Proceedings of Informing Science & IT Education Conference (InSITE) 2010

Introducing Scrum in Companies in Norway: A Case Study

Elin Brekkan and Eystein Mathisen
Bodø Graduate School of Business, Bodø, Norway

elin.brekkan@hibo.no eystein.mathisen@hibo.no

Abstract

This paper describes a case study about introducing, implementing, and using Scrum in three software development companies in Norway. The study is based on several interviews and the data was analyzed using grounded theory practices. The motivation for this case study is to increase knowledge on the process of adopting Scrum in a company. In addition, we wanted to identify problematic issues regarding introduction and adoption of Scrum, issues that will be used to formulate further studies.

The elements described in this paper are aspects of Scrum that the companies consider important. Specifically, we describe how Scrum was introduced in the companies, how it affected overtime, how self-organizing teams were handled, how the roles in Scrum functioned, and the relationship of the Scrum Team to the rest of the organization. The findings indicate that sudden implementation of Scrum is a painful process and that the role of the Scrum Master is unclear. Furthermore, there may be a lack of common understanding of the concept of self-organizing teams. In addition the companies did not experience the productivity gain they expected and finally, introducing Scrum implies large changes in organizational culture and work practices that may be difficult to adapt to.

Keywords: Scrum, agile methods, self-organizing teams, introducing Scrum, roles in Scrum, system development processes.

Introduction

Building software is complex. The IT industry has struggled with software projects that do not finish on time and have large cost overruns (Szalvay, 2004; Tichy & Bascom, 2008). To handle problems in building software, different methodologies and techniques have been used. In the 1980's and early 1990's there was a widespread view that plan based, controlled and rigorous software development processes was the best way to achieve better software (Sommerville, 2007). Dissatisfaction with such an approach led to proposals of new agile methods in the late 1990s (Sommerville, 2007). The core to "agile software development is the use of light-but-sufficient rules of project behavior and the use of human- and communication oriented rules" (Cockburn, 2007). Agile methods focus on embracing changing requirements and the software itself and rely on an iterative approach where increments of the software are delivered to customer continuously and within small periods of time (Sommerville, 2007). When new agile methods were proposed, the IT industry embraced them

(Beck, 1999). Scrum is one of these agile methods and has gained considerable popularity within the IT industry (Dybå & Dingsøyr, 2008; Marchenco & Abrahamsson, 2008; Maurer & Melnik, 2006). Some characterize Scrum as the "face of agile" and "the most popular exponent of agile over the past few years" (Danube, 2009). At the same time, a recent review of empirical studies within agile software development suggests that more empirical studies of agile development are needed (Dybå & Dingsøyr, 2008). This same review also shows that most research has focused on the agile method XP (eXtreme Programming), whereas little research has been done on Scrum. Thus, in light of Scrum's popularity, more research is needed (Dybå & Dingsøyr, 2008).

The motivation for this case study was to increase knowledge about Scrum and focus on the process of adopting Scrum in a company. The case study describes the experiences of introduction, implementation, and use of Scrum in three software development companies in Norway. Two of these companies can be characterized as small, while one is midsized. Important aspects of the companies' experiences are highlighted.

The remainder of the article is structured as follows. "Background" describes our research method, presents information about three companies that were investigated, overviews Scrum, and describes the situation of the companies prior to introducing Scrum. "Content" describes the case study and experiences of introducing Scrum. The elements described in this section are aspects of Scrum that the companies considered important. We describe how Scrum was introduced in the companies, how it affected overtime, how self-organizing teams were handled, how the roles in Scrum functioned, and the relation of the Scrum Team to the rest of the organization. Finally, a conclusion is presented.

Background

This section describes the method used for the case study and presents background information of the companies.

Method

Data collection

The study is based on semi-structured interviews, of both developers and managers in three companies in Norway. Semi-structured interviews are used by many, since it may generate a rich description of individual's experience (Coleman & O'Connor, 2007).

Prior to the interviews an interview guide was created that included a set of questions; however, participants were free to elaborate on the questions and add other comments and issues. The interview guide contained a number of issues and general questions about the interviewees' experience of introduction and adoption of Scrum. The issues we identified dealt with themes we regarded as central to our purpose with the study; namely the process of introducing and adopting Scrum in a company and to identify problematic issues regarding this process. The questions functioned as reminders for us and were asked when we wanted to encourage the interviewees to detail and elaborate on the issues. They were an aid to get all themes related to an issue covered or elaborated.

Figure 1 shows an extract of issues and questions from the interview guide.

Issues	**Questions**
Earlier situation	• Which development method did you use prior to Scrum?
	• How did this method function?
	Advantages/problems?
	• Describe the organization
Introduction of Scrum	• Who introduced the idea of Scrum?
	• Who was responsible for the introduction?
	• How was the implementation planned?
	Who participated?
	How long time did you spend on planning?
	Any pilot projects?
	• How was the organization prepared?
	Was the whole organization prepared?
	• Was the organization motivated/committed?
	• How was the organization reorganized?
	• …………..

Figure 1 An extract from the interview guide

The interviews were carried out in two rounds. In the first round employees in all three companies were interviewed. We saw, however, that we had more questions to company A, so we took a new round and interviewed three more people in this company.

Fourteen persons were interviewed: six from Company A, five from Company B, and three from Company C. Both managers and developers affected by Scrum were interviewed.

The interviews accomplished the following:

- Company A: three persons were interviewed as a group and three were interviewed alone. Of these, three were managers and three were developers.
- Company B: Four persons were interviewed in groups of two, and one person was interviewed alone. Of these, three were managers and two were developers.
- Company C: two developers were interviewed as a group, while the manager was interviewed alone.

All interviews were taped and transcribed.

Data analysis

In order to be able to identify the key themes and to compare the data collected, grounded theory practices were used (Glaser & Strauss, 1999), (Charmaz, 2000). Grounded theory is also known as the constant comparative analysis model. NVivo was used to code and examine the data and perform data analysis. NVivo is a qualitative data analysis software tool and has been designed for qualitative Data Analysis (Bazely, 2007).

After transcription, all interviews were coded in NVivo and analyzed. During this process we used an open coding method through line-by-line coding where pieces of data where coded with concepts or descriptive terms that were grounded in the text. Throughout this process we constantly compared the pieces of data with each other (a constant comparative approach), looking for similarities and differences. This means that each new incident was compared to all incidents recorded before, in order to develop the final categories and their sub categories. The coding process increased our knowledge of the data and we got a good understanding of what was going on. The coding process gave a good description of what the interviewees talked about.

