Identifying Critical Areas for Improvement in Agile Multi-site Co-development

Raoul Vallon, Klaus Bayrhammer, Stefan Strobl, Mario Bernhart and Thomas Grechenig
Research Group for Industrial Software, Vienna University of Technology, Vienna, Austria

Keywords: Distributed Software Development, Scrum, Agile Software Development, Software Development Process, Behavior Driven Development.

Abstract: Agile processes potentially ease distributed software development by demanding regular communication and self-management of virtual team members. However, being designed for collocated teams, extensions to the regular process need to be made. We investigate critical areas of improvement based on a case of distributed Scrum involving two unaffiliated Austrian IT organizations that collaborate to build software. We identified eight critical areas for improvement originating from interviews, retrospective meetings and an in-depth case analysis. Key suggestions for practice include the establishment of long-lived single-site Scrum teams and the application of Behavior Driven Development (BDD) to make implicit requirement knowledge explicit and transparent to all of the distributed parties.

1 INTRODUCTION

Distributed development challenges one of the core strengths of Scrum: team members need to interact and communicate on a daily basis to form self-organizing teams and meet sprint goals. However, distributed environments complicate communication and coordination (Korkala and Abrahamsson, 2007). Technical tool support plays a bigger role in the process (Dullemond et al., 2009); (Niinimäki, 2011) as well as knowledge management and transfer (Dorairaj et al., 2012). Consequently team members need to work harder to synchronize and meet sprint goals.

Although originally designed for collocated teams, related agile studies have reported the adaption of agile principles to e.g. a distributed Scrum (Paasivaara et al., 2009); (Bannerman et al., 2012) or Extreme Programming (XP) (Hildenbrand et al., 2008) implementation in recent years.

In this paper we identify critical areas for improvement based on our exploratory case study (Vallon et al., 2013), discuss solutions and examine our findings with regard to both traditional and agile related studies. The distributed Scrum implementation involves two unaffiliated Austrian IT organizations, which are separated by about 300 kilometers. We will investigate challenges and areas for improvement, when two organizations with different corporate cultures join forces to develop a software product.

We defined the following research question:

"What are critical areas for improvement in agile multi-site distributed software development?"

2 RESEARCH DESIGN

2.1 Research Setting

Two unaffiliated organizations, the main supplier (MS) and the additional supplier (AS), collaborate to develop three software products that share a common codebase. Both suppliers have successfully applied regular Scrum before and chose to implement an adapted version of Scrum to better suit the needs of a distributed development environment. The two organizations develop at their own sites, separated by about 300 kilometers.

The MS is a large company whose IT department is involved in the development of the three software products. It is responsible for requirements engineering with all three customers and provides the bigger part of the development staff.

The AS is a medium-sized core software development company. It complements the MS's

Vallon R., Bayrhammer K., Strobl S., Bernhart M. and Grechenig T..
Identifying Critical Areas for Improvement in Agile Multi-site Co-development.
DOI: 10.5220/0004446501650172
In *Proceedings of the 8th International Conference on Evaluation of Novel Approaches to Software Engineering* (ENASE-2013), pages 165-172
ISBN: 978-989-8565-62-4

development staff and has no contact with customers.

Table 1 shows the distribution of team members over the two suppliers. The MS has one product owner (PO) for each software product and three Scrum masters (SM) serving three teams. The AS does neither have a PO nor a SM on site.

Table 1: Distribution of team members over the two suppliers.

Parties	Dev	Test	SM	PO	Sum
Main Supplier (MS)	11	3	3	3	20
Additional Supplie (AS)	8	2	0	0	10
Overall	19	5	3	3	30

2.2 Research Method

The research is divided into three phases: observations, case analysis and discussion as illustrated in Figure 1.

