. The service organization all over the world sets up new releases for customers, e.g. by configurating the system and in some cases they make also modifications. The product development organization has approximately 20 persons in one location in Norway and another 20 persons in one location in Malaysia. The Malaysian organization is slowly growing. In this study we focus only on the product development organization.

Our case oraganization has used Scrum for 1,5 years. Earlier, their development process was a combination of waterfall and iterative way of working, containing lots of up-front planning activities and a massive amount of documentation. The old process was considered very rigid and bureaucratic. The development was done in increments. The quality was not considered good enough after the first increments, and only after the last increment everything worked. There was also a perceived customer need for more agility. The organization had had an unchanged way of doing things already for many years. This was another driver for change. The project manager in Norway proposed Scrum, and discussed it with key development personnel. Subsequently they decided to try it; to do something completely new. They got an approval from a higher-level manager to test Scrum for three months. They have been using Scrum ever since.

When starting to use Scrum they gave a brief description of the process, and basically explained: "The old processes are not mandatory anymore. You can use whatever process you want". However, there were some minimal requirements for which practices to apply and some rules: Specifications should be written for everything that is implemented, a detailed design document should be produced, and there has to be some evidence for all testing that is done. The most important goal for the teams was to produce release quality in every sprint.

The overall product consists of five modules or "products" as the teams call them. A customer can buy one or more "products". Since the program is constantly developing new versions, each release project is called "a project". The product development organization is divided into seven teams. Five of them are built around the different modules or "products", and each of them has a separate product owner. The product owners are not seen as team members, maybe because they travel a lot collecting requirements. The two remaining teams include a framework team and a maintenance team. The maintenance team is seen as the most important team, where the most experienced persons are required, since they need to know the whole product and their work has direct and constant impact on customer satisfaction.

The number of persons in each team varies between release projects and also between iterations, mainly because different releases emphasize different modules. In each team, there may be from two up to nine persons. Some teams are distributed between Norway and Malaysia, but not all. In some teams all the developers are in Malaysia and only their product owner is in Norway. Actually, all the product owners are located in Norway. There are also two scrum masters and one back-up scrum master in Norway and one scrum master in Malaysia. This means that one scrum master may have several teams to work with. The development organization is depicted in Figure 1.

The time difference between the two sites is seven hours during the winter and six hours during the summer. However, the daily working time is a bit longer in Malaysia and they also have longer breaks. Thus, the common working time for the sites is normally a couple of hours during winter and one hour more during summer.

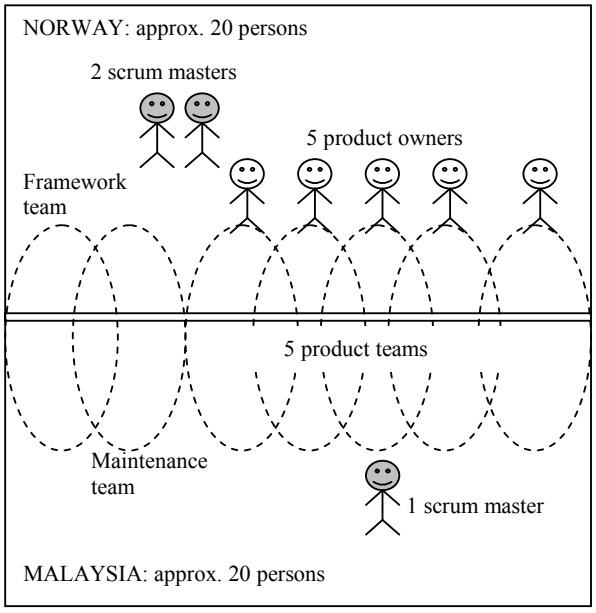

Figure 1. Product development organization.

4.2 Agile practices

In this section we describe how agile practices were applied in this distributed product program.

Daily scrum meetings. Daily scrum meetings between sites are arranged using telephone conferencing and web-cameras. Application sharing is also possible, but is not normally needed. The meetings take place during the two hours that the teams have common working time. The meetings for different teams are consecutive, and take place in the same meeting room. The first meeting starts at quar-

ter to nine Norwegian time. Then the next one starts at nine and so on. The consecutive order makes it possible for the same persons to participate in several meetings, if needed. This is the case, for example, for the scrum masters. In addition to the team members, also the product owner normally joins the meeting. Even if the team has members only from Malaysia, the meeting is still distributed, since the product owner is in Norway. After the three scrum questions there are normally some discussions or questions.

In the beginning, it was difficult to encourage everybody to talk and tell enough about their tasks and impediments. This was the case especially for persons coming from an Asian culture. Thus, in the beginning, the meetings lasted only a few minutes. Now everybody is used to the meetings, and communication is better. The meetings take 15 minutes each. If more discussion or clarifications are needed, the team schedules additional meetings after the scrum meeting.