We also created models of the coded nodes. The models visualized the cases, gave us an overview of the coded nodes and gave room for comparison and discussions of the different cases. During this stage we found relationships and connections among the nodes and the models gave support for reflections, analysis and examinations of the cases that were coded. In addition, the modeling process aided us in developing the categories and sub categories.

A total of 32 categories were initially found and we ended up with 8 categories. These 8 categories explained the main concerns in our data. Figure 2 gives an overview of the categories and their subcategories that we ended up with. The categories are the yellow circles, whereas the sub categories are the text attached (by a line) to the categories.

A journal was written throughout the coding - and the modeling process, where we noted our findings, reflections on the code and models, questions that arose and issues that should be object for further research.

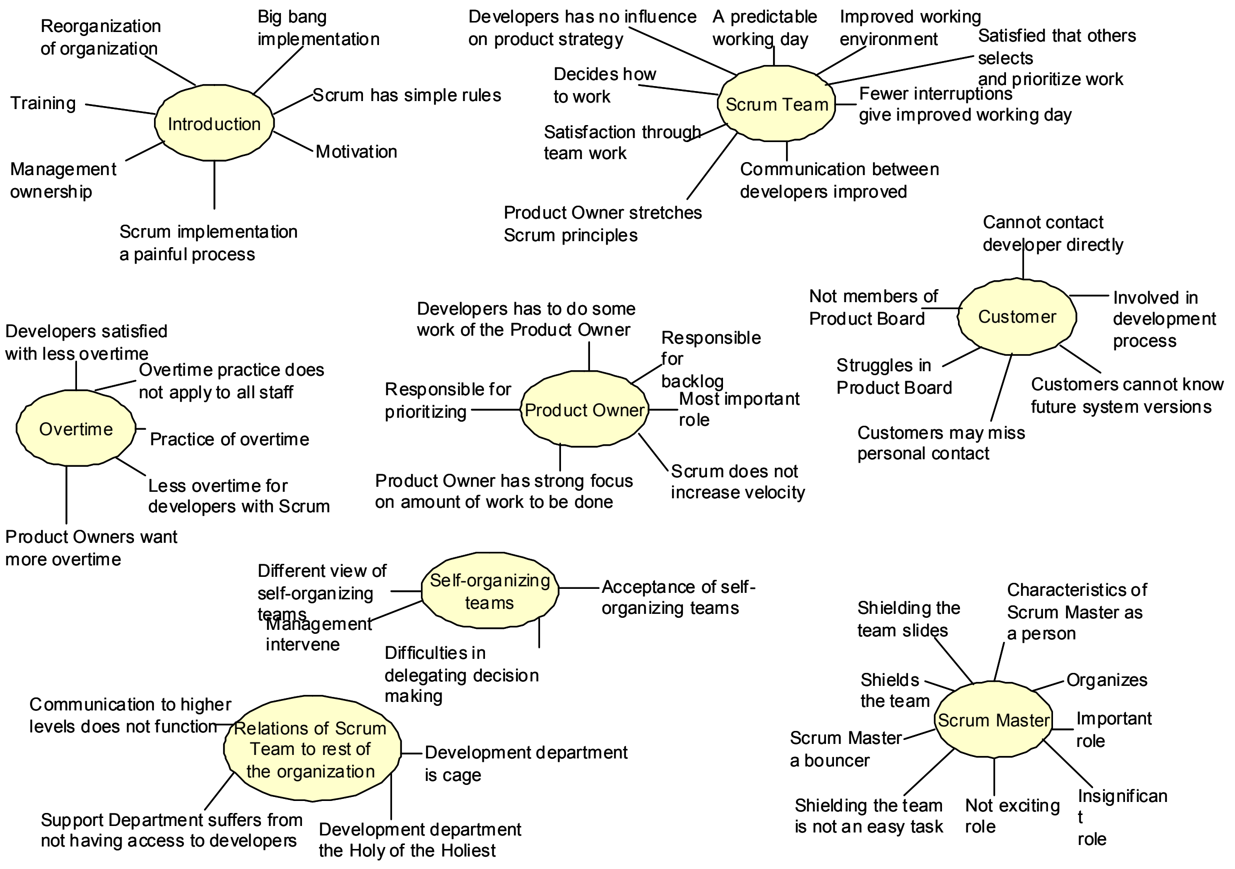
Introduction
Reorganization of organization
Big bang implementation
Training
Scrum has simple rules
Motivation
Management ownership
Scrum implementation a painful process
Scrum Team
Developers has no influence on product strategy
A predictable working day
Improved working environment
Satisfied that others selects and prioritize work
Decides how to work
Fewer interruptions give improved working day
Satisfaction through team work
Communication between developers improved
Product Owner stretches Scrum principles
Customer
Cannot contact developer directly
Not members of Product Board
Involved in development process
Struggles in Product Board
Customers cannot know future system versions
Customers may miss personal contact
Overtime
Developers satisfied with less overtime
Overtime practice does not apply to all staff
Practice of overtime
Less overtime for developers with Scrum
Product Owners want more overtime
Product Owner
Developers has to do some work of the Product Owner
Responsible for backlog
Responsible for prioritizing
Most important role
Product Owner has strong focus on amount of work to be done
Scrum does not increase velocity
Self-organizing teams
Different view of self-organizing teams
Management intervene
Acceptance of self-organizing teams
Difficulties in delegating decision making
Relations of Scrum Team to rest of the organization
Communication to higher levels does not function
Development department is cage
Support Department suffers from not having access to developers
Development department the Holy of the Holiest
Scrum Master
Shielding the team slides
Characteristics of Scrum Master as a person
Shields the team
Organizes
Scrum Master a bouncer
Important role
Shielding the team is not an easy task
Not exciting role
Insignificant role

Figure 2 Categories and sub categories

Based on these categories and their sub categories, we then ended up with the following themes for this article: Introducing Scrum, Overtime, Self-organizing teams, Roles and Relationship of The Scrum Team to the rest of the organization.

Company information

The case study is based on three software development companies in Norway, which are referred to as Company A, Company B, and Company C.

Company A

Company A develops information systems for the health care services. The main system manages electronic information related to patient records and is used by doctors, nurses, and clerks. In addition, the company delivers sub-systems to laboratories, radiology departments, and pure X-ray institutions. The company has 140 employees and locations in four cities in Norway.