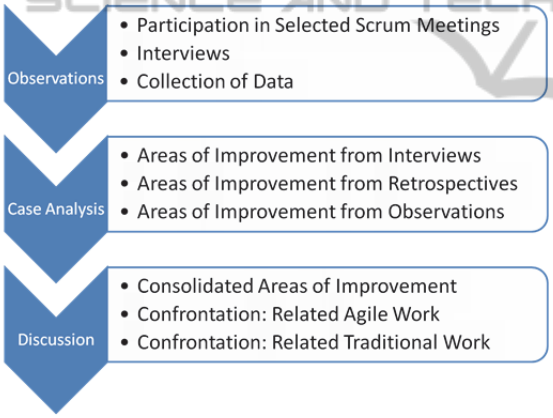

Figure 1: Research phases and corresponding tasks.

Observations. One of the authors examined the Scrum implementation in use as an external observer. As such, he took part in various meetings and conducted interviews with members of all roles (product owner, Scrum master, developer, tester). The interviews lasted from 20 to 45 minutes and have been audio-recorded. He took field notes, pictures and collected planning sheets and meeting minutes. He has been granted read-only access to several electronic tools involved such as the issue tracking system. This phase lasted for three months.

Case Analysis. After the observation phase the collected data was analyzed by the authors. A problem root cause analysis showed problem clusters and identified relevant root causes in a top-down fashion, i.e. most prominent problems first. This phase lasted for two months.

Discussion. The last phase involved a presentation and discussion of results with team members including critical areas of improvement. This was the concluding step in the last month. The goal of this phase was to confront the findings from this case study with related work.

3 OBSERVATION PHASE

The following observations summarize the different aspects of the Scrum implementation applied in the case study including strengths and weaknesses.

Formation of Scrum Teams. Three Scrum teams have been formed across all products and based on logical requirement areas. The product owner and the Scrum master roles are both on the MS's site. The AS complements the MS with additional developers and testers but has no official Scrum roles. However, two developers have emerged as unofficial Scrum masters for the AS. They care most for the process implementation and discuss impediments with the MS. One of these unofficial Scrum masters travels to the MS's site once a week for face to face updates and discussions.

Each Scrum team holds a daily video conference meeting, where respective team members of the MS and AS participate. Additionally a Scrum of Scrums (SoS) meeting is established for inter-team communication.

It is held daily at the MS's site. Since the AS does not have official Scrum roles, the MS handles all inter-team coordination. Testers are assigned to the three Scrum teams, but hold an additional daily meeting to stay synchronized and also send a representative to the SoS. Product owners also participate to evaluate the progress.

Two-tiered Planning Process. Planning covers one month, i.e. two sprints. It is a two-tiered process: at first, planning is done at the MS's site with one of the two unofficial Scrum masters of the AS present. The Scrum teams decide, which of the prioritized user stories in the product backlogs they want to implement in the next two sprints.

The second level planning continues at the AS's site: The unofficial Scrum master returns from the MS with pre-estimated (via planning poker: Grenning, 2002) user stories for the AS. Team members volunteer for certain tasks until all tasks are assigned. When a team member accepts a task, it adjusts the original estimation of the MS to its own. One of the unofficial Scrum masters updates a

planning spreadsheet during the meeting and shares it with the MS afterwards.

Joint Sprint Review. The sprint review is held after each sprint. It is primarily held at the MS, but the AS joins via video conference. Additionally, one of the AS's Scrum masters is present at the MS's site to represent the AS in person as well. The rest of the AS's team is mainly observing the review, but can raise questions or concerns when necessary. The review consists of feature demonstrations and discussions about different areas of the current product increments and takes about two hours.

Joint Retrospective. The sprint retrospective is held monthly after two sprints with the same setup as the sprint review. It includes an individual evaluation of the last month and the impact of measures taken. Then new remarks are presented, clustered to topics and weighed by team members (by assigning points). Top three issues will get picked up and measures discussed.

Product and Sprint Backlog. Each product owner maintains a product backlog on the MS' site for his product. At the time of the observation phase the AS did not have access to the product backlog, but worked with the sprint backlog only (planning spreadsheet from the two-tiered planning process).

Scrum Board. Both the MS and the AS are using paper Scrum boards. Each Scrum team operates one board. Since the two suppliers are based at different locations, six boards would be needed, but the AS currently only uses one general board on its site covering all three teams. The workflow on the board is defined as: *User Stories*, *TO DOs*, *In Progress*, *Review* (by the other supplier) and *Done*.