Weekly scrum-of-scrums. A scrum-of-scrums meeting is arranged once a week. One team member from each team participates in this meeting. The team decides who participates; the participant is not always the same person but can vary. In addition to these team members, all scrum masters typically also participate in this meeting. The meeting takes half an hour. During the meeting the three scrum questions are answered. Now the questions do not address single persons, but a whole team. Thus, each team representative tells what his or her team has been doing since the last meeting, what it plans to do before the next meeting and what kind of impediments they have. 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 is to ensure successful integration.

Synchronized 4-week sprints. The five module teams and the framework team have synchronized 4-week sprints. This means that all the sprints start and end at the same time. They have the same code trees, and at the end of a sprint they build an environment for the sprint demonstrations. Even though the teams during one release project stay quite stable, it is possible to change the emphasis between different modules between the sprints and move persons from one team to another.

2-week maintenance sprints. The maintenance team is the only exception to the 4-week sprint cycle, as their sprint cycle is two weeks. The reason is that hot fixes are released every two weeks. The maintenance team has a sprint planning session, during which they choose a set of issues from the backlog for the sprint. Since customers sometimes need fixes right away, they have a fast track routine for handling such requests. To accommodate for this, the team leaves a buffer of 20% of the capacity for handling fast track issues. The needed buffer is possible to forecast at least to some level, since it depends on the situation in the customer installation projects at that moment.

Distributed sprint planning meetings. The sprint planning meetings are divided into three phases: distributed meeting, local meeting in Norway and local meeting in Malaysia.

The first part is time-boxed for three hours. Malaysian and Norwegian team members participate in this virtual meeting through teleconferencing. For application sharing, Microsoft NetMeeting is used. In these meetings, webcameras are not used. In the meetings, the Scrum framework, a document that describes how Scrum is applied, is typically reviewed first. In particular, any adjustments done since the last sprint planning are highlighted. An adjustment could be, for instance, a new concept for doing peer reviews. Thus, sprint planning sessions can have the additional role of a training session.

After that the product owner starts to go through items in the backlog, and the team asks questions. Before the meeting, the product owner has prioritized the backlog and made preliminary estimates for the backlog items. This phase of the meeting lasts until lunch hour in Norway. Then it is time for the Malaysian team to go home.

The Norwegian team typically continues the meeting for the rest of the day. The team divides the backlog items into more detailed tasks and adjusts the estimates made by the product owner. Moreover, the team makes at least the initial assignment of the tasks to different team members.

The Malaysian team continues the work the following morning. They meet locally and discuss and comment on the draft plan they have received from Norway. They might do some adjustments and also add their names to some tasks. If needed, the issues raised are discussed together in the next daily meeting.

Sprint demos. Demos are arranged using the same technology as in the sprint planning meetings: teleconference and application sharing. In addition to the team, their product owner and scrum master participate. The team has prepared an agenda for the demonstration. Following the agenda, each issue is gone through. If the quality seems to be good, everybody applauds.

Retrospective meetings. Retrospective meetings take place directly after demos. The team, their product owner, and scrum master participate in the retrospective, which is arranged as a distributed meeting. It is time-boxed for a maximum of one hour, during which 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?".

Nightly builds and automated testing. The teams check in their code at least daily to CVS, a centralized version control system in Norway. It can be accessed by all team members. During every night the whole product is built. If the build is not successful, the team that broke it will fix errors and rebuild after the errors have been fixed.

Separate backlogs for each team. The project uses a tool called Jira for managing backlogs. All team members

have access to Jira. Each team has its own backlog in Jira. It is updated by the respective product owners. The maintenance team has an own backlog, to which all product owners can add new issues. The process of adding issues to the maintenance backlog is a bit different from what the other teams have: All customers have access to Jira, where they report found bugs. The service organization checks each issue first, and if they find a product related bug they move that task over to product development and assign it to the product owner of that specific area. The assigned issues are prioritized using categories describing their criticality. The product owner who received the bug analyses it and when he marks it "verified", it automatically moves to the maintenance backlog. The product owner decides, as well, to which maintained versions a fix is going to be done. All teams have been very satisfied with this tool.

Team rooms. The basic principle is that in Malaysia and in Norway there is one room for each team, so that at least for collocated team members it is easy to discuss. When a person switches his team, he also relocates into the new team's room. The project has gradually moved to this.

Transition backlog. The transition backlog is not yet in use, but in the planning state. The idea is to put in good practices that some team has tried and found useful into the backlog. In this backlog they could keep track of the improvements as well as prioritize them.

