12-10-2016

Agile requirements work in a digital transformation project: Managing diverse and dispersed user needs

Kathrine Vestues
kathrine.vestues@idi.ntnu.no

Finn Olav Bjornson
bjornson@idi.ntnu.no

Agile requirements work in a digital transformation project: Managing diverse and dispersed user needs

Kathrine Vestues
Norwegian Univ. of Science and Technology
Kathrine.vestues@idi.ntnu.no

Finn Olav Bjørnson
Norwegian Univ. of Science and Technology
bjornson@idi.ntnu.no

ABSTRACT

Successful requirements engineering is vital to the success of software projects. Agile software development seeks to limit the risk of misunderstanding requirements by emphasizing evolutionary delivery and more end-user involvement. But what happens when features are not accepted because the customers cannot agree among themselves? In this paper we report on an ongoing study where a software development company is creating a software system from scratch for a complex, diverse, and dispersed customer organization. We describe our ongoing study in which we follow a feature of the software system from idea to implementation. We attempt to explain our observations through three theoretical lenses: User participation and involvement, power relations in complex organizations, and balancing of local and global needs in system development.

Keywords

Agile development, Agile project management, Agile requirements engineering, User participation

INTRODUCTION

Since the formulation of the agile manifesto in 2001, agile methods have transformed software development practice (Dingsøyr et al. 2012) by emphasizing tolerance for changing requirements, evolutionary delivery and more end-user involvement. According to (Dybå et al. 2014), agile software development represents a new approach for managing software projects, that puts less emphasis on up-front plans and strict control and relies more on informal collaboration, coordination, and learning. In their chapter on agile project management, they provide software project managers with a set of four principles for handling complexity and uncertainty inherent in agile software projects. In this paper we focus on their first principle, minimum critical specification, which deals with how requirements should be elicited and specified. Understanding “the problem” the system is intended to address, they claim, is one of the keys to project success.

In the study described in this paper, we follow the development of a new software tool that is to be used by inspectors in a public, Norwegian directorate. The directorate is responsible for ensuring sustainable use of Norwegian natural assets, and as part of their operation, they perform a large number of onsite inspections to ensure that laws and regualtions are being met. A surveys performed within the directorate revealed that to fulfill this task in a consistent and efficient manner, they needed a new software system. Prior to the development of this system, inspections had been performed using pen, paper and simple text editing tools.

Despite intense focus on user involvement in all stages of development of the new system, the project manager still struggles to get users to approve the system. Key features are deployed, but remain unused. This failure to meet user expectations and deliver usable functionality is both frustrating and costly for the project management.

In this paper we describe our ongoing investigation of the challenges of requirements engineering in this complex customer organizations. To better understand the process of feature development in the project, we follow the development of a single feature from idea to the deployable functionality. This gives us a fruitful perspective when looking at requirements engineering in complex organizations, and seeing how this process affects the organization.

The paper is structured as follows: First, we provide a detailed description of the customer organization, where the main complexity of this case seems to reside. We move on to briefly describe our research design for investigating the case, before providing a theoretical background for our current understanding of the challenges faced by the project. We then move on to discuss the case in light of the different theories in order to find possible explanations for our observations. We conclude by outlining our planned research for following the case further.

CASE

In the case study, we look at the development of a new control and supervision tool that is under development for a public, Norwegian directorate. The directorate is responsible for the management of natural assets, and among its tasks is the supervision and control of how private corporations use these assets. A survey revealed that they needed a new and improved software system to support these supervisory and control operations. The new system is ment to ensure efficient work processes, quality of data and equal treatment of supervised bodies. It will support the work of two distinct user groups with very different requirements and knowledge: (1) Inspectors working in the field and (2) office personnel analyzing inspection data and compiling reports. There existed no system on which the new system could be based.

The directorate has approximately 500 employees, of which 220 work at the head office that is located in the west of Norway. The rest of the employees are distributed across 20 regional offices throughout the country – from Kjøllefjord in the north, to Kristiansund in the south. Each regional office has responsibility for the natural assets in whitin their allotted, geographic region. For instance case processing and operational control takes place at the regional offices, while administration and support is handled by the head office.

