Towards super user-centred continuous delivery: a case study

Joakim Klemets[1,2] and Tore Christian Bjørsvik Storholmen[1]

[1] SINTEF Digital, Dept. of Health Research, Trondheim/Oslo, Norway
tore.christian.storholmen@sintef.no
[2] Norwegian University of Science Technology (NTNU), Dept. of Computer Science, Trondheim, Norway
joakim.klemets@ntnu.no

Abstract. To develop well designed socio-technical systems for a particular context user involvement is essential. Emerging software development approaches that enable continuous delivery of software functionalities provide new opportunities as to how users can be involved in shaping the design. We present a case study of a software project at a Norwegian engineering and construction company in their effort to move towards a user-centred development approach that leverages on continuous delivery principles. We investigate how users-representatives in form of super users have been involved in the process and their role in forming and implementing the system in practice. We conclude that utilising super users have been instrumental to design a system that meet users' needs. Providing opportunities to test the system in a production environment gave rise to new ideas on how to further refine the design. Involving super users also facilitated system adoption among workers.

Keywords: User-centred design, Continuous software engineering, Agile development, Super users, Qualitative research

1 Introduction

To remain competitive in a challenging market many companies seek to make work processes and procedures more effective through developing and introducing new digital tools. However, to utilise the new technology it needs to be adopted and be taken into use. From a socio-technical standpoint, developing and introducing new digital solutions is as much about forming new work practices and routines as designing the tool itself meant to support work [1]. Hence, in enhancing the possibility of the tool to be accepted into practice one should ensure that it aligns with organisational goals and values.

Scandinavia has a rich history of politically empowering workers to 'have a say' in shaping their own working environment [2]. The Scandinavian approach to participatory design has therefore emphasized principles that are today central in user-centred design approaches [3]. User involvement enhance user acceptance and ease system adoption through generating more usable solutions [4, 5]. It has also been singled out

as an important factor for project success [6-8]. Although user involvement has received limited attention in software engineering [9], there is a growing interest in looking to combine user-centred approaches with existing system development methodologies [6]. In Particularly agile approaches been singled out as a good fit with user-centred design.

While agile approaches are commonplace today, more and more companies are also moving towards continuous delivery, where new functionality is delivered to actual users frequently. To be able to release new functionality into production more rapidly, the development and operations (DevOps) teams collaborate tightly to automate parts of the delivery process. Whereas there are little empirical studies on DevOps implementations [10], there are even fewer looking at combining a continuous delivery with user-centred principles [11].

We report on a case study carried out at a large Norwegian engineering and construction services company that through a digitalisation strategy seek to transform its IT-development towards a DevOps and user-centred approach. The research objective is to investigate the transformation from a traditional and linear software engineering model towards an iterative and user-centric approach. Particularly, we zoom in on the end-user involvement and ask the following research questions: how are end-users involved in different phases of the process and what are the challenges? How do the end-users representant, or the super-user, role facilitate system improvement and implementation?

2 Background

2.1 User-centred design in agile and continuous software development

A number of scholars have discussed different perspectives on how to integrate agile and user-centred practices as a mean to produce more usable software [12-15]. From a process perspective [12], many scholars advocate having up-front design activities that seek to explore the problem space and possible design solutions before conducting any actual software development. To better adhere to the lean principles of agile practices, little design upfront is advocated [15], where whereas this phase often is more extensive in traditional user-centred design.

One approach to implement design up-front is through carrying out usability studies one iteration ahead of the development [12]. Kuusinen [16] argues that the approach, which usually separates developers and usability-experts into separate teams, prolongs the time to feedback, as well as risk wasting development resources on unusable functionality. Instead, she proposes a framework where cross-functional teams collaborate. First, through a short up-front exploration phase and later within the same iteration. And that feedback is collected continuously from end-users (Fig. 1).