The development of the information system started at a hospital in 1987, and in 1989 several hospitals in Norway decided to use the system. Today the company delivers the system to 55 percent of all hospitals in Norway, geographically spread all over the country. The company is a joint venture and the economic situation is good. The result in 2008 was 4, 3 million Norwegian kroner before taxes and the turnover was 138 million Norwegian kroner.

Company A is organized with one centralized administration and several line departments. The line organization deals with all personnel, technical, and administrative conditions. Production occurs in projects or teams.

The company has large projects delivering software to customers. Consultants from the company's consultant department and from a second firm participate in these projects with the customer. All projects developing software are organized according to Scrum.

The company has little staff turnover.

Figure 3 shows the organization of the company.

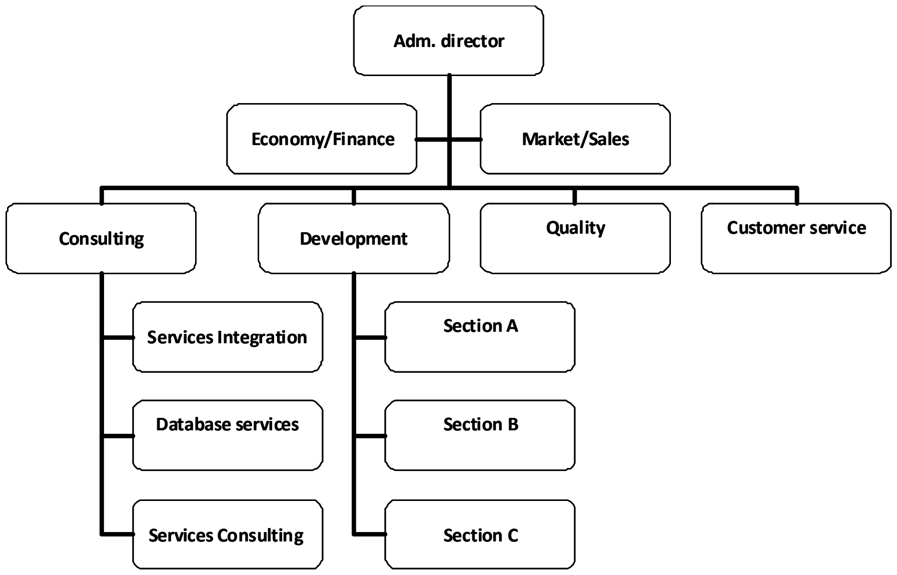

Figure 3 Organization of Company A

Company B

Company B is the largest company delivering ticket systems for cultural events in the Nordic region. The system is used by cinema, sporting, theater, and concert arrangers. The company is located in Norway and has a subsidiary in Sweden.

The company has 30 employees. The turnover in 2008 was 24 million Norwegian kroner.

Figure 4 shows the organization of the company.

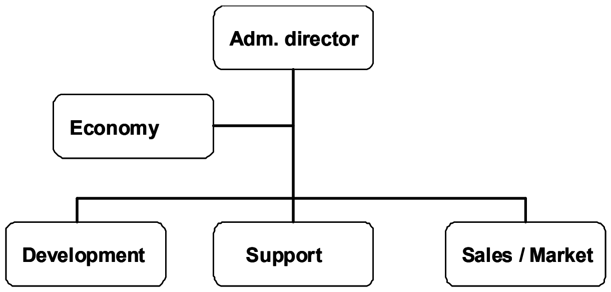

Figure 4 Organization of Company B

In Company B, all software development is now organized according to Scrum practices.

Company C

Company C, established in 2006, started out as consulting company with five employees. These consultants are hired out to other software development companies.

In 2007 the company decided to develop its own software. For a period, staff met and discussed the kind of software product they should develop and finally decided to develop a system related to managing board meetings in large companies. In 2008 Company C began developing this system. They established a group to develop the system, which consisted of six experienced developers. During this development period they used Scrum.

Due to the financial crises customers stopped financial funding of the project in 2009, and thus, the development of the system is momentarily on hold. The data collected and the research for this paper focuses on the period of development of the system for board meetings.

Today, Company C is a pure consulting company that delivers services within the areas of system development, system integration, and database technology. The company has five employees, four of whom are system developers.

Turnover in 2009 was 5 million Norwegian kroner and expected result is 0, 5 million Norwegian kroner.

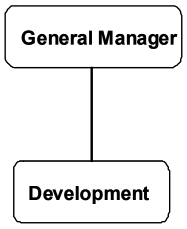

Figure 5 Organization of Company C

Scrum developers use most of their time developing software, and the Scrum Master acts as a firewall in terms of interruptions and interference. The Scrum Team is *shielded* so that the developers can concentrate on finishing the work.

At the Daily Scrum meetings the Scrum Master asks the team if there are any impediments; if so, the Scrum Master is responsible for removing and resolving them to enhance the team's productivity. The team simply expects and assumes that the impediments will be removed so that it can continue to progress and continue to be focused and effective.

When a Sprint is finished, a *Sprint review* and a *Sprint Retrospective meeting* are held. At the Sprint review meeting the team demonstrates the product increment to management and customers. The functionality is reviewed and its value considered. The meeting is informal and questions and discussions are allowed. Based on the review, decisions are made about what should be developed next. At the Sprint Retrospective meeting the team and the Scrum Master come together to assess what went well during the Sprint and what improvements can be made for the next Sprint. The aim is to make improvements for the next Sprint.

Setting the Stage

Here we describe the earlier situation, prior to introducing Scrum, for all three companies.

Company A

Prior to introducing Scrum, Company A followed the classical waterfall model and used Microsoft Solutions Framework (MSF) process model. Following the waterfall model they had long iterations of development, with delivery of the software several months later. The developers spent much time on specifying the system before they started to code. When they finally delivered the system to the customers they experienced that the system did not fulfill the customers' requirements. They also experienced that following the waterfall model led to a mentality where a lot of work was done at the end of the development period and that code was not tested during development. The software contained many errors and the developers had to work much overtime to correct the errors. In short they experienced that the period before delivering the system to the customers were chaotic and came out of control.

