Why the Development Outcome Does Not Meet the Product Owners' Expectations?

Conference Paper *in* Lecture Notes in Business Information Processing · May 2015

DOI: 10.1007/978-3-319-18612-2_8

4 authors:

Timo O. A. Lehtinen

Academy of Finland

10 PUBLICATIONS **44** CITATIONS

SEE PROFILE

Risto Virtanen

Aalto University

2 PUBLICATIONS **3** CITATIONS

SEE PROFILE

Ville Tomi Heikkilä

Aalto University

16 PUBLICATIONS **61** CITATIONS

SEE PROFILE

Juha Itkonen

Aalto University

31 PUBLICATIONS **193** CITATIONS

SEE PROFILE

Why the Development Outcome Does Not Meet the Product Owners' Expectations?

Timo O.A. Lehtinen, Risto Virtanen, Ville T. Heikkilä, Juha Itkonen,

Department of Computer Science, Aalto University School of Science, P.O. BOX 15400,

FI-00076, Aalto, Finland
{timo.o.lehtinen, risto.virtanen, ville.t.heikkila, juha.itkonen}@aalto.fi

Abstract: Many software development projects fail due to problems in requirements, scope, and collaboration. This paper presents a case study of the mismatch between the expectations of Product Owners and the outcome of the development in a large distributed Scrum organization. The data was collected in retrospective meetings involving a team of Product Owners and two software development teams. A focused root cause analysis of the problem "Why the expectations of Product Owners do not meet the outcome of development teams?" was conducted. The analysis aimed at explaining why the problem occurred and how the causes were related to one another. The outcomes were analyzed both quantitatively and qualitatively. Our results illustrate the challenges of implementing the Product Owner role in the context of complex, high-variability requirements and distributed development. We highlight the importance of true collaboration, effective requirements specification activities, and sufficient resources for the Product Owner role.

Keywords: Scrum, Product Owner, root cause analysis, software process improvement, requirements engineering, global software development

This document is the camera ready version published by Springer. The final publication is available at link.springer.com, http://link.springer.com/chapter/10.1007/978-3-319-18612-2_8

1 Introduction

Matching the expectations of customers and software development outcomes is a fundamental business issue and research objective in the software engineering domain. The mismatch between customer expectations and software development outcomes has caused many software projects to fail, see e.g. [1].

The identified key factors of successful software product development [2] include the good requirements and flexible collaboration between the customers and developers. Respectively, the common factors of software project failures [3] include the difficulties with the customers, requirements specifications, and collaboration over the stakeholders. Most of the underlying causes of the success and failure have also been presented to be context dependent [3] and thus the in-depth analysis and dissemination of each new case is important in order to create generalizable knowledge of software engineering problem causes. Prior studies have mostly ignored the contextual differences in software projects [4].

This paper disseminates a case study on the problems in the collaboration between Product Owners (PO) and software development teams in a large distributed Scrum organization. The problems are analyzed from both the customer and development perspective. The main objective is to disseminate the causes of the mismatch between the software development outcomes and POs' expectations. In addition, the solutions to identified causes are proposed and evaluated. This study answers the following two research questions:

RQ1: Why the expectations of Product Owners do not meet the outcome of development work?

RQ2: What problems were perceived the most important to control in order to minimize the risk for a failure in the Product Owner expectations?

2 Background and Related Work

The way the expectations of a PO are communicated to the developers in a Scrum team is well-documented [5]. The PO creates and prioritizes the product backlog items. During the sprint planning meeting, the team creates a feasible sprint backlog for the following sprint. During the sprint, the team and the PO have a constant dialogue regarding the implementation of the backlog items. At the end of the sprint, the Scrum team and relevant stakeholders review the sprint outcome. Thus, the PO communicates her expectations to the team at the beginning, during and at the end of the sprint. Any mismatches are identified easily and amended quickly.