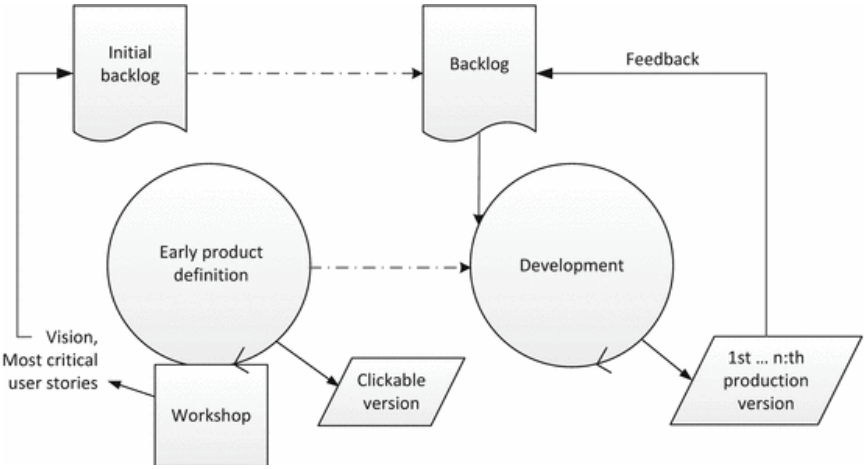

Fig. 1. The BoB framework [16].

Continuous involvement of stakeholders is another central user-centred agile software development principle that has received less attention in form of empirical studies [12]. Particularly, finding good ways to involve the end-user in the process is important [17]. However, user involvement in the development is complex, and whom to involve, when, and how is not straightforward [9, 18]. Whereas user involvement in the requirements phase has shown to be effective [8, 19, 20], studies also indicate positive effects of user involvement throughout the process [18, 21]. Doing so can, for example, enable users to feel greater ownership of the system [18] and, hence, enhance adoption.

Haake et al. [19] found that it is central that a large share of affected users are involved in the implementation phase, as failing to do so could have negative effect on project success. Bano et al. [18] reports that failing to identify motivated and positive user representatives also has negative effects. With regard to continuous software engineering (CSE), where new software functionality is realised frequently to the end-user [22, 23], data-driven approaches have been developed that automatically monitor system use and quantified user feedback [24]. However, Johanssen et al. [25] who inquired 17 companies adopting CSE, found that the companies also collected explicit (i.e. written or verbal) user feedback in addition to passively collecting user metrics. Yet, the feedback context was mostly neglected, which made it difficult to fully utilise the collected feedback [26].

2.2 Super users and design-in-use

Whether it is super user, power user, or champion user, the role goes by many names. Hansen and Anders [27] define super users as 'regular employees with in-depth knowledge of one or more of the organization's computer applications without being programmers'. Hence, skilled workers in their own field that also play a significant role as user representatives in system development, implementation, training, and support.

To be successful in this role, the super user should be positive towards adopting new technology, good communicators and be patient teachers, as reported by McNeive [28].

The use of super users has also been studied within end-user development, where users are empowered to make own configurations to a system or through the close collaboration with developers and IT-professionals [27, 29, 30]. End-user development or design-in-use has been suggested as an alternative to traditional system implementation, which takes place once the design is finished. The design-in-use approach acknowledges the idea that implementation is context-dependant and tightly coupled with work practices and processes. Hence, once a minimal viable version of the system is ready to be deployed, end-users continue the design of both the system and associated practices [29, 30].

3 Methods

3.1 Research setting

The Mill Co.[1] is a large Norwegian engineering and construction company specialised in offshore constructs that recently has initiated significant digitalisation efforts. The company operates mainly on two different sites within Norway. Whereas one of the digitalisation aims is to make existing work processes more effective through developing new information technology, it also aims to empower employees to take more control and responsibility of their work. One particular project that seek to do so is the CoConstruct project. The aim of CoConstruct is to build a mobile platform through which construction workers (welders, electricians, plumbers, etc.) and their foremen can use smartphones to access information such as work orders, plans, and documentation. It is also meant to provide a platform for team communication and allow real-time work progress reporting.

Mill Co. has an in-house IT-department that lead development projects and are responsible for operating and maintaining existing systems. However, the majority of the development workforce is hired on a project basis from another software engineering company with whom Mill Co. has a long-lasting collaboration. As part of the CoConstruct project, the development team also made the transition from using a traditional waterfall model to a more agile inspired approach, incorporating DevOps practices to release new software version to production roughly every 2 weeks. This meant that during the span of the project, the team had to acquire new skills, adopt new practices, and adopt new development tools (e.g. Azure DevOps). Further, the team also decided to incorporate user-centred design principles into their way of working by involving usability and user experience (UX) experts.

The entire CoConstruct project team consisted of a project manager, two UX researchers, a UX designer, a system architect, five front-end developers, and four back-end developers. Further, platform architects specialised in existing production systems were involved to ensure system integration. Two experts in mobile platforms (Xamarin) and cloud services were also included in the development team.