A manager in Company A said that they experienced the waterfall model as cumbersome. When customers had new or changed requirements, they experienced that it was difficult to adapt to these, since they already had committed to a plan that described the functionality that should be delivered. In addition the method made it difficult to make effective use of new employees. The last couple of years the company had hired many new employees and the manager expressed that the waterfall model made it difficult to gain profit proportional to the growth of employees.

Developers describe the earlier situation as stressful. Customers took direct contact with developers and wanted developers to solve urgent problems quickly. It was then up to each developer to prioritize tasks and then decide which tasks to carry out. Each developer ended up with their own personal backlog of tasks that they had to deal with. In time the backlog grew as more and more tasks were to be handled and a developer expressed that:

> "You get your own personal backlog that you feel you have to solve. And then you get squeezed, because the more you work, the more tasks come in. So you cannot go home and say that you have finished the work of the day. You bring it along and you get bad conscience for not having solved all the things you fell you ought to have solved."

Company B

Prior to the introduction of Scrum, the work situation for the developers in Company B was complex and unclear. Employees described their situation as ad-hoc; i.e., they never knew what they were going to work with and how their working day would be structured. Customers or stakeholders from different parts of the organization were allowed to interrupt their work and instruct them to do other work that they considered important. Work to be done was not clearly prioritized; thus, tasks that were considered important for the moment had to be carried out. These interruptions happened repeatedly and led to half-finished work and lack of overview of work to be done. The organization suffered from a fire-fighting style of work and absence of plans. Developers complained that they could not plan their working day.

The developers' experience was that management and control were absent. Projects were not followed up, people in the organization and developers were unaware of what others were doing, and resources were scattered. In addition, projects were delayed, which led to negative publicity in the press.

People who had worked for the company for a considerable period experienced particular pressure. Since they knew the system well, they were always called on when a critical situation occurred. This led to much overtime for the experienced people, while others may have had little work to do. In addition, the amount of work was not well balanced among the developers.

This situation led to a poor working environment and low morale among employees. People quit and started to work elsewhere. As one developer said,

> "You might say that the hallmark of previous times was that a developer was totally worn out after a week or a month because you never had predictability in what kind of work you were going to carry out when you arrived at work in the morning. You thought you should carry on with some task, but when you came in the door, there was somebody that gave you another task that was critical."

Company C

Company C had no experience developing software since it was a start-up. This was the company's first major project, which experienced developers, coming from different companies, had been hired to work on. The developers had all previously used the waterfall model when developing software and some had some experience with XP. None had experience with Scrum.

Content

Facing the problems described above, the organizations decided to introduce Scrum as a method to develop software.

Introducing Scrum

In all three organizations management was committed to Scrum. In Company A the top manager initiated the idea of implementing Scrum and in companies B and C the initiation came from correspondingly a development manager and the developers themselves. Top management was enthusiastic and decided that they would use Scrum.

All employees received training in Scrum. In addition, Company B had workshops in all departments and discussions and debates about the consequences of implementing Scrum in the organization. Company A had follow-up training in addition to the initial courses. As one developer in Company A stated,

> "I have to say that management has taken the introduction of Scrum seriously. We have had two of the world's best lecturers on Scrum here. And the inventor of Scrum has also been here and told us of the process. So management has taken that part seriously."

The employees in all three organizations were highly motivated to start, and enthusiastic about, using Scrum. A developer in Company A indicated

> "High motivation right down the line."

A developer in Company B stated,

> "And it was very good that when the company introduced Scrum, the whole company was behind the decision; it was implemented in full understanding with the whole organization from day one."

Prior to implementing Scrum, the organizations were reorganized. Roles were identified and people were allocated to the roles of Scrum Masters and Product Owners. Developers were then allocated to the Scrum Teams.

Company A established ten Scrum Teams and eight Product Owners, with two Product Owners responsible for two Scrum Teams. Eight Scrum Masters were established and two Scrum Masters were responsible for two teams. The ten Scrum Teams all worked on one large software system, but worked on different part of the system; that is, they were responsible for different parts of the system. Companies B and C defined a single Scrum Team with one responsible Product Owner and one responsible Scrum Master.

Since Company A had as many as eight Product Owners, they also defined and established three so called Product Line managers. That is, they got a product management hierarchy with three Product Line managers where two Product Line managers were responsible for three Product Owners and one was responsible for two Product Owners.

In both Company A and Company B a Product Board was defined and set up. One of the tasks to the Product Boards is to prioritize tasks; that is, prioritize what functionality to deliver to the customers in the next versions of the data system. In company A the Product Board consists of top manager, the three product line managers, research – and development manager, marked/sales director and a quality manager. In company B the Product Board consists of support manager, sales – / market manager, manager from the subsidiary in Sweden, development manager and the director.

When the training and the reorganization of the organizations were finished, the companies implemented Scrum for all software development projects at once, without any testing or pilot projects. In company A, however, they had a plan to test and verify Scrum in two pilot projects before they rolled Scrum out in the whole company. Two projects then started to use Scrum, but after a couple of months there was a pressure from top management to implement Scrum for all development projects in the company. Management would not spend time on testing Scrum. In our study we found some interesting metaphors used about this kind of implementation. In Company A, interviewees used the concept "big bang" implementation of Scrum and in Company B a developer described the implementation of Scrum as "Pang. Now we have to use Scrum".

All interviewees thought that the ground for implementing Scrum in the organizations was well prepared. Despite this, all organizations indicated that implementing Scrum was a painful process and that it took time to establish the new ways of working. A developer in Company A said,

> "When you start to use Scrum you shouldn't have an idealized picture of it; it takes a while before you reach the vision that you feel that we are there, that it works, that the process as a whole works."

Employees all agreed that Scrum as a method is simple with simple rules. But to change the normal way of working seemed difficult. A developer in Company A expressed this by saying that

> "Scrum has…simple rules, but changing a way of life and living by these rules is not necessarily easy: many people have to change their way of thinking and work, and when everybody has to do it at the same time [doing so can be difficult]."

Common for all three companies were the immediate implementation of Scrum, and all experienced implementation as painful. Thus, using the "big bang" method to employ Scrum may be questionable. It is possible that organizations should do a more comprehensive investigation of

Scrum and how it fits into own companies, before adopting Scrum. This could make implementation less painful and problematic. In our case study all employees were trained in Scrum, but there was no reflection and research on how Scrum could be optimally adopted and integrated into the organizations. Lindvall et. al. (Lindvall et al., 2004) suggest that that an organization understand the effects and side effects of a new agile method before introducing it. Furthermore, they state that the conditions in which agile methods work are unclear, and that organizations should see compelling evidence that the agile method works within their context. One way of getting this information is to run pilot projects.

