Software Development Using Agile and Scrum in Distributed Teams

Youry Khmelevsky
Xitong Li
Stuart Madnick

Working Paper CISL# 2017-02

February 2017

Cybersecurity Interdisciplinary Systems Laboratory (CISL)
Sloan School of Management, Room E62-422
Massachusetts Institute of Technology
Cambridge, MA 02142

Software Development Using Agile and Scrum in Distributed Teams

Youry Khmelevsky*
Computer Science, Okanagan College
Kelowna, BC Canada
Email: ykhmelevsky@okanagan.bc.ca

Xitong Li
École des Hautes Etudes
Commerciales de Paris, France
Email: lix@hec.fr

Stuart Madnick
Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA USA
Email: smadnick@mit.edu

*Also Affiliated with UBC Okanagan, Canada

***Abstract*—Agile software development practices, like Scrum, that allow teams to focus on delivering product and improved communication has made it one of the easiest and best software development techniques. On the other hand, such agile methods have been designed for collocated software development and are thus not directly applicable to distributed agile development.**

In this paper, we present findings from case studies and real life distributed Agile and Scrum projects conducted since 2011, as well as the challenges and benefits the case projects reported and unique lessons learned from them.

I. Introduction

A study that involved 58 practitioners in 23 organizations over 4 years reported two interesting pieces of data [1]: (1) self organizing scrum teams naturally perform a balancing act between freedom and responsibility; cross functionality and specialization, and continuous learning and iteration pressure, and (2) self organizing teams have its members assume some well defined roles, such as mentor, coordinator, translator, champion, promoter and terminator. They are not prescribed explicitly as part of any of the agile development philosophies, but they arise as part of successful use of the methodology [1].

In 1995, Sutherland and Schwaber presented the first paper describing the Scrum methodologies [2]. They collaborated during the following years to merge the writings, experiences, and industry best practices into what is now known as Scrum. Scrum focuses on project management institutions where it is difficult to plan ahead [3]. Agile software (SW) development has become a popular approach to the engineering of software systems in the commercial world. A project must employ Agile development methods and must also fit within an Agile product development system: The development organization must be willing to practice refactoring, or lose the benefits of Agile SW development. The software itself must be Agile, lending itself to rapid incremental deliveries and must be architected accordingly [4].

Dingsoyr et al. [5] in 2012, based on their literature review on agile software development, found that there are ”very few on the Scrum development process” and explicitly call for academic research on Scrum. Ramesh et al. [6] in 2006, on the other hand, pointed out that traditional agile methods rely on informed processes to facilitate coordination, which is different from what distributed software development traditionally requires (i.e., formal mechanisms for coordination). Therefore, the product provides a unique context for us to conduct a case study that evaluates the applicability of the Agile principles and Scrum practices in a real-world distributed global software development (GSD) project and examines the inconsistencies between theoretical assumptions behind Scrum and practical observations. Based on our experiences, we found that although the distributed agile development (DAD) was to be based on ”Twelve Principles of Agile Software” [7], they were not rigorously used by all members of the team and do not fit well into the global software development (GSD). Parts of the reasons are that ”GSD typically involves stakeholders located in different time zones and geographic locations, from different national and organizational cultures, using different and, at times, unreliable technologies to collaborate” [8]. Such temporal, geographical and socio-cultural distances can result in significant communication, coordination and control challenges that need to be overcome for the benefits of GSD to be realized.

In recent publications, we found discussions about Agile at global companies and in distributed teams [9], [10], [11], [12]. Some authors believe, “that if agile is to thrive over the next 10 years then it not only has to work in a distributed environment, but it has to work well in order to deliver the most value to an organization” [9]. The following issues are widely discussed:

- Many organizations in local proximity really work in distributed environment.
- Many organizations don’t have an “utopian situation”, when they have one team, one product owner and one location. It is common to have multi-team, multi-location, multi-business area and multi-time zone (see Section II and Section III for examples from our projects) [9].
- In many cases the “Agile process” is really a “Water-scrumfall”, where a “coding factory” model is present. That means you send a code specification and wait for the code to come back. This is not a true Agile way.
- The biggest problems are: lack of clear (1) goals; (2) no rationale; (3) absence of context; (4) a high risk of confusion during start of the project.
- Some authors even recommend a de-Agile approach for the distributed teams [10] by taking out processes that don’t make sense and tweaking those that need to be modified to suit your needs. In other words,

removing the sense of being distributed (additional communication responsibilities, being open and available).