The development of the new software system is organized as a project on the customer side, with a dedicated project manager, steering committee, and a group of user representatives known as “product owners”. The development of the system was awarded to an external supplier, with a development team distributed across three Norwegian cities: Trondheim, Oslo and Bergen. The project uses agile software development methodology, working in one month iterations. Each iteration is initiated with a planning session, and ended with a customer demo and retrospective. The product backlog is continuously groomed and reprioritized.

One especially important group of features in the system are checklists for planning and execution of controls and audits. The checklists are seen as vital for ensuring predictability and quality during inspections. As one of the caseworkers at the head office expressed it: “You cannot save a bad inspection with good case management". The functionality was part of the original tender, and was designed and developed in close collaboration with end users. At present, the project has been running for a year, and is estimated to continue for ten more months. The checklist functionality has been completed, but not used. It is currently uncertain if it ever will be taken into use.

RESEARCH DESIGN

The objective of this study is to see how user participation, organizational politics and tension between local and national needs affect the requirements process. To do so we have chosen to use a longitudinal case study. The rich descriptions given by a qualitative study can help us understand, explain and provide guidelines for requirements engineering in projects where user groups are dispersed and diverse.

The case was chosen because of the complexity of its organization: Both the development team and the customer is distributed across multiple locations. In addition, user groups are diverse, and there exists no prior solution on which the new system can be based. It therefore requires extensive negotiation and communication both within the customer organization and between developers and customer representatives to agree on requirements. The case is currently into its second year, and is estimated to continue until June of 2017. We will follow the case until the system has been completed and deployed.

We will explore the case from the view of the development team, observing and interviewing team members. In addition, we will have access to interviews with key members of the customer organization. These interviews will be performed by fellow researchers engaged in a related research project. To narrow down the complexity of the case, we will follow a single unit of functionality from early needs analysis, through discussions, prototyping, development, deployment and use. By looking at the evolution of one requirement, we can deepen our understanding of the requirements process, and be able to better understand the pitfalls and challenges of requirements engineering in complex socio-technical environments.

In our future data analysis we plan to use triangulation to broaden our understanding, and test the validity of the data. More data is needed before this triangulation can be perfomed. We will analyse data from observations, interviews and written artefacts, such as backlogs and memos. As formal interviews are performed, they will be transcribed and analyzed. We have not yet concluded on method of analysis.

Table 9 - Collected data

Artefacts	Description
Project wiki	Contains descriptions of initial requirements. Edited and used by client representatives, as they were elected for development, these requirements were transferred turned into Epics and User stories and handed over to development team
Product backlog	Collection of Epics, User stories and taks that have been elected for development. Registered in project and issue tracking tool
Observations	Observations done of the team at their work stations and in meetings. Only team mebers in Trondheim will be observed in person. Other team members (Oslo and Bergen) will only observed online (video meetings)
Informal interviews with development team	Conversations taking place after meeting, during luch and at coffe machine. The content of these conversations are logged in note book after conversations have taken place. Informal interviews will only be done with team members in Trondheim
Formal interviews with development team	The following interviews will be performed: - Scrum master / project manager (Trondheim) - 2 developers (Trondheim) - Test manager (Trondheim) - Interface designer (Oslo)
Formal interviews with customer	So far 6 interviews have taken place by other researchers. 13 more interviews have been planned. We will have access to use these in our analysis
Official documents	Original tender found on the web. Organizational charts and descriptions of goals and missions of the directorate

At present we have done observations of development team, had informal interviews with the development team, and looked at publicly available documents describing early project requirements. Observations include project meetings internal to the development team, and meetings where the customer has been present over video-link, as well as team members working at their work stations.