4.3 Agility supporting distributed practices

Unofficial distributed meetings. After starting to use daily scrums, the teams feel that there is more one-to-one communication between the sites. This kind of communication often takes place after official scrum meetings. For example, if one or two persons from Malaysia are starting a bigger task they may decide to have a start-up meeting with the product owner to discuss details. Unofficial communication may take place through teleconferences, chat, email, or over the internet using headset. Some team members also use web cameras.

Centralized version control. The project has a centralized version control system, a CVS server, in Norway. It can be accessed by all team members through VPN. The teams check in their code at least daily.

Visiting engineer during the first iteration. When the daily scrum was taken into use in the Malaysian office, one engineer, a scrum master from the Norwegian office, traveled to Malaysia and stayed there during the first sprint. He had studied Scrum, but it was still quite new to him. Thus, he worked together with the teams and they jointly found solutions for how to apply Scrum.

Onsite system expert. When the Malaysian office was established four years ago, two persons from Norway moved there. One of them runs the office, and the other one is an expert who knows the customers' business and how different components of EnergySoftware work.

Frequent visits. During the first half a year after starting to apply Scrum, the project had several persons from Norway working full-time in Malaysia. Now, the team members travel frequently between the sites. These visits normally last between two to four weeks, which makes it possible for a team to really work together. Especially during the critical phases, it is 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.

At the moment, people travel when needed. In the beginning, they had a traveling plan for the Norwegians. Now they have noticed that the Malaysian team members travel more often. Thus, they are thinking of taking into use a traveling plan for the Norwegians again to make them travel. Most Norwegians are subject area experts, whose knowledge it would be beneficial to share in Malaysia. At the moment, approximately half of the project team has visited the other site and all have met face-to-face in the annual gatherings.

Annual gathering. The project has arranged social gatherings for the whole project, including also the service organization. The last gathering was in Rome and all 190 persons traveled there for a long weekend. The product development organization held a half-day session together during which they heard and discussed future actions and had a team building exercise. The rest of the time was used for common social events. Everybody we interviewed saw this as a successful event and managers thought that it was an investment for the future. It was an event that people were still talking about.

4.4 Challenges faced

No possibility for videoconference. One challenge at the moment is that the network connections between the offices are not fast enough for videoconferencing. However, they hope this to be solved soon, in order to make it possible to see people and their reactions in meetings. The web camera solution, which is in use at the moment, is not good enough: the resolution is poor, and there are transmission delays.

Misunderstanding requirements. Misunderstandings are common. A developer might have a totally different picture in his head about some functionality than a product owner. This is especially the case when developers are sitting far away from the product owner, but it has also happened in Norway. In the worst case, the misunderstanding might not be noticed until the iteration demo. To verify the correct understanding, the product owners have learned to ask follow-up questions. By asking such questions, they make a developer explain in his own words what he is planning to do or what the functionality should do. The follow-up question can be for example: "Do you have an idea how you are going to code this?" These kinds of ques-

tions can be asked at the end of daily scrum meetings to begin a discussion.

Silence caused by distance. In the beginning, the daily scrum meetings were very short when team members did not know how to communicate and how much to tell. Now, after a period of daily practice, this problem is over. However, the same problem is still present in some sprint planning sessions, where discussion is less guided. Especially the Norwegians think that the Malaysian team members are very silent. We got comments like this: "It can go maybe 20 minutes before we hear anything or we have to ask: Are you still there?" The Norwegians felt that they themselves are very eager to ask questions from the product owner in these meetings, almost like attacking him with questions, whereas the Malaysians seem to prefer just listening. Thus, the Norwegians felt that they cannot be sure whether the off-site team has really understood everything. They were hoping to find a better solution for this.

4.5 Positive experiences

Even though facing challenges, the overall experience on starting to use agile practices in this large distributed project has been very positive. Based on our interviews, it seemed that the whole organization was satisfied with the decision to start to use an agile method. They saw many things that had helped them make this change successful: their organization was already very experienced, they had competent people, they got commitment from the management and they had a version control system and regular builds already in place.

The agile practices were considered very suitable for distributed projects, as one of our interviewees stated:"I think the strength of working in an agile way when we are different, and in different locations is that you actually have more frequent communication. And also focus on issues, impediments, and that you solve them on the way and do not wait until they make a big risk for the project."

Taking agile practices into use had many benefits according to our interviewees. The most important benefits mentioned are discussed next.

Better quality. One of the biggest differences to the earlier way of working was an improvement in overall quality. Although increments had been used before, the quality of the product was not good after the first increments: only at the end of the last increment everything worked. Now, at the end of each iteration, a working demo is built. Thus, the quality at the end of an iteration is quite close to release quality.