Although the PO is extremely important for the success of Scrum development, there has been little research on the causes of failed expectations in Scrum. Some studies exist on the topic of implementing Scrum in global software development. Lee and Yong [6] found that the distribution of development created misunderstandings between stakeholders and difficulties in adjusting priorities. They suggested on-site customer representatives as a solution for those problems. Paasivaara et al. [7] studied the challenges in scaling the PO role in globally distributed Scrum projects. They identified on-site PO representatives, teaming POs, frequent communication, and clearly communicated priorities as the means for a successful PO function in large distributed settings.

Moe et al. [8] studied teamwork in a Scrum project. They found that developers expected the PO to be able to provide answers to their questions on short notice. However, the PO lacked clear understanding of what the system was supposed to do and he was not always able to answer the developers' questions. Subsequently, the developers were often unsure of what they were supposed to do. The lack of planning and a rapidly changing environment were also identified as causes for the developers working on tasks that did not originate from the sprint backlog. Moe et al. identified the insufficient coordination of information dissemination and the lack of responsibility for the overall technical solution as the causes of failed PO expectations.

Strode et al. [9] propose that having an on-site customer representative leads to highly efficient coordination, but a team-external customer representative leads to more complex coordination needs. They suggest that a team member should take the role of an explicit coordinator if an on-site customer representative is not available.

According to Bjarnason et al. [10], overscoping refers to setting a larger iteration scope than the available resources allow. They identified the following causes of overscoping in large-scale software engineering [10]: Continuous requirements inflow from multiple channels, no overview of available resources, low development team involvement in the early planning phases, unclear vision of the overall goal and scope, and deadlines dictated by the management. They also found that overscoping caused failures to meet the customers' or clients' expectations.

3 Methodology

The overall research method was a case study [11]. Root cause analysis [12] was used in the data collection and analysis.

3.1 Case Company

The case company was a large distributed software development company which employed approximately 800 employees. The company developed complex systems that were integrated into customer specific systems with varying business logic. The company representatives reported that they had started to use the Scrum method [5] about a year prior to this study. The organization had 30 members who were software developers, Scrum Masters, or POs. The developers were split into two software development teams. Both teams developed independent sub-systems for the software product. The teams followed two-week development sprints, conducted daily stand-ups, sprint demonstrations and retrospectives.

Four POs conveyed the customer needs to the development teams. The organization was divided into three European countries, each having one local PO and members from both teams. The fourth PO was responsible for quality assurance. Both teams worked with all four POs. One reason for this was the regulatory localization: The product needed to fulfill country specific regulations and therefore it was useful to have multiple POs, each being responsible for a different set of regulations.

The case was purposively selected [13] as it was a rich source of data. It enabled us to accumulate understanding about the reasons for failed PO expectations in a large, distributed Scrum organization.

3.2 Data Collection with Root Cause Analysis

The data collection was made by focused retrospective meetings following the ARCA root cause analysis (RCA) method [14] and tool [12]. The organization had used such an approach in retrospectives preceding our case study, which enabled us to collect the data and observe the analysis in the real context of use.

The case study was based on on-going research collaboration including frequent knowledge sharing. The case study started with a 60-minute focus group meeting with one PO and one Scrum Master. The goal was to define the main problem in the soft-

ware development activities. These discussions resulted in understanding that the expectations of POs did not meet the outcome of the development work.

The formal data was collected in three face-to-face retrospective meetings (3 x 60 minutes in total). Each retrospective meeting considered the following question: "*Why expectations of Product owners do not meet with the implemented functionality?"* The first and second retrospective meetings were conducted with a development team (three and five participants, respectively). The third meeting was conducted with the POs (four participants). One of the Scrum Masters chaired all of the retrospective meetings. All three retrospectives were conducted in succession during one day and the participants did not communicate between the retrospective meetings.

The teams used the RCA method and software tool in the retrospective meetings to detect the causes of the main problem and collaboratively create a cause-effect diagram that expresses the problem causes and their causal relationships in an electronic format. The method was used in the following way. First, the retrospective participants were given 5 minutes to individually write down causes related to the main problem. Thereafter, they presented their findings to others. Then, the participants were given another 5 minutes to add additional causes and sub-causes for the previously detected problems, followed by discussion. During the discussion, further additional causes were entered to the cause-effect diagram when detected. The participants also considered how the causes were related to one another. This helped them to express how the conflicts between the POs' expectations and the development work outcomes came to be. At the end of each retrospective, the participants voted on the causes that needed immediate corrective actions and were also feasible to solve.