[1] To preserve the anonymity of the people involved we use fictional names

A pool of super users was established from the beginning of the project, including 15 users from each construction site (a total of 30 super users). The super users consisted of both foremen and construction workers that represented different disciplines. Particularly, super-users that were curious about new technology and positive about the project were selected. The development team tried to ensure variety by including super users that were not so experienced with digital technology, such as smartphones, and recruiting super users of different age groups.

3.2 Research design

A case study strategy [31] was adopted to investigate the development team's processes and practices, as well as experiences of the ones involved in the CoConstruct project. Particularly, we focus on how the end-user representants have been involved in the project and its implications. The case study lasted for about a year, between 2019 and 2020. The CoConstruct project was still active after the data collection ended.

Within this time period we collected qualitative data through semi-structured interviews and a focus group with a variety of people involved in the project (Table 1). An interview guide was designed that covered the different project phases: context study, requirement analysis, development, testing and implementation. A separate interview guide designed for user representatives (super users). Seven of the interviews were recorded and seven were transcribed. For the remaining we made notes during the interviews. We also got access to documents and presentations providing information about the CoConstruct project and associated work processes. All data material was thematically analysed by both authors. The study has been approved by the Norwegian Centre for Research Data (NSD) and informed consents was collected from study participants.

Table 1. Data collection occasions.

Participant(s)	Data collection method	Duration
Project manager, 1 developer, 1 UX researcher, 1 test engineer	Focus group interview	1,5 h
Developer	Interview	1 h
Test engineer	Interview	1 h
UX researcher	Interview	1 h
Project manager	3 online interviews	2,5 h in total
4 super users from 3 different construction teams.	4 separate online interviews with each participant	30 – 45 min for each interview

4 Results

4.1 User involvement in the development process

A high-level overview of the development process is illustrated in Fig 2, which also highlights where and how in the process users were involved. An up-front context study

phase initiated the process (1), which was followed by requirement analysis (2), development (3), test/deploy (4), release (5), and implementation (6).