REQUIREMENTS ENGINEERING THEORY

Understanding the intended purpose of system, and the needs of its users, is imperative for successful software development. For the development of complex systems with dynamic, non-deterministic and non-linear characteristics, where accurate estimates, stable plans, and predictions are difficult to establish in early stages, many software project will employ agile methodologies. Agile software development methods are a collection of methods that promote teamwork, collaboration, and process adaptability. Some researcher see requirements engineering in agile software development as different from traditional requirements engineering (Bjarnason et al. 2011), and that requirements must be seen as evolving elements in an ecology (Jarke et al. 2011). In agile projects, requirements engineering is an integral part of the development process throughout the project lifecycle. An agile project will often have an initial design phase, but these early requirements are coarsely defined, and will be gradually elaborated throughout the project. Ideally, this will prevent the project from specifying features that will never be developed, and ensure that at any given time, the functionality that gives the most customer value will be prioritized and developed. Research has shown that many challenges faced by traditional requirements engineering will be solved with an agile approach (Inayat et al. 2015), especially relating to project communication and stakeholder involvement. Customer collaboration will continue throughout the project lifecycle, and requirements are added, altered and reprioritized at regular intervals. The use of agile requirements engineering will however introduce some new challenges. Among these are expectations for customer involvement, and customer inability and disagreement, which could clearly be seen in our case study. When choosing vendors, the directorate had agreed that the project would require intense involvement on their part.

Based on a preliminary analysis, it seems useful to look at data through the following theoretical lenses in an attempt to explain observations: (1) User participation and involvement, (2) power relations in complex customer organizations, and (3) the balancing of local and global needs in system development.

User participation

Most software development projects have some degree of user participation. Participation is used to avoid user resistance, gain user commitment and ensure that user requirements are met. These ideas of empowerment and democratization are central to agile software development methodologies such as Scrum and XP, where customer involvement is central to requirements elicitation and prioritization. Users are involved through a range of practices, such as constant planning, extreme prioritization, prototyping and test-driven development (Cao and Ramesh 2008). Most studies show a positive correlation between user participation and successful software design (Abelein and Paech 2015), but there still exist many example of projects that have failed in spite of user participation (Cavaye 1995). One has to pay careful attention to how and when users are involved. Due to time constraints, users often find it difficult to engage themselves until they are forced to use the new system. Management often fails to dedicate time for users to test and give feedback on prototypes and test installations. In our case study, developers complain that users are unable to answer questions and attend meetings due to operational tasks. This in spite of promises to devote adequate time to the project. Apart from the project manager, who is dedicated full time to the project, customer resources are obliged to maintain their usual responsibilities. The Directorate has to fulfill its responsibility to the public regardless of the new system development. Techniques such as prototyping and sketching might be useful tools for involving users, but require high users engaged. If users are not sufficiently engaged, prototyping will only give an illusion of involvement. User´s reaction will await final release of the software, leaving developers baffled and frustrated when they find users unwilling to accept the software. In our case study, failure to meet customer requirements was a recurring headache: Even after rounds of prototypes and demonstrations, customers were displeased.

Requirements engineering work will be especially challenging when the new system does not replace or build on an existing system: "*In a new development project, developers have a poor understanding of product requirements, and are often unable to make wise product decisions because of requirements ambiguity. Ideally, they would benefit significantly from user inputs in these projects. However, given the novelty of the technology involved and their uncertainty about "what the end product should look like", users too are likely to find it difficult to express their needs and suggest design guidelines. In these projects, project managers and leaders can play a strong role in clarifying the expectations of the customers/users as well as the software development firm*" (Subramanyam et al. 2010). This too could be seen in the case study. There was no existing software on which to base the new solution. In addition, the complex organization with its diverse and dispersed needs, made it difficult for project managers and leaders to give clear advice and make conclusive decisions.

Power relations in complex customer organizations

In their paper on understanding the political ecology of requirements engineering, Bergman et al. (Bergman et al. 2002b) addresses the political nature of requirements, and argues that requirements engineering practice and theory must become more engaged with this issue. They describe requirements as emerging from a set of solution spaces depending on the different problem spaces of the involved principals in the organization. Navigating and synthesizing the heterogeneous technical, social, economic and institutional factors are a key to successful requirements engineering. Further, they argue that the traditional functional ecology of requirements engineering is dependent on goal congruence among the stakeholders and if this is not the case, a political ecology of requirements engineering is more suited to establish valid requirements.