Burndown Chart. The burndown charts are drawn and updated on paper at the MS's site only (one per team). The AS does not operate one on its own, but the MS includes the AS's tasks in its chart.

Behavior Driven Development. The two suppliers develop software using behavior driven development (BDD: North, 2006), which is an extension to test driven development (TDD: Beck, 2003). The goal is to have executable yet human-readable specification in terms of different scenarios for each story.

Means of Communication. The main means of communication between the two suppliers are joint meetings via video conference and telephone calls. Individual concerns are discussed in emails, instant messaging and screen sharing sessions.

3.1 Retrospective

In the three retrospective meetings during the observation phase, issues overweighed strengths by far due to the complex development environment. Named strengths were *improved communication and collaboration* in general and *continuous improvement*. Team members identified the following drivers for improvement:

- Willingness and commitment to change and improve
- Good working atmosphere and employee attitude
- Highly motivated people
- Team work

The list of problems taken from retrospective meetings is notably longer. Both team members of MS and AS reported to suffer from **constant stress in the two-week sprint** due to the following reasons:

- Workload too high in relation to available staff
- Planning delay in general and also between the two suppliers
- Too little time to follow BDD workflow in a two-week sprint

Late planning was reported since inter-organizational planning was frequently not ready until a few days into the sprint iteration. This made it very hard for team members to reach sprint goals. The **BDD workflow** introduced a lot of overhead, which was underestimated and resulted in broken test cases and thus **bad code quality**. Problems with the speed of **remote access for the AS** arose, which slowed down co-development. Minor issues regarding the **quality of use cases** were also reported.

3.2 Interviews

Three prominent issues have been identified from one on one semi-structured interviews. Issues regarding an **overhead of communication and coordination** addressed the inadequate quality of video conferences, especially with larger groups (joint sprint review/retrospective). The **lack of electronic tool support** has been criticized, especially by the AS. The AS did not have access to the main paper Scrum boards and burndown charts at the MS's site and progress was synchronized mostly during daily Scrum meetings. **Two-tiered sprint planning** put pressure on team members' commitment since planning took too long and was frequently not ready at the beginning of new sprint

iterations. Interviews have only been conducted with the AS, so a bias concern needs to be raised. Table 2 presents selected quotations from interviews concerning the three problem categories.

Table 2: Selected quotations from interviews with the AS.

Issues	Selected Quotations from Interviews
Lack of Tool Support	"There are different systems, one for tickets, one does time management, planning is done in spreadsheets. So you see there are many small systems, but not a single one that can do it all. [...] It does not make sense to use a tool just for ourselves, because then we would have yet another tool and that would [only] increase effort on our side." ('unofficial Scrum Master', AS)
Communication and Coordination Overhead	"We [AS] sent it [BDD Feature File] to them [MS], sometimes too late, because we did not get requirements in time, but then nothing happened for three days. We are running out of time, so we start to develop code. Then we get the reply 'this and that is not okay and needs to be changed' and then we need to start adjusting already developed code." ('unofficial Scrum Master', AS) "They [MS] are not used to work with other suppliers collaboratively and hence naturally the process is focussed on their staff and site. They need to learn that there needs to be a planning that involves both suppliers because we are not within earshot. There is a lot to learn in both directions." ('unofficial Scrum Master', AS)
Two-tiered Sprint Planning	"In one sprint we started 1.5 weeks late, which is a 'great' thing in a 2 weeks sprint. In the end we had 1.5 weeks delay when the team was done, exactly the amount that we started late." ('unofficial Scrum Master', AS) "Once again we do not have enough input for our planning, but still we need to sit together and try to plan to get an idea what everybody is about to do this week." ('unofficial Scrum Master 2', AS) "I would love to break down tasks to a decent level, but if we do not know what should be developed exactly, that is hard to achieve." (Developer, AS)

4 CASE ANALYSIS

After the observation phase, the data collected was analyzed. Problems were identified in a top-down problem root cause analysis and clustered to form areas for improvement. Table 3 shows the identified critical areas of improvement, corresponding problem root causes in the case study and suggested solutions.