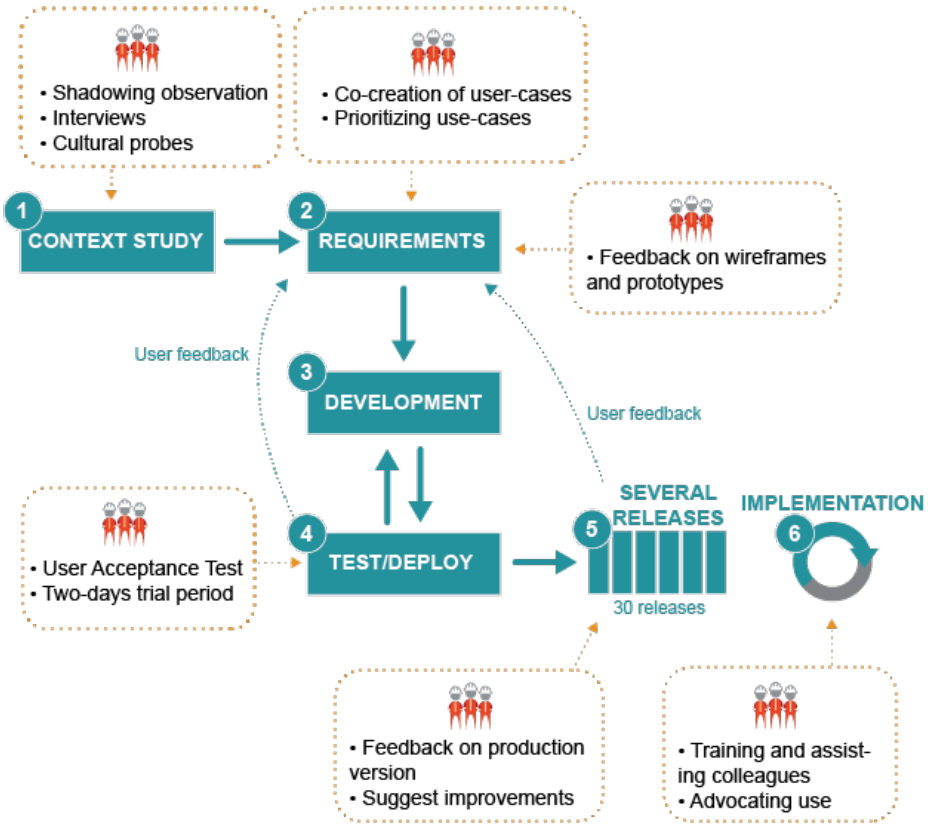

Fig. 2. User involvement in the development process

Context study. The first project phase was undertaken by a separate UX team, an assembly of 2 UX researchers, a UX designer, the project manager and a system architect. The aim was to gain an understanding of the context of use and user needs by investigating current tools, processes and practices at both sites. The UX team conducted data collection that lasted for 2-3 days, for each site, and included activities such as workshops, shadowing observations, interviews, as well as utilising WhatsApp[2] and disposable cameras as cultural probes [32]. The use of WhatsApp groups made it possible to collect great amounts of photos and live data, where workers described their work situations and the work environment. This activity continued after the team had left the sites, and was, according to the team members, particularly useful to collect data from the site that was located geographically distant from the development team.

[2] A mobile messaging application

Fig. 3. Workshop participants discussing the experience map.

Based on the collected data, the UX team developed a timeline experience-map presenting activities and interactions between workers, as well as challenges and opportunities for new technology. The workers were invited for a co-creation session (same people that was shadowed), where they were encouraged to further refine the experience-map (Fig. 3).

Requirements. The UX team that figured in the context study phase, collaborated closely with the super users to write an prioritise initial use-cases in the subsequent phase. In order to make more accurate prioritisations, the UX team tried to understand how valuable a use-case is from both a user and business perspective. The team also aspired to estimate the technical complexity of implementing a certain functionality. A team member explained that 'a use-case could be very relevant, but if it would take four months to deliver the functionality it was assigned a lower priority.'

This process led to the selection of 14 use-cases. The selected use-cases were presented to super users at both sites and to a steering-committee. The super users were also asked to individually prioritize the use-cases in an Excel-spreadsheet, which led to 6 more use cases being added.

Based on these use-cases, the UX team initially developed wireframes[3] for the first planned features. Later the UX team created clickable prototypes as well. To get feedback on these early designs, the prototypes were presented to super users.

Development. To hand over the initial system requirements to the development team, a 7-day kick-off workshop was held where the UX team presented the selected use cases. Initially a total of 60 features were examined to estimate a total development time for the project.

[3] Non-interactive prototypes illustrating the applications user interface

The development was carried out using a Scrum-influenced approach with a 2-week sprint interval. Two separate teams worked in parallel iterations but delivered functionality as one unit. If one team ran out of tasks within a sprint, it assisted the other team if suitable resources were available. The teams held daily 'scrum-talks' and review meetings after each sprint. Both UX experts and the developers involved in the initial UX team was also part of the development team, which allowed the team to more fully take advantage of the experience gained from the previous phase in the development.

Test, deploy, and release. At the end of each sprint, the development team arranged a meeting to present the newly developed functionality to the super-users, and to run a user acceptance test (UAT). The super users participated either on site or through Skype. The number of participants varied; some participated every time while others participated less frequent. During the UAT the super users installed a test version of the CoConstruct application on their smartphones and were handed individualised test scripts to carry out during the session.

The aim was to discover potential bugs and to prioritize these in different categories according to severity. Identified defects and issues were entered into the development backlog. After this session, the super users were provided the opportunity to use the application for two days and report feedback. If still no major issues were found after the 2-day period, the application was released to production, making it available for everyone. After the release, feedback was generally collected by superusers and communicated to the development team at the following UAT meeting.

Implementation (training, support, and dissemination). To support the implementation and adoption of the CoConstruct application among construction workers and foremen, the development team are establishing web-based training courses. A separate department of education at Mill Co. was also prepared to train smaller groups in how to use the application. The project manager told that they are monitoring various use statistics, which is used to more easily identify departments that possibly are in need of additional support and training.

The CoConstruct project has also developed user manuals presenting core functionality of the application to educate users. The project group have been sending out newsletters on email to keep people informed and presented teasers on information screens in the canteens, reception, etc.

4.2 Reflections on the development processes

Involving super users. The CoConstruct project ensured user involvement by establishing a pool of super users. Some were recruited from the beginning of the project; others were onboarded later. The main idea was to involve the same users throughout the project for discussing and validating needs, desires, requirements, and solutions. They were also responsible for collecting and forwarding feedback from colleagues.