Overtime

Scrum has no particular rules about overtime, but the Scrum Teams in all organizations promoted the agile principle of a 40-hour work week (Beck, 1999).

In line with other research (Mann & Maurer, 2005), the developers' experience was that after Scrum implementation they worked less overtime. Our research found that this did not mean that they did not work overtime, but that they worked less overtime. Prior to Scrum it was not uncommon for developers to often work long days with much overtime. A developer in Company B said,

> "I've gone from working a 70-hour week and eating pizza and drinking soda, to end the day at four o'clock and keep the rest of the day for myself."

A developer in Company A stated,

> "I think that when it comes to the developers the overtime is reduced a good deal. Not this month, because I know that overtime is planned. But for developers overtime is reduced."

And a Product Owner in the same company stated,

> "Here we have the practice that the team should not work overtime; that is, there shall be no extra effort to reach the goals."

For Company A, however, it seemed as though the agile principle of 40-hour work week was more promoted among developers than among other types of staff. It seemed that there was a greater acceptance that Product Owners worked overtime. As a Product Owner in Company A said,

> "For developers the overtime is much less than before. For the Product Owners,…, I've worked 130 percent this spring; and that's too much. I do intend to reduce overtime."

All developers in the companies expressed satisfaction regarding reduced overtime, which, at the same time, created dissatisfaction among the managers. Product Owners wanted the developers to work more to reach the Sprint goal and resisted taking tasks out of a Sprint. But since their desire for more overtime was not acceptable according to agile principles, they felt that they could not order the team to work more. A Product Owner in Company A expressed this frustration:

> "I feel that we have removed tasks during the first four or five Sprints, instead of ordering the team to work overtime when there is a fire and work has to be carried out."

A Product Owner from Company B expressed a similar feeling:

> "The best thing is that the team itself [can] initiate overtime in order to reach the Sprint goal. Sometimes we just have to order the team work overtime."

Self-organizing Teams

All organizations accepted the idea of self-organizing teams. When the Scrum Teams had committed to a Sprint backlog, the teams themselves were allowed to decide how to carry out the tasks. They also evaluated themselves and in their Daily Scrum meeting reported to the Scrum Master on how the work was proceeding. The Product Owner in Company C said,

> "The developers were mostly allowed to work in peace and make their own decisions."

And the Product Owner in Company B stated that

> "The developers manage themselves now. It's not up to me to decide how they do the changes in the database or how they change the user interface; that's up to the team."

But all organizations experienced different problems regarding self-organizing teams. Developers in Company A experienced that the organization did not have a uniform view on what self-organization meant and that employees had different views on what the teams could decide and make decisions about. A developer said,

> "There are different views in the company on what self-organization is. There has been some fuss around it: what is self-organization? What levels of self-organization is a team supposed to have? What decisions can the team make and what do they have to ask permission for?"

He further complained about management's difficulty delegating decision making:

> "It's a challenge with a management that is used to make decisions. It is hard to start using Scrum in [such] an organization."

It also seems managers have difficulties in "walking the talk"; that is, managers don't act in the way they say that they will act. In Company A, where they have ten Scrum Teams, developers particularly express problems concerning self-organizing teams when it comes to work that affects all the ten teams. A developer in this company stated that management has expressed in words that the Scrum teams are self-organizing, but sometimes they don't treat the teams as self-organizing. This is particularly seen when the teams try to solve problems that occur across the ten teams. A developer says that:

> "We tried to organize groups where people should work on important architectural elements in the system. When management got to know this, we were just stopped. They meant that we were not allowed to spend resources on this work.. … It is expressed that we are to be self-organizing, but from above we are not always treated like that"

Managers were also somewhat concerned about the principle of self-organization. In all companies, staff claimed that they sometimes had to intervene in the teams to ensure that the right decisions were made and that the team kept on track. The Product Owner in Company C stated,

> "I don't think, as a Product Owner, that Scrum works with just leaving it: by letting the group stay in its own world. As a Product Owner I have to identify and make the decisions on behalf of the developers. The result is necessarily not the best, but at least you see to that something is being done."

He further stated,

> "We had to use good old-fashioned management--straighten up things when things got too loose."

A Product Owner from Company B said,

> "[The team] claim[s] that they are in control, so we have to believe them. But then we will follow them up closely; if we don't see any improvement, then we know that they don't have control. Then something has to be done."

And a product owner in Company A stated that:

> "There have been a couple of cases where the teams have tried to self-organize; they wanted to plan and provide a common architecture for the whole system. This has been put dead. There has been a fear that they spend too much time on organizing, discussions and so on"

Similar findings are found in a study performed by Moe, Dingsøyr and Dybå (2009). In this study they found that it was difficult for teams to improve and one of the reasons for this was the low level of team autonomy.

These findings are also compatible with Cohn and Ford's (2003) research stating that many managers feels that traditional management gives them control and enable them to follow up on work. This control makes them reluctant to give up Gantt charts or other plan-driven processes. However, this same research also states that some managers feel satisfied when the group commits to carry out a certain amount of work within a certain time period, findings that are not supported by our research.

Another question that occurs is how agile are actually the companies? Management wanted to implement Scrum, but at the same time it seemed that it on times was difficult to leave the traditional command-control management. Cockburn and Highsmith (2001) say that agile organizations practice agile leadership-collaboration management. This means that management understands that it is not so important who makes the decisions, but the important thing is collaboration on information to make the decisions. And agile managers trust individuals "to apply their competencies in effective ways".

Roles

Scrum Master

All teams in the organizations had their own Scrum Master. In both Company B and Company C the interviewees stressed the importance of the Scrum Master role. They particularly emphasized the task of shielding the Scrum Team from interruptions. A Product Owner in Company B said that

> "[the Scrum Master] sees to [it] that all visitors keep away."

and that

> "support staff are not allowed to enter development department directly."

Developers in the same company saw the Scrum Master as a bouncer:

> "Scrum Master functions as a filter all the time. He is a bouncer."

In addition the interviewees in both companies listed characteristics that a Scrum Master had to possess in order to be able to carry out the role in a proper way. They said that the Scrum Master had to be a strong person and that (s)he had to be able to argue. These characteristics were related to the Scrum Masters tasks of shielding the team and following up the team. A Product Owner in Company B stated that:

> "The Scrum Master has to be resolute and resist interruptions, even if management wants them [to do something else]."