Better and more frequent communication. Our interviewees thought that communication quality is better than before and communication is much more frequent, especially between the sites. There is also more one-to-one communication between the sites, e.g., after the daily scrum meetings.

A big difference between the current situation and the situation one year ago was perceived. Now everybody has to talk, and is trained to talk in the scrum meetings. There is a feeling that the Scrum setting has forced increased communication, which is considered to be very beneficial. Surveys to all team members after each release project have been conducted already before the agile time. In these surveys they were asked, e.g., about processes used, testing concepts, code reviews, documentation, product owners, and cooperation and communication. The scores for communication and collaboration have risen continuously since they started using Scrum. Currently, they score very high. One of the scrum masters commented on this saying: "This gives a good indication that Scrum works."

Frequent communication also makes it easier to identify misunderstandings, e.g. regarding the understanding of the requirements, making it possible to take corrective action earlier.

Improved motivation. The team members are now more motivated than before. A scrum master explained: "They motivate themselves (…) They get clear and quick clarification so they don't have to wait. They get access to the right people at the right time and everyone has the same value in the team (…) That they can have an impact on the work that they and also their team is doing, is something that makes you feel good."

5. Discussion

In this paper we have described how agile practices based on Scrum have been applied to a large distributed software development program. The overall experiences in using Scrum in a distributed setting have been very positive. All our interviewees stated that according to their experience an agile method, like Scrum, suits well to distributed settings and actually helps in solving the biggest problem of GSD projects, namely communication, by almost forcing distributed team members to communicate frequently and really learn to communicate.

The contribution of this paper is a detailed description on how agile practices have been successfully applied to a distributed software development program. We believe that these experiences are useful for other companies planning to apply agile practices to similar settings.

5.1 Limitations

In this study we were able to interview only seven persons. By being able to interview a few more persons from different teams and from different roles, we might have received a bit more complete picture. Moreover, we were able to interview face-to-face only one Malaysian team member. Two other interviews were telephone interviews, which is always more difficult. It is especially more diffi-

cult to build a confidential atmosphere while not seeing the other person, which may cause that not everything is said.

5.2 Future research

In the future we plan to study a few more distributed projects using agile practices and compare the way they have applied agile practices to distributed settings, the challenges faced and positive experiences gained. Actually, at the moment we have already interviewed persons from two other distributed projects, which have taken agile practices from Scrum into use.

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 would also like to thank our case company and all the interviewees.

References

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

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

[3] Boland, D. and Fitzgerald, B. Transitioning from a Co-Located to a Globally-Distributed Software Development Team: A Case Study at Analog Devices Inc. *Proceedings of the ICSE Workshop on Global Software Development*, Edinburgh, Scotland, 2004.

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

[5] Cockburn, A. *Agile software development.* Boston, MA, USA: Addison-Wesley, Longman Publishing Co., Inc., 2002.

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

[7] Danait, A. Agile offshore techniques - a case study. *Proceedings of the Agile Conference*, July 24-29, 2005, pp. 214-217.

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

[9] Fowler, M. Using an agile software process with offshore development. 2006 (Available: http://martinfowler.com/articles/agileOffshore.html) Referenced: 19.12.2007.

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

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

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

[13] Jain, N. Offshore agile maintenance. *Proceedings of the Agile Conference*, 2006.

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

[15] Karlsson, E., Andersson, L. and Leion, P. Daily Build and Feature Development in Large Distributed Projects. *Proceedings of the International Conference on Software Engineering,* Limeric, Ireland, 2000, pp. 649-658.

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

[17] Kussmaul, C., Jack, R., and Sponsler, B. Outsourcing and offshoring with agility: A case study. *Proceedings of the 4th Conference on Extreme Programming and Agile Methods*, Calgary, Canada, August 15-18, 2004. Springer Berlin/Heidelberg, pp, 147-154.

[18] 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, 2006, pp. 781-794.

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

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

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

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

[23] 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.

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

[25] Poole, C. J. Distributed product development using extreme programming. *Proceedings of the 5th International Conference, XP 2004,* Garmisch-Partenkirchen, Germany, June 6-10, 2004. Springer Berlin/Heidelberg, pp. 60-67.

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

[27] Royce, W. Managing the Development of Large Software Systems. Reprinted from *Proceedings, IEEE WESCON*, August 1970, pp 1-9.

[28] Schwaber, K. and Beedle, M. *Agile Software Development with Scrum*. New York, United States: Prentice Hall, 2001.

[29] Schwaber, K. and Beedle, M. *Agile Software Development with Scrum*. Prentice-Hall, Upper Saddle River, N.J., 2002.

[30] Schwaber, K. *Agile Project Management with Scrum*. Microsoft Press, Redmond, Washington, 2004.

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

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

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

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

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