The case analyses resulted in four corrective actions. These actions were evaluated by the organization members during the "improved" software development activities. In order to evaluate the process improvement impact, the company representatives conducted three 60 minutes interviews. One Product Owner and two developers, one from each team, were interviewed.

3.3 Data Analysis

The data analysis was conducted with the company representatives. The raw data from each retrospective meeting was analyzed separately first. The detected causes were classified applying the process area and problem type classifications proposed by Lehtinen et al. [3]. This made it possible to generalize the causes and map them to software development process areas, which helped to express what happened in the affected software processes.

After the detected causes were classified, the process interconnections were analyzed. A new modularized cause-effect diagram was made from each retrospective data separately. The diagram included process areas and cause types separated into "local causes" and "bridge causes" [3]. The bridge causes explain the external causal relationship between two process areas and the local causes explain the outcomes of a process area internally. The most controllable problems, voted by the retrospective participants, were also emphasized in the modularized diagram. This helped to communicate the analysis results to the members of the organization and decision makers, which was considered highly important because the aim of the company was to rec-

ognize feasible targets for process improvement, develop action proposals, and finally implement the corrective actions.

After the results from each retrospective were analyzed, the results were synthesized into a combined analysis. This made it possible to conclude the common problems and generalize the results to cover the whole organization. The reliability of the combined results was expected to be high because they were identified in each retrospective meeting separately. Corrective action interview results were analyzed by dividing the answers into "positive comments" and "comments why the corrective action should be improved".

4 Results

This section presents the results of the case study. First, the analysis outcomes from the retrospective meetings are presented. Thereafter, we describe the synthesis of the results analyzing similarities and differences between the outcomes of the different teams.

4.1 Retrospective Outcomes of Development Team 1

Development Team 1 (D1) concluded that the main problem was caused by insufficient requirements, which were not specific enough to be understood correctly. They were made with insufficient guidance from the customers. The requirements were also affected by the lack of collaboration, the lack of values and taking responsibility, and the lack of available resources.

D1 explicated causes from the development work. The team members did not communicate actively enough during the sprints. They did not ask for clarifications to unclear requirements. The team members also explained that they did not communicate enough directly with the POs, but too much thorough the Scrum Master. Additionally, the team members reported that development work usually included challenging tasks.

Furthermore, D1 presented that the software testing suffered from missed deadlines and insufficient requirements. Third parties provided test data for software testing activities. Occasionally, the third parties provide the data too late, which caused delays in the software testing. Insufficient requirements caused inaccurate workload estimates, which caused missed deadlines in software testing.

The lack of information flow between the sales & requirements and development work was identified as an important target for process improvement. Similarly, the collaboration practices were emphasized important to be solved.

4.2 Retrospective Outcomes of Development Team 2

Development Team 2 (D2) identified many similar causes than D1. D2 concluded that the main problem was affected by insufficient communication between POs and developers, which caused unclear specifications with vague priorities. The lack of

knowledge, lack of values and taking responsibility, lack of collaboration between Product Owners and developers, lack of processes for communication, and lack of resources explained why the requirements were insufficient. In addition to D1 findings, D2 members explained that it was also the work practices of prioritizing everything as "must have" which caused the main problem.

The inactive communication between the POs and developers during the sprints was also elaborated as one of the important causes of the main problem. One team member found that "Dialogue might be missing during sprint", which was then explained by stating "No tradition in ongoing communication between POs and Developers" and "Product Owners [are] not involved as much as they should".

The lack of collaboration between the sales and requirements process and development work was concluded as an important target for process improvement. Furthermore, D2 proposed that the collaboration and development process needed to be improved.