Virtual Teams. All three Scrum teams are staffed by members of both suppliers, yet all product owners and Scrum masters are on the MS's site. Nevertheless, two of the AS's team members have emerged that do more coordination work than their colleagues. They care more for the Scrum process than others (Scrum master) and travel to the MS to attend meetings and discuss user stories in person (product owner). The team members on the AS's site are 10 people scattered over three different Scrum teams. It is very hard to remain self-organizing and in compliance with the Scrum process, when contact to the remaining team members is hard to establish and no role is officially assigned to look after the process at the AS's site.

This poses a big problem for the AS, as these two to three team members are separated from the rest of the MS-based team. As a result, the AS has formed a virtual team to manage its own resources with a single paper Scrum board covering all three teams.

The follow-up planning session is also held for the whole virtual AS's team (including members of all three real Scrum teams).

We suggest forming only single-site Scrum teams to disburden intra-team collaboration and organization. Additionally the AS should also have a Scrum master and a product owner. Members of the AS's team should be able to participate in the SoS. Sprint plannings should be held jointly to improve collaboration and knowledge transfer between the two suppliers.

Transparency is a big issue between the two suppliers due to the physical distance of 300 kilometers. The whole process becomes more complex and less transparent. Low quality video conferences and little available documentation further handicap communication and coordination. There is no high level overview of the progress of all three teams available to everyone since Scrum boards and burndown charts are drawn on paper.

We suggest writing meeting protocols for important meetings like sprint planning, review and retrospective. Furthermore equipment needs to be acquired to allow high quality video conferences. The workflow needs to be illustrated truthfully. At the time of the study the BDD steps were not part of the paper boards.

Commitment is hard to achieve with late planning and frequent changes within the sprint iteration. The teams cannot commit to sprint goals when user

Table 3: Areas for improvement, identified problem root causes and suggested solutions.

Area for Improvement	Identified Problem Root Cause	Suggested Solution
A1: Virtual Teams	No Official Scrum Roles at the AS Joint Estimation and Planning Inter-Company Distribution of Members	New Composition of Scrum Teams New Composition of SoS Joint Sprint Planning
A2: Transparency	Suppliers not Collocated Communication Issues Little Documentation No Overview over All Teams	High Quality Video Conferences Meeting Protocols Realistic Illustration of the Workflow
A3: Commitment	Commitment Fails Due to Insufficient Planning Commitment Fails Due to Late Planning Commitment Fails Due to Frequent Changes Little Respect for Iterations	Respect Iterations No Changes Within a Sprint Planning for Commitment Product Backlog Refinement
A4: Planning	Late Actual Beginning of Sprint Little Participation of AS Little Information for AS	Plan In Time Product Owner Team Include AS in Planning
A5: Estimation	User Story Estimation in Hours Pre-estimations by MS	Estimate in Story Points Team Estimates Include Research and Learning in Estimations
A6: Predictability	No Proper Sprint Velocity Further Impediments for Better Predictability	Measure Sprint Velocity with Story Points Discuss Impediments with the Team Long-lived Teams
A7: Self-Organizing Teams	Estimations Based on Individuals Cross-Team Working Agreements	Focus on Team, not Individuals Establish Cross-Team Working Agreements Send Team Members to the SoS Long-lived Teams
A8: Tools	Tools Lack Scrum Compatibility Limited Remote Access for AS Paper Scrum Board and Burndown Chart	Electronic Tool with Scrum Support Introduce Knowledge-Sharing Tool Grant Full Tool Access to AS

stories are not properly and timely specified. As a result, estimations are not reliable.

In order to allow the teams to achieve commitment, the product owners must respect iterations. This includes that no changes to the backlog are allowed within an ongoing sprint. Regular product backlog refinement can be used to update prioritizations and identify dependencies between user stories.

Planning and Estimation. The MS pre-estimates user stories and uses this estimation as a basis for planning. The AS is thus not adequately involved in the planning process apart from updating the estimations of the MS (for its own user stories only). Planning is often not ready until a few days into the sprint, which causes delays for both suppliers. Estimation is done in hours. This does not represent complexity well because different people need different amounts of time to work on a user story.