A Product Owner in Company C said, in relation to following up the Scrum Team,

> "The Scrum Master has to be merciless."

The two organizations recognized the importance of shielding the team, but also admitted that shielding the Scrum Team was not easy. In Company a B Product Owner indicated that "it is especially a challenge to make the sales and support departments feel that they are not being restrained [in their work] by not having direct access to the developers."

The Product Owner at Company C said, "It is my greatest challenge to respect the Scrum method and leave [the team alone]."

Initially, when Scrum was implemented, the principle of shielding the team was taken seriously and strictly maintained in all organizations. But in our research we found that this task was straying after a while. When talking about the subject a Product Owner in Company A said:

> "The teams open up more; maybe not as strict as initially. An approach will take place where things will become as pragmatic as they can be. Practical."

And a developer in the same company stated:

> "And we have perhaps seen that management doesn't always follow the methodology they have introduced; where they call in staff to meetings without contacting the Scrum Master first"

And a developer in Company B said:

> "It has strayed a bit, since we started. Initially it was forbidden to enter the development department."

In company A the interviewees confirmed that the task of shielding the team and the task of organizing the team's work as prominent tasks of the Scrum Master. But in this company we found that staff had a different view on the importance of the role of the Scrum Master. Here the interviewees thought the Scrum Master's role was quite small and insignificant. This finding made them quite surprised, because this role was emphasized as very important during their training. A developer stated that

> "The Scrum Master is very invisible, based on the fact that the Scrum Master is so central in the course. He is much less central on our real world."

The reason why the Scrum Master's role was looked upon as insignificant is not clear. But it could be related to the person who possessed the Scrum Master role and how he felt about the work of the Scrum Master. We found that he implied that the Scrum Master role was not an interesting role, and this could perhaps influence the way he exercised this role:

> "The work of the Scrum Master is not my dream work. It's not exciting"

Product Owner

In all companies the interviewees stated that the role of the Product Owner was the most important role. They saw the Product Owner as the person with most power regarding product strategy and what work the team should carry out. A Scrum Master in Company B said that:

> "Product Owner is on top in this hierarchy. This is the most important role…. This person is the most powerful person in the organization"

And a Product Owner in Company A stated:

> "The Product Owners have much power regarding what to do. The power is centered on the Product Owners."

This power seemed to be related to his/her responsibility for the product backlog and his responsibility for prioritizing tasks. This made him/her in turn responsible for how the product would develop in the long run. A Scrum Master in Company B said:

> "..he manages the product backlog; the log of all tasks that the team is going to work on. Everything is there. Nothing should be elsewhere."

The tasks that were emphasized were the responsibility of the product backlog and the responsibility of prioritizing tasks. In Company A it seemed to be a problem that the tasks, described as user stories by the Product Owner, were poorly written. Developers were not able to estimate the tasks due to lacking information in the user stories. Thus, it ended up with developers doing the work of the Product Owner. Similar findings are found in a study by Srinivasan and Lundquist (2009). They found that developers had to spend a great amount of time (around 50 to 75 percent of the developer's time) to refine the requirements in order to make them implementable.

In all companies it seemed that the Product Owner was the person that was most willing to stretch Scrum principles; that is, s(he) was willing to break Scrum principles in order to get some important work carried out. For example, all Product Owners found it difficult not to interrupt and go directly to team members to get some work to be carried out. A Product Owner in Company A says that:

> "Following the method, to go through Scrum Master, is a challenge when you want quick answers"

A developer in the same company said:

> "The Product Owner wants work to be carried out…, he is likely to stretch the principles"

And a developer in Company B said:

> "The Product Owner may have people in the other end that pushes ahead and want more tasks to be carried out"

The challenge to keep to the Scrum principles was also seen in relation to the amount of work that was to be carried out in a sprint. The Product Owners had a strong focus on being able to carry out as much work as possible during a sprint. The Product Owners complained that the teams had low velocity and that they wanted the teams to carry out more work than they had committed to. In some cases they therefore ordered the team to take more tasks into the sprint than originally was agreed on. A Product Owner in Company A said:

> "If we are going to deliver something this spring, we have to finish the work"

And a developer in the same company said:

> "A sprint wasn't a sprint, the sprint was continually filled with new tasks…The list [of tasks] just grew bigger and bigger. And it was very tempting for the Product Owner, who actually was more project oriented than Scrum oriented, to understand what we had committed to"

This urge to complete tasks, even if time does not allow, is also seen in Company B and Company C. Here it is accepted that instead of taking out tasks of sprint, the team was sometimes ordered to work overtime to be able to achieve a delivery date. A Product Owner in Company B said that:

> "It happens during a sprint that the content [of the system] is so important that the last thing we will do is to take something out [of the system], so overtime is a means to be finished."

In company C the product Owner said:

> "It was in a way their [the team] responsibility [to be able to carry out the work in the sprint], so they just had to hurry up and use overtime"

Similar findings are also found in the research of Moe, Dingsøyr and Dybå (2009). In this study they found that the Product Owner moderated the level of team autonomy, because he often had some emergent issues that he needed the team to do in an ongoing iteration. And this interruption made it difficult for the team to stick to the plan.

The Scrum Team

The developers that were interviewed all stated that their working environment had improved after Scrum implementation. Different aspects of Scrum were mentioned as factors that contributed to this improvement and we found that several of the interviewees related this improvement to team work, that is, co-operation with others gave satisfaction and that the team could organize they work according to own decisions. In addition predictability of what they were going to work with and fewer interruptions from other people played an important role.

Improvement of employee morale when transforming to agile method is also found in other research (Cockburn & Highsmith, 2001). A Product Owner in company B stated:

> "The most important thing is that I noticed a change in the organization. People started to smile and enjoy themselves. This heavy and depressive spirit was gone. .. They [the developers] now have a predictability regarding the tasks they are to accomplish. They have a freedom, they are able to self-organize and they can work with others."

In Company B a developer expressed his content of teamwork when he stated:

> "We feel we accomplish things because we co-operate; we [as a group] have become more gelled together… Everybody feel they can contribute, and they feel that they can participate."