4.3 Retrospective Outcomes of Product Owner Team

Similarly to both development teams, the POs concluded that the requirements were insufficient. They explained that the problems in the sales and requirements process were caused by the lack of knowledge, lack of values and taking responsibility, lack of collaboration, lack of work practices, lack of communication processes, and lack of resources. The POs thought that creating good requirements was difficult. The organization employed third parties to create the requirements and occasionally the POs could not understand those requirements.

The POs detected the causes of the main problem similar to the ones detected by the development teams. These included the lack of knowledge, lack of collaboration, and task difficulty. In contradiction, they explained that the quality of the development work outcome was occasionally insufficient. The POs emphasized that the lack of collaboration with the developers was the most important cause in explaining why the main problem occurred. They claimed that the organization members were inactive in collaboration. The POs stated that "Developers have not asked for meetings" and "No initiatives from POs or Scrum Masters to have meetings and discuss complex use cases". They also stated that the language barrier and geographical distances negatively affected the collaboration activity. The POs elaborated that the development work was not properly monitored during the sprints.

The POs stated that the main problem was also caused by management problems and the organization in general. They claimed that the knowledge of managers was limited and that the management suffered from the lack of collaboration, lack of communication processes, and lack of resources. The POs also identified problems in the whole organization. They claimed that the organization members did not know how to do Scrum. Finally, they explained that the complexity of the product caused problems for most members of the organization.

Insufficient requirements were identified as the most important targets for process improvement. Similarly, the lack of processes, insufficient task outcomes, the lack of taking responsibility, and the lack of resources in sales & requirements were important to be improved. The POs also emphasized organization-wide problems for

process improvement activities. These included the lack of instructions and experience, and the complexity of the existing product.

4.4 Synthesis of the Individual Retrospective Results

Fig. 1 summarizes the causes that were identified in all retrospective meetings. It also shows the causes that were perceived as feasible targets for process improvement. The missing collaboration during the software development activities is one of the main problem causes in the organization. The output from the sales and requirements process was also found to be insufficient. Following from these causes, the developers did not know what to do and what to test. The members of the organization perceived that they did not receive enough instructions.

Furthermore, the participants presented that the lack of values and taking responsibility was a problem in the sales and requirements process area. The sales & requirements process area also suffered from the lack of resources. The feasible process improvement targets were mostly similar among the teams. Requirements understanding needed to be increased. The collaboration was also a feasible target for process improvement. A total of four corrective actions were finally developed. First, the POs were asked to participate in the teams' daily stand-up meetings. Second, the organization hired two new employees to work with the requirements specifications and usability designs. Third, the POs increased their mutual collaboration during the development sprints. Fourth, the POs started to regularly participate in sprint planning activities.

Only the first corrective action was not implemented. In the interviews, the impact of the other corrective actions was concluded as significant. On the other hand, it was noted that these corrective actions do not solve the main problem completely.

5 Discussion

This section answers our research questions and considers the implications for practitioners. At the end of the section, we also discuss the limitations of this case study.

5.1 Answers to the Research Questions

The main research objective of this case study was to conclude the problems between the outcome of Scrum teams and POs in large distributed Scrum organization. Table 1 summarizes the problems and detected causes including their solutions in relation to the related work.

RQ1: Why the expectations of Product Owners do not meet the outcome of development work? The first research question contributes to prior studies on software project failures by considering the problems to meet the expectations of customers. Regarding our results, the lack of collaboration between the customers and software

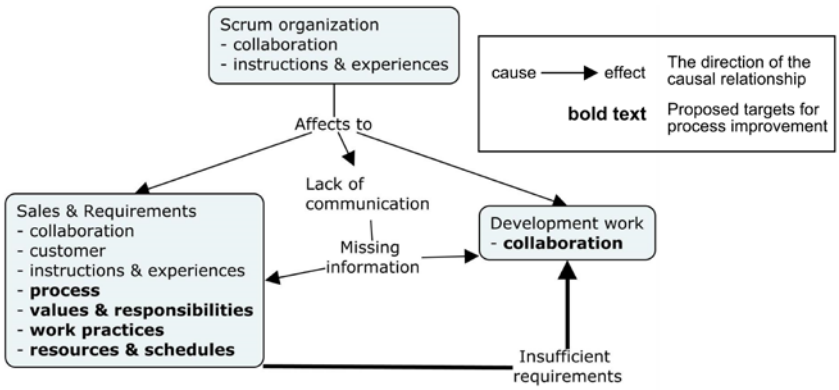

Fig. 1 Synthesis of the causes detected in the retrospective meetings.