Planning should be done in time, e.g. at the first day of the new iteration, and with participation of the AS. A product owner team can be formed to coordinate more easily between product owners and have backlogs ready before the beginning of the next sprint. They may run Scrum as well including daily standup meetings.

The team should estimate itself without and pre-estimations (by MS or product owners). It is also important to estimate in story points (Cohn, 2005) instead of hours to represent complexity of stories. Research and learning should also be included in estimations. When a new developer joins the team then learning should be taken into account to get realistic results during estimations.

Predictability. Sprint velocity cannot be properly measured because the MS runs a paper burndown chart that is based solely on tasks (derived from user stories). The only available ratio is tasks per sprint, which does not represent any complexity because it does not take into account hours (or story points). Further impediments to a better predictability are a varying understanding of the BDD workflow among team members and code quality issues.

Sprint velocity can be used to measure empirically how many stories a team will be able to implement. To achieve further predictability impediments must be discussed with the team in daily standups.

Self-organizing Teams. Two developers emerged at the AS's site that do more coordination work and impediment handling than others. The distributed environment complicates coordination between

teams and it is hard for the AS to efficiently complement the MS-based teams.

Moreover, cross-team working agreements regarding the BDD workflow need to be elaborated and agreed upon to reduce interdependency issues.

Scrum directs attention to teams, not individuals. It is very important to respect teams and not to interfere with their self-organization. Teams should be long-lived to increase learning and productivity.

Tools. The electronic tools in use all lack Scrum support, which prevents a proper process implementation. There are currently four paper Scrum boards in use, three at the MS's site for each team and a combined one at the AS's. These are cumbersome to synchronize, which slows down the tracking of other teams' progress. The burndown charts are also drawn on paper and only available to the MS.

In distributed software development paper boards are not a viable option and proper electronic tools need to be installed with full access for both suppliers. Also knowledge-sharing between suppliers and teams for a minimum amount of documentation should be possible, e.g. in a wiki.

5 DISCUSSION

The six-month-long case study involved two suppliers MS and AS from unaffiliated IT organizations, who joined forces to co-develop three software products. The research question was:

"What are critical areas for improvement in agile multi-site distributed software development?"

Based on the case study, we identified eight critical areas for improvement that deserve careful attention when setting up a distributed development environment:

- A1: Virtual Teams
- A2: Transparency
- A3: Commitment
- A4: Planning
- A5: Estimation
- A6: Predictability
- A7: Self-Organizing Teams
- A8: Tools

We will discuss our findings with regard to related agile and traditional work and offer suggestions for practice.

5.1 Related Work

Many studies report on challenges and success stories in both agile and traditional distributed software development.

Hossain, Bannerman and Jeffrey (2011) report that the Scrum model offers improved visibility potential (A2) even in global software development with the remark that collaboration tools (A8) are critical to realize benefits.

Hanssen, Šmite and Moe (2011) argue that agile practices fit well to solve some of distributed development's inherent challenges (A2, A7).

Kamaruddin, Arshad and Mohamed (2012) identified 13 chaos issues in agile global software development including a poor communication infrastructure, low quality and unprepared communication tools (A8). So sources of problems are many and a solid communication infrastructure is one of the most critical factors as has also been shown in our case study.

Deshpande et al. (2010) propose having an onsite coordinator to deal with cultural issues (A1). In our case study we suggest having a Scrum master and product owner on all sites. Although both organizations are Austrian, there is still difference in business culture in every organization that needs coordination and adjustment.

We can see that also traditional distributed development processes suffer from the same key issues (Noll, Beecham and Richardson, 2010). Ester et al. (2012) examined 66 industry projects and concluded that choosing an agile or a structured process does not seem to be crucial decision in globally distributed projects. We believe that a well-adapted Scrum process provides a very suitable toolset to deal with distributed development issues, but that further studies are needed to validate this claim.

5.2 Suggestions for Practice

Based on our research in this case we formulate the following pieces of advice for setting up a distributed development environment.