Following the track on politics in requirements engineering, a new research agenda for power and politics in requirements engineering was proposed (Milne and Maiden 2011). It is argue that given the increased complexity, uncertainty and organizational embeddedness faced by requirements engineering in practice, power and politics have become increasingly relevant factors that have not been given adequate consideration.

Balancing local and global needs

Much has been written about the need to adapt software systems to the local context, as the local context always will be unique with its local practices and existing infrastructure resources. A failure to recognize this need will prevent employees from performing their work tasks in a suitable and efficient manner. It has, however, been demonstrated that in global information infrastructure systems, a balance must be found between the local and global needs (Rolland and Monteiro 2002). The larger organization wants standardization across sites, while local inspectors require autonomy and flexibility in the face of local variations. Although the organization only operates within Norwegian boarders, the large geographic and cultural difference between different regional offices and between regional offices and the head office makes analogy with global versus local needs relevant and suitable.

RESULTS AND DISCUSSION

Initial observations and informal interviews with members of the development team show that important functionality fails to meet customer expectations. Checklists, which are seen as central features in the new control and supervisory tool, have still not been taken into use, months after deployment. Rework and delays result in increasing costs and widespread frustration. Although recognizing the need for close collaboration with developers, the directorate fails to follow up on their good intentions. As the project proceeds, the customer organization is absorbed by daily chores, failing to give the project sufficient attention. Developers are often left to wait for feedback on important clarifications, slowing down progress and causing widespread frustration. The organization is prone to internal disagreements, and are unable to conclude on controversial issues. According to developers, decisions they think final are often refuted and reemerged during customer demonstrations and retrospectives, leaving the development team ill at ease. To complicate the situation further is the fact that there existed large local variations between regions and types of inspections. Satisfying the needs of one user group seems to impede the work of others. Additional requirements come from caseworkers located at the head office, who have different work practices and information needs compared to inspectors.

In our search for explanations to these project shortcomings, we look in different theoretical directions. One of these is the participation of users in system design. Following an agile methodology, users are involved in every stage of the project, and they make extensive use of collaborative design techniques such as prototyping and sketching. Still, users are dissatisfied with system functionality once deployed. This failure to meet expectations causes frustration, delays and cost overruns. A key reason for this behavior seems to be time pressure on the part of the customer representatives. They are not alleviated of their regular responsibilities, and operational tasks have to be prioritized, leaving the development team to wait. Developers spend a substantial amount of time waiting for customer feedback and clarifications, showing that if user involvement is to be efficient, users must be at least partly dedicated to the project. Otherwise, feedback will come as functionality is deployed, undermining the agile mindset.

Another part of the explanation for the project´s inability to develop adequate features may be the complexity of the customer organization. It seems that both the customer side project manager and the supplier has underestimated the role of intra-organizational politics. Power structures within a complex organization can hinder efficient system development (Bergman et al. 2002b; Milne and Maiden 2011). Drawing on the theory of Bergman et al. the constant rounds of renegotiation of requirements can be seen as a sign of political ecology problems, since they have a tendency to manifest as repeated rounds of systems change orders that are sometimes described as requirements creep (Bergman et al. 2002a). Apparent agreement is reached by defining the requirements too ambiguous, and problems then arise by placing undue influence on system designers who are working at a concrete level and have to make decisions on the ambiguous features. Bergman et al. describes the consequence of this eventuality as important stakeholders losing control, and that this may lead to them emerging later a opponents to the current system because their needs are not being met. Although the theory on political ecology was developed with large-scale requirements analysis in mind, we would like to argue that the theory is also valid in a smaller setting. It would also be interesting to see how the model can be applied in an agile setting. Data from the case study suggest that internal politics and power struggles have affected the project. Further analysis is required to find the degree to which this affected the design of checklist functionality.