development teams causes such failures. In the case organization, the POs had difficulties communicating and prioritizing the requirements to the development teams because the requirements were not clear for the POs themselves. This was caused by the lack of PO resources and the use of third parties in eliciting and defining the requirements. The geographical distance, language barriers, and inactive software developers exacerbated the problems. Clarifications for the requirements were neither requested during the development work.

The problems of the case organization do not seem to be problems with the Scrum method, but problems in implementing it. The in-depth data analysis including the problems and corrective actions of the case revealed that the PO role was not fully implemented. The POs did not belong to the Scrum teams, but were isolated customer representatives occasionally participating in sprint planning and review meetings. They did not actively participate in the sprint planning, daily stand-up, and sprint retrospective meetings. These causes partially explain the failed PO expectations. The company representatives concluded that the POs were not parts of the development teams and subsequently the developers did not know which PO was responsible for which requirements.

RQ2: What problems were perceived the most important to control in order to minimize the risk for a failure in the Product Owner expectations? The second research question contributes to the research problem of how to successfully describe the requirements and communicate them to the Scrum teams. In our result, lacking instructions and experience, values and responsibility, collaboration, methodology, and resources and schedules are all important targets for process improvement.

The case organization members perceived that increasing the amount of collaboration between the POs and software developers would improve the outcome of the sales and requirements process. They also concluded that this would require changes in the methods used by the POs. The POs should have met with the development teams more often. Especially they should have actively participated in the sprint planning meetings, as was proposed in the corrective actions developed in the case. Additionally, the POs should have had frequent meetings together. This would have helped them to review the requirements before they were introduced to the development

Table 1. Comparison with the related work

Problem	***Causes in this study***	***Causes in related work***
Insufficient collaboration between the PO and the team(s).	• The distribution of development and heavy PO workload caused inactive collaboration during the sprints, which led to misunderstandings between the stakeholders.	• The distribution of development created communication misunderstandings between stakeholders [6]. • Low development team involvement in early planning phases [10] caused overscoping.
Challenges in adjusting the priorities.	• POs' heavy workload caused lack of resources to do prioritization. • POs' lack of knowledge caused unclear understanding in the development teams.	• Difficulties in adjusting priorities [6]. • The continuous requirements inflow from multiple channels [10]. • PO lack of understanding of product features [8]. • No overview of software development resource availability [10].
Developers working on the wrong tasks	• The development work was not properly monitored.	• Rapid change and lack of planning [8]. • Insufficient coordination of information dissemination and the lack of responsibility for the overall technical solution [8].
Lacking understanding of requirements.	• The third parties provided unclear requirements for the POs which caused problems reaching a consensus with the developers.	• Unclear vision of the overall goal, and scope and deadlines dictated by the management [10].
Solutions	***Solutions in this study***	***Solutions in related work***
Local customer representatives	• On-site customer representatives.	• On-site customer representatives [6], [7].
Product Owner resources	• A PO team. • PO meetings. • Increasing the number of POs. • Active PO participation in sprint planning.	• A PO team [7]. • Team internal coordinator if an on-site PO not available [7], [8]. • Frequent communication [7]. • Clearly communicated priorities [7]. • POs with technical background [7]. • PO pairwork [7].

teams. Furthermore, sharing instructions and project experiences over the members of the organization were perceived as important for increasing the match-rate between the expectations of the POs and outcomes of the development teams.