Dual roles. The super users interviewed were generally pleased with their role in the process, and some took great pride in participating. The super users admitted that the additional role sometimes caused stress due to the extra workload. The project manager suggested the possibility of rotating the super user role with others. He told: 'I believe some of them still want to continue as super users, but some are maybe getting tired. They have spent a lot of time on this.'

However, having a management that support and understand that the super user role will reduce time devoted to 'regular work' was considered important. As a super user explained: 'it has worked out nice to have the super user role in combination with regular work. The management understands that I spend time on this. This is no problem in my department. And I'm also passionate about making this work'. Further, having both a foreman and a construction worker from each construction team involved as super users was described as valuable, as it enabled them to support each other.

Super users' role in dissemination and training. According to the project manager, superusers main responsibility was to be involved in the development and to provide feedback back to the development team. In addition, the superusers were also expected to serve as 'ambassadors' for the CoConstruct application. As well as to provide some support to their work colleagues. But providing training was not part of their responsibility, the project manager explained.

However, super users from two of the construction teams involved in the study (three in total) reported that they were very engaged not only in advocating use, but also in training their colleagues in how to use the system. The superusers in these teams were very active in taking the application into use at their workplace, which was done on their own initiative. One of these teams' strategy was to hold training sessions with groups of 3-4 people at a time. They preferred smaller groups that allowed for closer interaction as the users try out the application and ask questions during the session.

For people that were not used to smartphones, or otherwise resistant to new technology, they held individual training sessions. The interviewed superuser explains: 'there has been resistance among some individuals that do not want to use digital tools, that find it difficult. We have taken these aside and gone through and thought them one and one. Walking through and repeating the tasks that the person should carry out. And at the end things have went fine.' The superuser continues, 'One-to-one walkthrough is the best way to achieve this. If you take them on one-to-one, they become a bit more comfortable with the situation, and you can explain it to them slowly, and then they understand it better. A lot of it is about lack of knowledge and insecurity. If you do this one-to-one much of the insecurity disappears.'

Feedback from test and production versions. Within the span of 1,5 years, about 30 new production versions of the application were delivered to the users. Whenever a new version of the CoConstruct application had been finalised, it was made first available as a test version only for the super users. The super users said that the initial test version they got was mainly checked for major flaws regarding the applications user interface and functionalities. And most of the times, the test version was accepted by the super user without any complaints.

However, releasing the version into production meant that the application also could be tested by a larger number of users, and in a real work context. Hence, how well the new features align with work practices, and the influence of a new version on work efficiency could be scrutinised more thoroughly. The super users told that is was during this phase (after production release) that most major issues with the application emerged. And that issues became much more visible once a lot of people started complaining or simply stopped using the application.

One super user said that the feedback typically came after using the application for a while and when people had been facing the same challenges over time. He explained: ’in the beginning, there are often few people who give feedback, because they haven't been annoyed enough about it. But as time went on, people stopped using the mobile to report progress. Because it took too long compared to just ticking off the job package, like you did previously. It's mainly when many people use it that you get feedback. When you face the same problem as your colleagues’. Similarly, both the super users and the development team anticipated that the application would improve as the user base grows, which meant more feedback on how to improve it.

Another super user also suggested to speed up the feedback process when facing critical usability issues. The super users were supposed to report feedback in the following UAT meeting, which could be sometimes a month away. Considering development time, it could take a significant amount of time before the workers had an improved version available. When asked about how the process could be improved, he told: ‘maybe not having to wait for another UAT meeting to address critical issues we had. It took a few extra weeks to fix the problem as well, and I noticed that the motivation to use the app was very low at that time. We should have contacted the team directly and said that this must be fixed’. The super user regretted that he had not used more effort to get the development team to resolve the issue more quickly, saying: ‘I should have pushed them regardless’. A similar story is also shared by another superuser, where a software update led to that operators went back to ‘doing it the old way’.

The project manager also told that they could have spent more time with the users to identify their daily challenges. But he admitted that it is challenging to know how to allocate available resources. He said: ’we should have been more visible at the construction area to follow up on daily challenges, but this is challenging. Should you spend time developing or should you focus on follow-ups and support?’