And in Company A a developer said:

> "Now we're working in a team, and that means that we relate to the same persons on long term. So, socially we become well acquainted with the people we work with"

Developers particularly emphasized that they were more in control of their working day and that this lead to less overtime and stress. In company B the developers said that their working day now was more predictable and that they now knew what they were going to work with:

> "So we plan for a month ahead. And there is everything that we are going to accomplish. So it's predictable what we are going to do the next month"

A similar aspect of getting more control is also seen in company A. Here the developers were particularly satisfied that somebody else (the Product Owner) now prioritized and selected the work that should be accomplished. As for all companies, the developers experienced before Scrum that people contacted them directly in order to make them carry out tasks that were seen important at the moment. This resulted in personal backlogs of tasks for each developer and the backlogs just grew larger and larger over time. Each developer was then responsible for prioritizing and selecting the tasks that were to be carried out. With Scrum this changed and the developers were spared for the responsibility of prioritizing and selecting tasks to be carried out. This job was now left to the Product Owner. A developer expressed:

> "I think many developers have experienced that the responsibility you feel when your own personal backlog just grows and you feel that it is your own responsibility to solve it, was much greater before. Because you were the owner of it….. And then you prioritize many things, too many things because you want to do it… And you end up with the situa-

> tion that the more you work the more it escalates….So you cannot go home and feel you've finished today's work. Now you can concentrate on that thing [the task] and do it properly and finish it."

One of the Scrum Master's tasks was to shield the Scrum Team from interruptions. All developers in the companies expressed that this represented a huge improvement from their previous working situation. In our study we found that it was the teams themselves that were the ones that most strictly maintained this principle. In company B a Product Owner stated:

> "What is difficult is that the team has become so conscious of how they want their working situation. So when the director comes in and says: "do this", they [the developers] say: "Leave the room. Go to the Scrum Master.""

A developer in Company A said:

> "We [the developers] are most loyal to Scrum. Not Scrum Master or Product Owner. Product Owner wants things to be done."

Customer

Previous to implementing Scrum both Company A and Company B experienced that customers called directly to developers to get them to do tasks that were important to them. When Scrum was implemented this was not longer permitted and both companies experienced that they had to educate the customers to get along with this principle. A Product Owner in Company A said:

> "You just have to educate the customers. Make them get along with these concepts and ideas."

How customers experienced this is difficult to say, but some of the interviewees stated that this could be hard for some of the customer since they were used to personal contact with developers to get tasks to be done. In Company B a Product Manger said that:

> "It might be experienced by the customers, who were used to being able to get things done right away, that we have become more rigid….. I think that some customers have noticed it [Scrum] for both better and worse."

And a developer in Company A said:

> "I suppose that some customers miss the personal contact to get things done."

As mentioned in section "Developer" we found that developers found that their working day had become much more predictable after Scrum. This predictability, however, did not apply to customers according to a Scrum Master in Company A. In addition to his Scrum Master role, he also had a role of managing future versions of the system. That meant that he had to inform customers of what functionality new versions of the system would contain, and he thought that this task was not easy. Since Scrum, as other agile methods, embraces change, he meant it was impossible to inform the customers of future system versions and that this was a problem with Scrum.

> "I shall inform the customers of the content of the new versions. Previously we planned in a different way; it made things more predicable… So I feel that this is a problem with this methodology…. Prioritizations are changed much more often after Scrum. Previously we nailed the content and informed the customers. The customers were pleased, because they knew what was coming up"

All companies said that customers were involved in the development process. They were involved in Sprint planning, specification of new functionality and prioritization of tasks together with the Product Owner. What was questionable, however, was that there were no customers present in the Product Boards in neither Company A or Company B. This means that the final prioritizations of tasks were performed by the organizations themselves; that is in the Product Boards without customer representatives. This raised a question to what degree the customers actually have an influence on the future product and the functionality it will contain. It also seemed that it was not easy to prioritize tasks in neither Company A or Company B. In both of these companies we found that staff thought that the Product Boards meetings were hard regarding prioritization of tasks. In company A a Scrum Master said:

> "A part of it [the Product Board] shows sign of…., well I won't call it a power struggle, but there are discussions in the Product Board"

In company B a developer said:

> "There is a fight between the different fractions of the Product Board…. The person who shouted loudest before… it has moved into the Product Board"

Satisfied customers are of highest priority for agile methods. The first agile principle states that "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software" (Martin & Martin, 2007). In our research we found that customers were involved in the development process, but since they were not present in the Product Boards, it can be questioned if the involvement of the customers is optimal. The final questions regarding customer satisfaction are therefore: "Are the customers actually satisfied with the deliveries"? Or: "Do the deliveries create real value for them"?

Relationship of the Scrum Team to the Rest of the Organization

The Scrum Team is according to the Scrum rules shielded by the Scrum Master from the rest of the organization. The team's main focus is on carrying out the work that is defined in a Sprint backlog and reach the Sprint goal. It seemed like this way of organizing the team gives the team a special status within the organization. In our research we found some interesting metaphors that described the team and again could tell us how parts of the organization viewed the team. For example in Company B a Product Owner described the development department as "the Holy of Holiest":

> "We have a sale- and market department who follows up the customers. And they bring it [information from customers] into the Holy of Holiest"

And a developer in the same company described the development department as a "cage":

> "Scrum is agile development, but actually it is a rigid framework. That is, for us in the development department it is agile. We run our working day different than before. But for the rest of the organization it appears to be a kind of cage around the developers"

In company A a Product Owner said:

> "It [interruptions] occurs to a lesser degree. So I think we're talking about a virtual cage"

When it comes to communication, interviewees in all organization thought that communication between developers had improved after Scrum. But when it comes to communication to top management and communication to the product boards (where decisions on product strategy are taken) there were different opinions on how well it functioned. In company B developers complained that they no longer had any influence on product strategy

> "I think that Scrum has resulted in a situation where development department has too little influence. We don't have any influence at all; to decide which direction we think the products should take. Now it is the organization around us that decide….. We haven't been invited to the product board…. "

This lack of communication is also seen in company A. When a Scrum Master talked about product strategy he said that developers are now strongly focused on what to do during the next Sprint and that this makes them miss the long term view of the product:

> "We don't have any meetings where we look at things on long term. I do actually think that that has been lost, the possibility to…Or, I don't think we have become better to see on long-term after we started to use Scrum"

This strong emphasis on sprints is also seen in the research by Srinivasan and Lundquist (2009). In this study, however, they noted that the focus on the sprints obstructed organizational learning.