- Team size should be from 10 to 15 people and have mix of talent at every location (no centralizing one type of work at one location) [10].
- Keep fairly equal workloads in distributed teams. SWAT (Subjective Workload Assessment Technique) can be used.
- Pair Programming or show-and-tell hour or at least regular code review by other team members should happen.
- Understanding time and cultural differences is crucial.

Starting from 2005, agile practices, Scrum and Extreme Programming (XP) SW development methodologies were used within capstone and applied research projects with remote and local industrial clients at Okanagan College (OC) and University of British Columbia Okanagan campus (UBC O) [13], [14]. We were able to employ agile principles in a commercial SW development project.

The contribution of this paper consists of practical recommendations, which can be used by other companies that are planning to use Scrum and/or Agile practices within distributed projects. In our case study we used related research papers and internet publications, but we found inconsistencies between theoretical assumptions and our practical experiences, for example related to special scrum training before the project begins (which is almost impossible due to very short project duration and limited project budget).

II. Context of a Case Study

The project teams usually contained team members from the US, Canada, Western and Eastern Europe. The project structure can be shown in this way: the customers with employees and volunteers are located in the US, Germany and Eastern Europe; main SW development contractors are located in the US with employees in Canada and sometimes in Eastern Europe; the offshore SW Development subcontractor were located in Eastern Europe or in Canada.

A business Case Study project was started in April and was planned to be finished in middle of July by using Agile project development principles. During the first project development weeks, the development team found that the customer was unable to provide graphical design for all of the elements of the web-system, which was not included in the project development tasks and "generated" some new ideas "on the fly" during the project development. In the next following weeks, developers found many inconsistencies in the project specifications and project requirements. Due to such problems, the "Stage 0" was extended from 2 weeks to almost 1 month. The project team made a decision to use Scrum practices additionally to general Agile Manifesto principles, to reduce the project meetings' time and to improve performance within teams.

Originally the main contractor and offshore development team planned to finish the project in two months in three iterations (Stage 0 – Stage 2) and then support the finished system for about 8–12 months. The Stage 2 was planed mostly for the refactoring, final testing and bugs fixing only. When the project was started, the development team found inconsistencies between graphical design, list of requirements and project specification, which was developed in Eastern Europe in the past.

A. Core Scrum roles and relation to the project team members:

"A Development Team is made up of 3-9 people with cross-functional skills who do the actual work (analysis, design, develop, test, technical communication, document, etc.). The Development Team in Scrum is self-organizing, even though they may interface with project management organizations (PMOs)" [3].

In the case study project:

- **Product Owner** - represented by the customer.
- **Development Team:** represented by the offshore development team of 3–4 team members and by 3–5 customer's volunteers (and advisers) in Eastern Europe for beta testing, web-site design and development source code audit.
- **Scrum Master** — main contractor from the US.

Ancillary roles:

- **Stakeholders** — the project customer and his advisers/volunteers, which will use the final product in the business environment (from 3 to 5 stakeholders).
- **Managers** — contractor and subcontractor managers in the US and Eastern Europe. The project development manager — subcontractor in many cases share his role as a development team manager, beta tester, and developer.

B. Meetings

Daily Scrum: The project team scheduled a 15 minutes Scrum meeting mostly every day. The project team didn't have a Daily Scrum meeting only when customer or subcontractors asked to skip one or two meetings. The team had several instances when team members were tired or upset by the high amount of problems generated by this project.

During the daily Scrum, the project team discusses the following topics [3]:

- What have you done since yesterday?
- What are you planning to do today?
- Any impediments or stumbling blocks?

After the meeting the project team usually had a Product Backlog Grooming for about 60 minutes or even longer to estimate the existing backlog, refining the acceptance criteria for individual stories, and breaking larger stories into smaller stories.

Scrum of Scrums: Instead of the Backlog grooming meeting, the team managers could have a Scrum of Scrums on-line by using Skype's call conference (in later distributed projects in 2013–2016 we used WebEX as well). It depended on the project problems and development team performance and were decided on the Daily Scrum meeting. The duration of the Scrum of Scrums could be up to 1 hour or even 1.5 hours, but only customer and designated person from each team attended Scrum of Scrums on-line or in person.

Sprint planning meeting: At the beginning of the sprint cycle, the project team hold a ”Sprint planning meeting”, but usually informally.

Sprint review meeting: At the end of the Stages the project team had a ”Sprint review meeting”. Sometimes it was done more often or for longer periods of time, when the development team had many bugs or mistakes — at the end of a week or even several consecutive days at the end of the Stages.

Sprint retrospective: All project team members discussed the past sprint problems, suggested continuous process improvements. It went from 30 minutes to several hours. The main questions were the following [3]:

- What went well during the sprint?
- What could be improved in the next sprint?