It is also possible the project´s apparent inability to meet customer expectations is due to the imbalance between local and global needs (Rolland and Monteiro 2002). There seems to be a conflict between the desire to standardize work processes, and a need for local adaptations in the field. Inspectors are dispersed across large geographic areas, and are responsible for a multitude of facilities with differing information requirements. Case workers at the main office, on the other hand, require elaborate and consistent data, enabling them to make sound and predictable decisions. Facilities that are supervised and fail to meet laws and regulations can be subject to prosecution or injunction. The directorate therefore strives to be unified and fair in the way supervisions are performed.

LIMITATIONS OF THE WORK

The case study describes software development in a public, Norwegian organization. This means that Norwegian laws on public procurement policies strongly effect the acquisition and development process. In addition, the research is still at an early stage, and a more data collection and analysis is needed for final conclusions to be made.

CONCLUSIONS AND FUTURE WORK

Returning to the principle of "minimum critical specification" in agile project management, our case demonstrates how requirements can be misaligned, interpreted differently, and ultimately not be accepted, despite the best intentions of both customer and developers, and following agile practices to the letter. Our preliminary observations and analysis points to the complex structure of the customer as the source of the challenge, and several theories explains different aspects of this. What is clear, however, is that there is little support in the agile literature and methods to deal with these kinds of challenges.

By following the case to its conclusion we aim to contribute to the theoretical framework surrounding requirements engineering with a complex customer structure in an agile context. We also hope this research will produce practical advice for agile project managers dealing with complex customer structures.

REFERENCES

Abelein, U., and Paech, B. 2015. "Understanding the Influence of User Participation and Involvement on System Success–a Systematic Mapping Study," *Empirical Software Engineering* (20:1), pp. 28-81.

Bergman, M., King, J. L., and Lyytinen, K. 2002a. "Large-Scale Requirements Analysis as Heterogeneous Engineering," *Social Thinking-Software Practice*, pp. 357-386.

Bergman, M., King, J. L., and Lyytinen, K. 2002b. "Large-Scale Requirements Analysis Revisited: The Need for Understanding the Political Ecology of Requirements Engineering," *Requirements Engineering* (7:3), pp. 152-171.

Bjarnason, E., Wnuk, K., and Regnell, B. 2011. "A Case Study on Benefits and Side-Effects of Agile Practices in Large-Scale Requirements Engineering," *Proceedings of the 1st Workshop on Agile Requirements Engineering*: ACM, p. 3.

Cao, L., and Ramesh, B. 2008. "Agile Requirements Engineering Practices: An Empirical Study," *IEEE software* (25:1), pp. 60-67.

Cavaye, A. L. M. 1995. "User Participation in System Development Revisited," *Information & Management* (28:5), pp. 311-323.

Dingsøyr, T., Nerur, S., Balijepally, V., and Moe, N. B. 2012. "A Decade of Agile Methodologies: Towards Explaining Agile Software Development," *Journal of Systems and Software* (85:6), pp. 1213-1221.

Dybå, T., Dingsøyr, T., and Moe, N. B. 2014. "Agile Project Management," in *Software Project Management in a Changing World*. Springer, pp. 277-300.

Inayat, I., Salim, S. S., Marczak, S., Daneva, M., and Shamshirband, S. 2015. "A Systematic Literature Review on Agile Requirements Engineering Practices and Challenges," *Computers in human behavior* (51), pp. 915-929.

Jarke, M., Loucopoulos, P., Lyytinen, K., Mylopoulos, J., and Robinson, W. 2011. "The Brave New World of Design Requirements," *Information Systems* (36:7), pp. 992-1008.

Milne, A., and Maiden, N. 2011. "Power and Politics in Requirements Engineering: A Proposed Research Agenda," *2011 IEEE 19th International Requirements Engineering Conference*: IEEE, pp. 187-196.

Rolland, K. H., and Monteiro, E. 2002. "Balancing the Local and the Global in Infrastructural Information Systems," *The information society* (18:2), pp. 87-100.

Subramanyam, R., Weisstein, F. L., and Krishnan, M. S. 2010. "User Participation in Software Development Projects," *Communications of the ACM* (53:3), pp. 137-141.