Screening of feedback. Currently the CoConstruct application has approximately 500 unique (not necessarily active) users. This generates an amount of feedback that the developers find hard to process. The feedbacks include either bug fixes, suggestions for new functionality or suggestions on how to improve existing functionality. The team has been considering how they should screen and prioritise the feedback more effectively. A member of the development team explained the dilemma: ‘the more users you get, the more feedback you get. Huge amounts of feedback. And it's only positive that you get lots of feedback, but how do you screen it? How can we ensure that what ends up on the backlog is what the company actually need, and not just what users want. We need to look at this. How can a screening be conducted?’

The CoConstruct members are discussing if the construction department, the owner of the CoConstruct application, should establish and own a kind of ‘change forum’ where they discuss and screen incoming feedback. They already have existing arenas where different representatives meet and discuss topics related to work improvement, and the CoConstruct application could be a natural part of this, the project manager explains.

The benefits of involving super users. The people involved in the CoConstruct project found the involvement of super users in different phases very useful. An informant from the CoConstruct project said that the user-centred approach was highly appreciated by super users as it allowed them to have genuine influence on the system design. One from the development team explained: ‘some of them stated that they felt heavily involved for the first time. This led to two great benefits; firstly, we got involvement from the users, and secondly; the word started spreading about what we were doing. The users were also feeling that we are working hard to adapt the app to the way they work’. This is also reflected by the super users, as one them said: ‘the development team are very eager and want to hear what we think is needed and not. We've been listened to’.

Super users reported that the CoConstruct application provided benefits to both construction workers and foremen. Particularly, in terms of increased work efficiency and empowering operators to further take control over their own work progress. As a super user said: ‘I notice that many engage in reporting work progress using the app. You can see your own work progress and completion. Then it is more enjoyable to do the work. I think it makes the workers feel that they own their work’.

Both super users and developers appreciated the use of visual tools prepared by the UX teams when discussing different solutions. The visualisations made it easier for users to understand the product and share their opinions. They also explained that the experience map, for example, helped to them establish a mutual understanding of the wider socio-technical system, not just the application. The wireframes and clickable prototypes were also highly appreciated by the developers. As one developer said: ‘It is a lot easier to us programmers to talk about solutions when you see a UX design, rather than just reading text. I've been in the business for 10 years, and so far, I've never seen UX been used as a basis to development. It has often been a "developer design"; a design solution that the developer feel is a good design’

5 Discussion

5.1 A continuous delivery approach to user-centred system design

The super users had a central role as user representatives when the development team has taken its first steps towards a more agile and user-centred approach. Whereas the Mill Co. has a long tradition of utilising super users, they have not been involved in the development process to a similar extent before. Contradictory to agile principles, the CoConstruct project began with a rather extensive design up-front phase, in which super users were involved in defining the final system requirements that yielded a large initial work package.

While up-front discovery activities are recommended in the context of user-centred agile software development, there is a lack of knowledge regarding the recommended extent of this phase [12]. In the CoConstruct case study, further development and refinement of the application was needed even after a majority of the initial use cases were implemented, which led to the project contract being extended several times. Particularly, delivering production versions continuously enabled users to identify application features that was not suited to their work, which led to additional backlog entries. In this respect, continuous delivery provides mechanism that could be aligned with a design-in-use approach.

Although issues related to system implementation is often discussed within information systems research [9], it should perhaps also take a larger role in software engineering. In particular, as when moving towards continuous software engineering the boundaries of implementation and development gets blurred. While super users had an important role in influencing the system design in this project, they also were integral to the application being adopted. Some even voluntarily committed to arranging training and extended support for those struggling to take the digital leap. To ensure continues software engineering that produces useful, usable, as well as *used* software, one perhaps need to take a more holistic approach [33]. Skilled workers continuously configure the socio-technical systems they are part of through working around technical deficiencies to enhance work efficiency [34]. A continuous, user-centered system engineering approach could allow them to re-design also the technical part of the system to support such workarounds as these occur.

The approach adopted in the project somewhat reassembles the framework proposed by Kuusinen [16] in that development was carried out in multidisciplinary teams collaborating within the same iteration. However, the rather long feedback loop after a crucial deficiency was identified by the user was a significant bottleneck. This issue is interwoven with the question on how to screen a large amount of feedback to rapidly identify issues that matters the most. In addition to an enhanced dialog between users and developers implemented, for example, through a change forum, a more systemised approach to collecting feedback could be developed [26]. This could include combining descriptive contextually rich feedback with quantitative metrics of application usage. If issues related to a specific version for a specific department is detected early on, a simple version rollback could be instantiated for a limited group of people [35].