Allow Team Stability. During the observation phase in this case study teams used to be short-lived and subject to frequent changes. Stable and long-lived teams are very important to support the establishment of self-organizing teams and increase productivity. Stability helps to improve estimations as well as commitments and is crucial for predictable sprint velocities.

Proper Electronic Tool Support. Missing proper electronic tool support makes it impossible to synchronize with distant team members, even in long-lived teams. The emergence of sub-teams (teams within teams) is thus inevitable, which compromises the self-management of teams.

Form Single-site Scrum Teams. Even in a multi-site distributed development environment, we suggest forming single-site Scrum teams only. In the case study an additional Scrum team on the AS's site can be formed including a Scrum master and a product owner instead of having three multi-site teams. Figure 2 illustrates the transformation from three multi-site teams to four single-site teams (three at MS, one at AS). This measure also corresponds to the claim by Larman and Vodde (2009) that feature teams should be collocated.

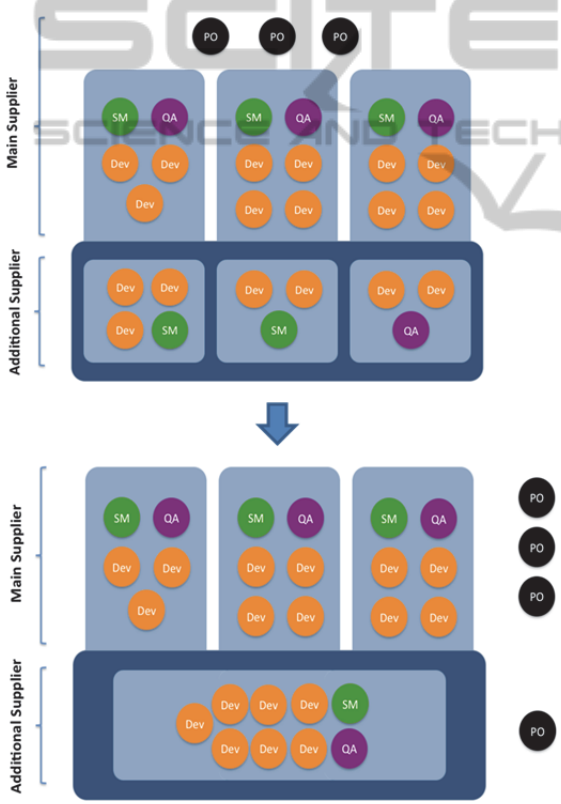

Figure 2: Suggested transformation from three distributed teams to four single-site ones including a Scrum master and product owner at the AS's site.

On-site Scrum Master and Product Owner. We suggest installing at least one Scrum master and product owner on each site. This measure help increase the involvement of all sites in the distributed Scrum process.

Make Implicit Knowledge Explicit with BDD. Most of the user stories have been specified by the MS, which also had more experience with the products beforehand. BDD can be used to make implicit knowledge explicit by specifying executable acceptance criteria available to all parties. In the case study knowledge should be shared about BDD by setting up a workshop. This may also be a good opportunity to meet face to face and have a team event. BDD also helps increase transparency and and the quality of requirements.

6 CONCLUSIONS

When implementing theoretic concepts in practice, compromises will have to be made due to individual practical constraints. Scrum has not been designed for distributed development environments, yet can be successfully adapted when done right. Underlying core agile values need to be respected for improvements to the process.

In this case study we identified eight critical areas for improvement: *Virtual Teams*, *Transparency*, *Commitment*, *Planning*, *Estimation*, *Predictability*, *Self-Organizing Teams* and *Tools*. Suggested measures include establishing long-lived single-site Scrum teams, enabling electronic tool support and using BDD to make implicit requirements knowledge explicit and transparent to all parties involved.

Although related work has reported similar issues, limitations to the study remain because only one case has been examined. Future work should investigate more cases with different distributed development setups (e.g. number of parties involved or physical distance between parties) to see if issues correlate.

REFERENCES

Bannerman, P. L., Hossain, E., Jeffery, R., 2012. Scrum Practice Mitigation of Global Software Development Coordination Challenges: A Distinctive Advantage? In *45th Hawaii International Conference on System Science*, Maui, HI, pp. 5309-5318.