Additionally, the organization needed more human resources. The POs were far too busy to handle the stream of all customer requirements and to communicate them in the large-scale distributed organization. Therefore, two new employees were acquired to the software organization. A new PO was hired to manage the product backlog, including the requirements elicitation, tracking, and prioritization. This was an improvement towards proper Scrum implementation. It also improved the software development outcomes. The organization members explained that the new PO had a positive effect on the clarity of requirements. The requirements were also provided well in time. Furthermore, a new employee was hired to ensure that the user interface was designed according to the requirements. This helped the organization to increase the match-rate between the implemented user interface and the customer expectations. This was a "great help" as was concluded in the post-interviews.

5.2 Implications for Practitioners

In a large distributed software organization, implementing Scrum successfully is challenging. It requires a fit between the customers and software development teams, which is highly dependent on the success of implementing the PO role. Scrum may also require heavy investments in collaboration, knowledge sharing, organization culture, and motivation.

We see that our analysis contributes to prior works on Scrum by considering the causes and possible solutions for the mismatch in the expectations of POs, proxies between the customers and development work. Our analysis studied why the expectations of POs did not meet the development work outcome, a gap in the existing studies. The prior works (see Table 1) have studied Scrum related problems, in general, and found that insufficient collaboration [6], [10], challenges adjusting priorities [6], [10], [8], developers working on the wrong tasks [8], and lacking understanding of requirements [10] are common challenges. Respectively, solutions to these problems have been presented, which include local customer representatives [6], [7] and PO resources [7], [8]. Our case organization suffered with similar problems and they also resulted in similar solutions. Therefore, our findings consolidate the prior works and make a contribution by showing how these problems and solutions are related to failed PO expectations.

We hypothesize that in a large distributed software organization, implementing Scrum is a longitudinal process which could be very time-consuming and difficult to implement fully. Our case organization had gone through an agile transformation about a year before our study was conducted. The Scrum roles were assigned to the employees and the Scrum rituals were conducted. However, despite the effort and time used to implement Scrum, our case analysis revealed that Scrum was not completely implemented. For example, the PO was not present at the daily stand-up meetings.

A PO needs to have both team-facing and business-facing expertise. Ideally, there is a single PO who works within one development team. This means that other human resources are needed when customer needs are highly variable and difficult due to, for example, complex country specific regulations. Scrum provides little help to increase the match between the highly variable expectations of multitude of customers and software development outcomes in a complex software engineering context. In addition to Scrum, effort in creating specific and clear requirements and in guiding the teams is required. POs cannot be solely responsible for the requirements elicitation. Instead, third parties, country specific experts, a lot of collaboration, and explicit prioritization decisions are needed in a domain such as in our case.

In our case, the POs were shared by the teams. This way the organization tried to manage the complexity of the country specific regulations and decrease the problems from the geographical distribution. It enabled the developers to communicate with at least one PO face-to-face during the development sprints. However, having a PO accessible was not enough. The POs did not have enough time to meet the developers during the development sprints because they had to work with highly varying customer requirements. This also caused insufficient requirements communication with the development teams. The case organization decided to increase the number of POs in order to solve these problems.

We also hypothesize that analyzing the causes of the match-rate between the PO expectations and development work outcome helps in adopting Scrum. Analyzing "*Why expectations of Product Owners do not meet with the implemented functionality?"* helped the case organization to understand what improvements are feasible for their needs. The root cause analysis approach was also well-liked in the case organization [12]. It helped them to develop corrective actions. They continued using it in their sprint retrospectives.

5.3 Limitations

The validity of the output of a root cause analysis method relates to human factors [15]. The problems and their underlying causes were detected in public retrospective meetings. Thus, our results may have been affected by group pressure and insufficient knowledge. The following measures were taken in order to mitigate these risks. First, our results are based on triangulation, the collective findings of three individual groups. Second, we asked the organization members to evaluate the "correctness" and "impact" of the detected problems, and both got good evaluations. Third, the main parts of the analysis were conducted by the company representatives who were experts of their domain. To mitigate the risk for reliability, the data was re-analyzed by the authors. All analyzers came to similar conclusions on the detected problems, their underlying causes, and the feasible targets for process improvement.