5.2 Study limitations

Through a triangulation strategy, in which multiple data sources have been utilised and both authors being involved in the data collection and analysis we have sought to enhance research validity. Our prolonged involvement could also reduce respondent bias. A large part of the qualitative data was also recorded and transcribed to reduce researcher bias. However, due to time constrains not all interviews were transcribed verbatim. Another limitation of the study is the small number of involved super users. Further, as access to super users was managed by through a gateway inside the company, it is uncertain whether the selection was representative. While, there has been

reported issues with adoption of the CoConstruct application at some departments, this was not the case at the departments of the super user that we interviewed.

Whereas the user-centred approach investigated in this study mainly focus on usability and system adoption, there are several other aspects such as, business strategies, security and privacy that need to be considered when developing and implementing IT-systems in an organisation. To this end, end-user involvement might not be sufficient, but participation of experts in other domains need to be included. However, we consider this as future work.

6 Conclusion

Through a case study we investigate how end-users representants are involved in the software development process and their influence on system design and adoption. The study shows that super-users have been central in forming the developed system according to their needs. Continuous delivery of system functionalities provides an interesting opportunity for end-users to further shape the system design along with work practices. This allows considering also aspects of work that often is overlooked in up-front design. Super users also had a central role in system adoption, taking on a role as ambassador, providing training and support to their fellow work colleagues.

References

1. Bossen C. Chapter 5 - Socio-technical Betwixtness: Design Rationales for Health Care IT. Designing Healthcare That Works. Academic Press; 2018. p. 77-94.
2. van der Velden M, Mörtberg C. Participatory Design and Design for Values. In: van den Hoven J, Vermaas PE, van de Poel I, editors. Handbook of Ethics, Values, and Technological Design: Sources, Theory, Values and Application Domains. Dordrecht: Springer Netherlands; 2014. p. 1-22.
3. Simonsen J, Robertson T. Routledge International Handbook of Participatory Design. Taylor & Francis; 2012.
4. Ritter FE, Baxter GD, Churchill EF. Foundations for Designing User-Centered Systems: What System Designers Need to Know about People. London: Springer London; 2014.
5. Mumford E. The story of socio-technical design: reflections on its successes, failures and potential. Information Systems Journal. 2006;16(4):317-42. doi: 10.1111/j.1365-2575.2006.00221.x.
6. Abelein U, Paech B. Understanding the Influence of User Participation and Involvement on System Success – a Systematic Mapping Study. Empirical Software Engineering. 2015;20(1):28-81. doi: 10.1007/s10664-013-9278-4.
7. Mohagheghi P, Jørgensen M. (2017). What Contributes to the Success of IT Projects? Success Factors, Challenges and Lessons Learned from an Empirical Study of Software Projects in the Norwegian Public Sector. IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C). p. 371-3.
8. Bano M, Zowghi D, Rimini F. User satisfaction and system success: an empirical exploration of user involvement in software development. Empirical Softw Engg. 2017;22(5):2339–72. doi: 10.1007/s10664-016-9465-1.