Beck, K., 2003. *Test Driven Development By Example*, Addison-Wesley. Boston, MA, 2nd edition.

Cohn, M., 2005. *Agile Estimating and Planning*, Prentice Hall.

Deshpande, S., Richardson, I., Casey, V., Beecham, S., 2010. Culture in Global Software development - a Weakness or Strength? In *5th IEEE International Conference on Global Software Engineering*,

Princeton, NJ, pp. 67-76.

Dorairaj, S., Noble, J. Malik, P., 2012: Knowledge Management in Distributed Agile Software Development. In *2012 Agile Conference*, Dallas, TX, pp. 63-73.

Dullemond, K., van Gameren, B., van Solingen, R., 2009. How Technological Support Can Enable Advantages of Agile Software Development in a GSE Setting. In *4th IEEE International Conference on Global Software Engineering*, Limerick, Ireland, pp. 143-152.

Estler, H.-C., Nordio, M., Furia, C. A., Meyer, B., Schneider, J., 2012. Agile vs. Structured Distributed Software Development: A Case Study. In *7th IEEE International Conference on Global Software Engineering*, Porto Aleg, Brazil, pp. 11-20.

Grenning, J., 2002. *Planning Poker or How to Avoid Analysis Paralysis While Release Planning*. Available at: http://renaissancesoftware.net/files/articles/PlanningPoker-v1.1.pdf [Accessed 11 April 2013]

Hanssen, G. K., Šmite, D., Moe, N. B., 2011. Signs of Agile Trends in Global Software Engineering Research: A Tertiary Study. In *6th IEEE International Conference on Global Software Engineering Workshops*, Helsinki, Finland, pp. 17-23.

Hildenbrand, T., Geisser, M., Kude, T., Bruch, D., Acker, T., 2008. Agile Methodologies for Distributed Collaborative Development of Enterprise Applications. In *2008 International Conference on Complex, Intelligent and Software Intensive Systems*, Barcelona, Spain, pp. 540-545.

Hossain, E., Bannerman, P. L., Jeffery, R., 2011. Towards an Understanding of Tailoring Scrum in Global Software Development: A Multi-case Study. In *2011 International Conference on Software and Systems Process*, Honolulu, HI, pp. 110-119.

Kamaruddin, N. K., Arshad, N. H., Mohamed, A., 2012. Chaos issues on communication in Agile Global Software Development. In *2012 IEEE Business, Engineering & Industrial Applications Colloquium*, Kuala Lumpur, Malaysia, pp. 394-398.

Korkala, M., Abrahamsson, P., 2007. Communication in Distributed Agile Development: A Case Study. In *33rd EUROMICRO Conference on Software Engineering and Advanced Applications*, Lübeck, Germany, pp. 203-210.

Larman, C., Vodde, B., 2009. *Scaling Lean & Agile Development. Thinking and Organizational Tools for Large-Scale Scrum*, Addison-Wesley. Boston, MA.

Niinimäki, T., 2011. Face-to-face, Email and Instant Messaging in Distributed Agile Software Development Project. In *6th IEEE International Conference on Global Software Engineering Workshop*, Helsinki, pp. 78-84.

Noll, J., Beecham, S., Richardson, I., 2010. Global software development and collaboration: barriers and solutions. *ACM Inroads Magazine*, 1(3), p.66-78.

North, D., 2006. Behavior Modification. The evolution of behavior-driven development. *Better Software Magazine*, Issue March.

Paasivaara, M., Durasiewicz, S., Lassenius, C., 2009. Using Scrum in Distributed Agile Development: A Multiple Case Study. In *4th IEEE International Conference on Global Software Engineering*, Limerick, Ireland, pp. 195-204.

Vallon, R., Strobl, S., Bernhart, M., Grechenig, T., 2013 (in press). Inter-Organizational Co-Development with Scrum: Experiences and Lessons Learned from a Distributed Corporate Development Environment. In *14th International Conference on Agile Software Development*, Vienna, Austria. (Accepted for publication February 2013).