Project Artifacts: For Product Backlog MS Excel and MS Word documents were used and for the web-site design images developed in PhotoShop, the DropBox was used as well. For the Sprint Backlog the dotProject was used initially, but starting from 2014 we used new SW project management web based tool Jira; for the Project increments or different stages of the web-system system the GoDaddy web-hosting was used additionally to the development versions, hosted in Eastern Europe. Instead of burn down chart the project team used a Gantt Chart with %% completion from dotProject or Jira. Later we moved our design, project documentation and project management process to Jira, Confluence, Slack, Bitbucket and Latex. We introduced Bamboo and Jenkins continues integration as well.

III. LESSONS LEARNED

- If the subcontractor never used any Agile practices in the past, the learning curve will take several weeks to months. This learning time is very critical for the project success and must be completed as soon as possible with the help of experienced team members. The experienced team members should be ready for resistance from the subcontractor’s team members against of new Agile practices, even if they understand the usefulness of the Agile practices and high probability of the project success by using Agile.
- If a team member(s) already used Agile practices such as XP or Scrum in the past successfully, he/she should be a Mentor and/or Champion/Promoter.
- In the case study we found a strong resistance from the subcontractor’s manager during the first month of the project. The project was in a critical situation and the ”Terminator” role had to be used [1] against the subcontractor manager. After a few weeks negotiations the project went in the right direction again. It is strongly advised to be ready for conflict situations, especially in the first few weeks of the project.
- The productivity of the project development team increased drastically, especially related to many hours of Skype conferences and discussions. After Scrum implementation, 1-2 hour online project meetings were reduced to 15-20 minutes Daily Scrums and 30-60 minutes Product Backlog Grooming meetings, depends on the project problems.
- It’s almost impossible to have face-to-face interactions between team members in distributed teams, but regular Sprint/Agile meetings can be done by using modern and free VoIP technologies (Skype, WebEx, Google Talk, etc.).
- If team members are located on different continents, team members, especially a Scrum Master or Mentor/Coordinator, should be ready to work in early morning at 6 or 7 am, or late at night from 10 pm to 1 am or even at 2 am and later. We tried to use optimal time for everybody (for example 9 pm TZ +2 and 11 am Pacific Time), but in 1 month we found, that for better project performance we need to create more comfort working time for the developers (at least 5 pm TZ +2) and less comfort time for other team members (6 am or 7 am Pacific Time).
- It’s impossible to have a special ”Proper Scrum training” in the beginning of the project if the team has never used Scrum or Agile before, because this will require extra time and resources, and will increase resistance from ”conservative” team members. The best approach is to train team members during the Daily Scrum meetings and/or more often sprint review meetings, when everybody could suggest or advise modifications for the development process. Support from other team members during meetings is crucial.
- Frequent visits of the development personal from collaborating sites are almost impossible. Even one intercontinental visit can cost a substantial part of the project budget, which is unacceptable for small or mid-size projects.
- Multiple communication tools must be used by a distributed team. The stress should be made not on videoconferencing, but on the project management and project collaboration tools. In our case we found that shifting from verbal communication by Skype and videoconferencing to even a very simple and free dotProject and DropBox improved project performance and project communication, especially by using tickets, tasks, file sharing, meetings’ minutes, meetings’ scheduling and files sharing. Traditional telephones were almost never used, but only for local communications.

IV. CONCLUSION AND RECOMMENDATIONS

In this paper we presented a case study and the following projects’ experiences, and lessons learned from distributed projects on the application of Scrum and Agile practices. We also would like to give the following recommendations for the SW development projects in distributed environments:

- It must be real collaboration and team work in distributed environment without any cultural barriers (see more in [9]). In our case almost every one of our teams in different projects had representatives from different countries. Different cultures have different approaches to the same problems/issues. It must be taken in account.
- Distributed projects lack the face-to-face communication, but we don't have any other choice but to replace it by "rich" communication channels and by "simulation" of high speed high quality face-to-face discussions. Since 2005 (for more than 11 years) we have regular face-to-face local meeting, weekly or bi-weekly telephone/skype/WebEx "rich" meetings with overseas collaborators as well as regular local meeting, when somebody doesn't have time for the face-to-face meeting. We always plan at least 1 week local project work per year with our distributed collaborators.
- Do real agile with overseas or distributed collaborators and avoid a "coding factory". Brainstorming and ideas exchange is crucial in true Agile.
- Frequent deliveries and daily integration. It's hard to achieve, but this approach gives the best results.
- Avoid "Agile but" or don't invent hybrids or "Water-scrumfall".
- Scrum is based on Agile, but they are not the same. Scrum is one of the well-know agile approaches only. Some authors believe that Scrum can't be used for distributed teams, but Agile can. In our many distributed projects we had situations when we introduced a manager to meet our deadlines and to finish our project on time and within budget. "Many supporters of agile see the role of a Project Manager as defunct, an irrelevance – if you are looking to use agile in a distributed context it is a necessity" [9]. "Scrum of scrums" really doesn't work and it didn't work for us in several of our distributed projects. We had to introduce a Project Management role instead of Scrum Master and Scrum of Scrum Master in several critical situations.
- No excuses for starting off a project with incomplete or inaccurate high-level requirements. We've seen fake "Agile Gurus", who tried to convince us that Agile doesn't need any planning or any requirements. Late instead of early top-level design and high-level requirements create many problems and communication issues in the team(s), especially a huge waste of resources and time in the project. User Stories must be supported by additional technical documentation (Use Cases, models, diagrams, requirements and specifications).
- Design and development tools are important. "Poor tools and a poor environment can lead to a poor project" [9]. Provide the right tools and training [10]. Some examples: WebEx, Google Docs, Slack, Jira and many other.
- Root cause analysis and continuous improvement, inspection and adaption are important for success.
- Plan frequent demos and retrospective meetings.
- Reorganize teams only as a last resort, especially because communication and track of everyone's progress in a distributed team is more difficult.
- Quality control and frequent deliveries are especially important.

Acknowledgments

Our thank to Amazon Web Services, Inc. for supporting our capstone software engineering and our research projects by AWS Grants for Research and Education, as well as to Atlassian for supporting us by community licenses for Jira project management and Bitbucket cloud tools.

Our thanks to IBM Education for supporting us with IBM Rational SW Architect licenses in student projects, as well as to VMWare, Microsoft and Oracle.

References

[1] J. Langford and R. Ortega, "Machine learning and algorithms; agile development." *Commun. ACM*, vol. 55, no. 8, pp. 10–11, 2012.

[2] J. V. Sutherland and K. Schwaber, "Business object design and implementation: Oopsla'95 workshop proceedings. the university of michigan," ISBN 3-540-76096-2, Tech. Rep., 1995.

[3] K. Schwaber, *Agile project management with Scrum.* Microsoft Press, 2004.

[4] F. Glaiel, "Agile project dynamics: a strategic project management approach to the study of large-scale software development using system dynamics," Ph.D. dissertation, Massachusetts Institute of Technology, 2012.

[5] T. Dingsøyr, S. Nerur, V. Balijepally, and N. B. Moe, "A decade of agile methodologies: Towards explaining agile software development," *Journal of Systems and Software*, vol. 85, no. 6, pp. 1213–1221, 2012.

[6] B. Ramesh, L. Cao, K. Mohan, and P. Xu, "Can distributed software development be agile?" *Communications of the ACM*, vol. 49, no. 10, pp. 41–46, 2006.

[7] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas. (2015) Manifesto for agile software development. [Online]. Available: http://agilemanifesto.org/

[8] P. L. Bannerman, E. Hossain, and R. Jeffery, "Scrum practice mitigation of global software development coordination challenges: A distinctive advantage?" in *System Science (HICSS), 2012 45th Hawaii International Conference on.* IEEE, 2012, pp. 5309–5318.

[9] K. Richards. (2014, May) Distributed agile: 8 ways to get more from your distributed teams. [Online]. Available: https://www.infoq.com/articles/distributed-agile-8-ways

[10] S. Joshi. (2012, January) Agile development - working with agile in a distributed team environment. [Online]. Available: https://msdn.microsoft.com/en-us/magazine/hh771057.aspx

[11] S. Parekh. (2015, February) Collaboration techniques for large distributed agile projects. [Online]. Available: https://www.thoughtworks.com/insights/blog/collaboration-techniques-large-distributed-agile-projects

[12] D. A. 2.X. (2014, October) Geographically distributed agile teams. [Online]. Available: http://www.disciplinedagiledelivery.com/agility-at-scale/geographically-distributed-agile-teams/

[13] Y. Khmelevsky, "Sw development projects in academia," in *Proceedings of the 14th Western Canadian Conference on Computing Education.* ACM, 2009, pp. 60–64.

[14] Y. Khmelevsky, V. Ustimenko, G. Hains, C. Kluka, E. Ozan, and D. Syrotovsky, "International collaboration in sw engineering research projects," in *Proceedings of the 16th Western Canadian Conference on Computing Education.* ACM, 2011, pp. 52–56.