9. Iivari J, Isomäki H, Pekkola S. The user – the great unknown of systems development: reasons, forms, challenges, experiences and intellectual contributions of user involvement. Information Systems Journal. 2010;20(2):109-17. doi: 10.1111/j.1365-2575.2009.00336.x.
10. Lwakatare LE, Kilamo T, Karvonen T, Sauvola T, Heikkilä V, Itkonen J, et al. DevOps in practice: A multiple case study of five companies. Information and Software Technology. 2019;114:217-30. doi: https://doi.org/10.1016/j.infsof.2019.06.010.
11. Dittrich Y, Nørbjerg J, Tell P, Bendix L. (2018) Researching Cooperation and Communication in Continuous Software Engineering. IEEE/ACM 11th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE). p. 87-90.
12. Brhel M, Meth H, Maedche A, Werder K. Exploring principles of user-centered agile software development: A literature review. Information and Software Technology. 2015;61:163-81. doi: https://doi.org/10.1016/j.infsof.2015.01.004.
13. Salah D, Paige RF, Cairns P. (2014). A systematic literature review for agile development processes and user centred design integration. Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering. London, England.
14. Hussain Z, Slany W, Holzinger A. (2009). Current State of Agile User-Centered Design: A Survey. In: Holzinger A., Miesenberger K. (eds) HCI and Usability for e-Inclusion. USAB 2009. LNCS, vol 5889. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-10308-7_30
15. Silva TSd, Martin A, Maurer F, Silveira M. (2011). User-Centered Design and Agile Methods: A Systematic Review. Agile Conference 2011. p. 77-86.
16. Kuusinen K. BoB: A Framework for Organizing Within-Iteration UX Work in Agile Development. In: Cockton G, Lárusdóttir M, Gregory P, Cajander Å, editors. Integrating User-Centred Design in Agile Development. Cham: Springer International Publishing; 2016. p. 205-24.
17. Harris MA, Weistroffer HR. A new look at the relationship between user involvement in systems development and system success. Communications of the Association for Information Systems. 2009;24(1):739-56.
18. Bano M, Zowghi D, Rimini Fd. User Involvement in Software Development: The Good, the Bad, and the Ugly. IEEE Software. 2018;35(6):8-11. doi: 10.1109/MS.2018.4321252.
19. Haake P, Kaufmann J, Baumer M, Burgmaier M, Eichhorn K, Mueller B, et al. Configurations of User Involvement and Participation in Relation to Information System Project Success. Cham: Springer International Publishing; 2018. p. 87-102.
20. Kujala S. User involvement: A review of the benefits and challenges. Behaviour & Information Technology. 2003;22(1):1-16. doi: 10.1080/01449290301782.
21. Bano M, Zowghi D. Users' involvement in requirements engineering and system success. 2013 3rd International Workshop on Empirical Requirements Engineering (EmpiRE)2013. p. 24-31.
22. Bosch J. Continuous Software Engineering: An Introduction. In: Bosch J, editor. Continuous Software Engineering. Cham: Springer International Publishing; 2014. p. 3-13.
23. Fitzgerald B, Stol K-J. Continuous software engineering: A roadmap and agenda. Journal of Systems and Software. 2017;123:176-89. doi: https://doi.org/10.1016/j.jss.2015.06.063.
24. Wüest D, Fotrousi F, Fricker S. Combining Monitoring and Autonomous Feedback Requests to Elicit Actionable Knowledge of System Use. Cham: Springer International Publishing; 2019. p. 209-25.

25. Johanssen JO, Kleebaum A, Bruegge B, Paech B. (2019). How do Practitioners Capture and Utilize User Feedback During Continuous Software Engineering? IEEE 27th International Requirements Engineering Conference (RE). p. 153-64.
26. Stade M, Fotrousi F, Seyff N, Albrecht O. (2017). Feedback Gathering from an Industrial Point of View. IEEE 25th International Requirements Engineering Conference (RE). p. 71-9.
27. Hege-René Hansen A, Anders IM. Super Users and Local Developers: The Organization of End-User Development in an Accounting Company. Journal of Organizational and End User Computing (JOEUC). 2006;18(4):1-21. doi: 10.4018/joeuc.2006100101.
28. McNeive JE. Super Users Have Great Value in Your Organization. CIN: Computers, Informatics, Nursing. 2009;27(3):136-9. doi: 10.1097/01.NCN.0000336479.50737.d8.
29. Dittrich Y, Bolmsten J, Eriksson J. End User Development and Infrastructuring – Sustaining Organizational Innovation Capabilities. In: Paternò F, Wulf V, editors. New Perspectives in End-User Development. Cham: Springer International Publishing; 2017. p. 165-206.
30. Torkilsheyggi AMá, Hertzum M. Incomplete by Design: A Study of a Design-in-Use Approach to Systems Implementation. Scandinavian Journal of Information Systems. 2017;29.
31. Yin RK. Case Study Research: Design and Methods. SAGE Publications; 2009.
32. Gaver B, Dunne T, Pacenti E. Design: Cultural probes. interactions. 1999;6(1):21–9. doi: 10.1145/291224.291235.
33. Forbrig P. (2017). Does Continuous Requirements Engineering need Continuous Software Engineering? 3rd Workshop on Continuous Requirements Engineering. Essen, Germany: CEUR.
34. Cabitza F, Simone C. “Drops Hollowing the Stone”: Workarounds as Resources for Better Task-Artifact Fit. Springer London; 2013. p. 103-22.
35. Agarwal A, Gupta S, Choudhury T. (2018). Continuous and Integrated Software Development using DevOps. International Conference on Advances in Computing and Communication Engineering (ICACCE). p. 290-3.