6 Conclusions and Future Work

Our results from a large-size distributed agile software organization, which claimed to follow the Scrum method, indicate that the lack of collaboration between the customer representatives and software development teams caused the mismatch of expectations between them. We found that the problems in the collaboration escalated during the software development activities. The outcome of the sales and requirements process was insufficient and caused the developers not knowing what to do and what to test. The members of the organization did not receive enough useful instructions for the customer collaboration and for the whole software organization.

Increasing the amount of collaboration between the customer representatives and software developers was perceived as a feasible target for process improvement. In the case organization, making an improvement to the collaboration required changes in the methodologies used by the customer representatives. Additionally, it required hiring more human resources. The customer representatives were far too busy to handle the stream of the customer requests and to communicate them in the large-scale distributed software organization.

In a large and distributed software organization working with a high number of country specific regulations and customers, converting the customer needs to satisfactory solutions is a complicated challenge. Scrum provides only a partial solution to this puzzle. In our case, the POs struggled with the heavy communication needs with the customers and the third parties. They had no time to participate in the daily stand-up meetings. Thus, we conclude the following research problem for future work:

“How to successfully improve the match between customer needs and Scrum development outcomes in the context of complex, high-variability requirements and distributed development?”

References

1. J. M. Verner and L. M. Abdullah, "Exploratory case study research: Outsourced project failure," Information and Software Technology, vol. 54, pp. 866-886, 2012.
2. K. Moløkken-Østvold and M. Jørgensen. A comparison of software project overruns - flexible versus sequential development models. IEEE Transactions on Software Engineering 31(9), pp. 754-766. 2005.
3. T. O. A. Lehtinen, M. V. Mäntylä, J. Vanhanen, C. Lassenius and J. Itkonen, "Perceived Causes of Software Project Failures – An Analysis of their Relationships," Information and Software Technology, vol. 56 (6), pp. 623-643, 2014.
4. L. McLeod and S. G. MacDonell, "Factors that affect Software Systems Development Project Outcomes: A Survey of Research," ACM Computing Surveys, vol. 43, pp. 24-55, 2011.
5. K. Schwaber and J. Sutherland, "Scrum guide," Scrum Alliance, 2011.
6. S. Lee and H. Yong, "Distributed agile: project management in a global environment," Empirical Software Engineering, vol. 15, pp. 204-217, 2010.
7. M. Paasivaara, V. T. Heikkila and C. Lassenius, "Experiences in scaling the product owner role in large-scale globally distributed scrum," in Seventh International Conference On Global Software Engineering (ICGSE), IEEE, 2012, pp. 174-178.
8. N. B. Moe, T. Dingsøyr and T. Dybå, "A teamwork model for understanding an agile team: A case study of a Scrum project," Information and Software Technology, vol. 52, pp. 480-491, 2010.
9. D. E. Strode, S. L. Huff, B. Hope and S. Link, "Coordination in co-located agile software development projects," J. Syst. Software, vol. 85, pp. 1222-1238, 2012.
10. E. Bjarnason, K. Wnuk and B. Regnell, "Are you biting off more than you can chew? A case study on causes and effects of overscoping in large-scale software engineering," Information and Software Technology, vol. 54, pp. 1107-1124, 2012.
11. R. K. Yin, Ed., Case Study Research: Design and Methods. United States of America: SAGE Publications, 1994.
12. T. O. A. Lehtinen, R. Virtanen, J. O. Viljanen, M. V. Mäntylä and C. Lassenius, "A tool Supporting root cause analysis for synchronous retrospectives in distributed software teams," Information and Software Technology, vol. 56 (4), pp. 408-437, 2014.
13. M. Q. Patton, Qualitative Research. Sage, 2002.
14. T. O. A. Lehtinen, M. V. Mäntylä and J. Vanhanen, "Development and evaluation of a lightweight root cause analysis method (ARCA method) – field studies at four software companies," Information and Software Technology, vol. 53(10), pp. 1045-1061, 2011.
15. P. Runeson and M. Höst. Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering 14(2), pp. 131-164. 2008.