Both Company A and Company B had a support department that handled questions from customers. Before Scrum, staff from this department could contact developers directly in order to get help in answering questions from the customers. After adoption of Scrum they were not allowed to do this. In company A the interviewees said that they experienced that this was a problem for the staff at support. A developer in Company A said:

> "We, the developers don't notice much of it. But we do notice that the organization has problems. They don't feel they have access to developers. Support experience that it's much more difficult to get help now. Previously they could come and talk to the developers."

And a Scrum Master in the same company stated:

> "There have been some discussions around it [the shielding of the team]. It doesn't necessarily make the day any easier for the staff at support"

In Company B a developer said when talking about introduction of Scrum:

> "In principle there was a wall put up between support and the development department. We knew that there would be some…, well, challenges. Previously they [support] could directly into the development department."

In both organizations they have tried to solve this problem by reserving a pool of hours that are used to support work. In Company B a developer said:

> "We won't get rid of support. That is an unrealistic thought. So we reserve about 15% of the Sprint to support work."

But even though both organizations tried to solve these problems by reserving a percentage of the Sprint to support work, they still had some problems with either interruptions from support (but in a much lesser degree than before) or that staff at support experienced their working day as more problematic when they could not contact developers directly.

Conclusion

The purpose of this study was to increase the knowledge of introducing and adopting Scrum in software companies. In addition, we wanted to identify problematic issues that can be used to formulate further studies. Our study has shown several areas that need further investigation.

In our study all companies implemented Scrum on a large scale, without any profound evaluation and testing and all companies experienced the process of implementing Scrum as painful. It can be questioned if such an implementation is an optimal way to introduce an agile method into an organization. As Lindvall et al. (2004) suggest, an organization should investigate effects and side effects of a method before introducing it. The organization could then be able to adapt the method to their context before implementing it. Little research has been done on Scrum (Dybå &

Dingsøyr, 2008); therefore, indicating suitable contexts for Scrum is not possible. This suggests that a more reflective view of Scrum is necessary.

Introducing Scrum implies large changes in an organization's culture and work practices (Nerur, Mahapatra, & Mangalaraj, 2005); therefore, the organization must be well-prepared prior to implementing Scrum. In our research we found that top management was enthusiastic to Scrum and it seemed that they had a drive to introduce the new method. Even though key management had ownership, we found that implementing Scrum was a painful process, there were problems around self-organizing teams and Scrum principles were stretched. This shows that is not easy to change the culture and move to an agile method and that this change might be formidable.

In our research we found managers that were somewhat skeptical as to whether self-organizing groups were able to control and manage themselves. Some managers felt that, at times, they had to intervene and manage the work, because they did not trust the team to be in control. We cannot make any conclusion of why some managers had to intervene, but areas that can be examined are the level of agility of the organizations; that is, how agile are actually the companies? In traditional organizations managers practice command-and-control management, whereas in agile companies the managers practice leadership-collaboration and trust the individuals to exercise the right decisions (Cockburn & Highsmith, 2001; Nerur, Mahapatra, & Mangalaraj, 2005). Another aspect of these findings is the important people factor discussed by Cocburn and Highsmith (2001) and Boehm (2002). Agile methods place emphasis on people and individual competence is critical in agile teams (Cockburn & Highsmith, 2001). This view was also confirmed by managers in our study and a Product Owner said that: "If you don't have the right people, then Scrum is destructive". Nevertheless, why managers felt that they had to take control at times is an area ripe for additional research.

All companies expected productivity to increase when they started using Scrum. But in our research we found that all companies complained that they had not achieved the productivity they had expected. However, we were unable to discover the reason for this. What our research did show is that the "big bang" implementation of Scrum, used by all three companies, was a method that was not thoroughly evaluated and tested in the companies before introduction and implementation. Lindvall et al. (2004) state that a new method should be evaluated prior to implementation to see if it fits the company; it could be the case that such an evaluation could have improved integration and productivity in these companies. What is clear, however, is that productivity and efficiency when using Scrum should be further investigated.

In terms of how Scrum affected employees, the developers gained a more structured working day and their expectations for work became clearer; in addition, they worked less overtime and had better control of the work they carried out in a Sprint. Some developers felt, however, that after Scrum was introduced, they no longer could influence future development of the product. They also felt that their opinions were no longer important and that strategies on further development of the product were lacking.

The Scrum Master's role was less clear. In Company B it seemed that this role was important and prominent; however, in Company A, the Scrum Master was more withdrawn. More research should be conducted on the role of the Scrum Master.

In our study we found that some managers and developers felt that Scrum was agile for developers, while in other areas they felt that Scrum was rigid. Why Scrum is perceived as rigid needs further investigation.

In all companies we found aspects of Scrum that were adapted. Future research should look into the aspects and elements of Scrum that are adapted by the companies to optimize Scrum.

Quality assurance seen in relation to Scrum is another area that should be researched. In our case study we found that when quality was discussed the companies equated quality with testing within a Sprint. Modern quality thinking suggests a holistic thinking in the organizations when quality is pursued (Beckford, 2002). An area that should be researched is how quality is pursued in companies where Scrum is used and how Scrum can fit into modern thinking on quality.

References

Bazely, P. (2007). *Qualitative data analysis with NVivo*: SAGE Publications Ltd.

Beck, K. (1999). *Extreme programming explained: Embrace change*. Addison-Wesley Professional.

Beckford, J. (2002). *Quality*. Routledge Taylor & Francis Group.

Boehm, B. (2002). Get ready for agile methods with care. *Computer*, 64-69.

Charmaz, K. (2000). Grounded theory objectivist and constructivist methods. In *Handbook of qualitative research* (2nd ed.). Thousands Oaks, CA: Sage Publications.

Cockburn, A. (2007). *Agile software development*. Pearson Education, Inc.

Cockburn, A., & Highsmith, J. (2001). Agile Software Development: The people factor. *Computer*, 131-133.

Cohn, M., & Ford, D. (2003). Introducing an agile process to an organization [software development]. *Computer, 36*(6), 74-78.

Coleman, G., & O'Connor, R. (2007). Using grounded theory to understand software process improvement: A study of Irish software product companies. *Information and Software Technology., 49*, 654-667.

Danube. (2009). Has Scrum become the face of agile? Retrieved November 15, 2009 from http://scrummethodology.com/has-scrum-become-the-face-of-agile/

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology, 50*, 833-859.

Glaser, B. G., & Strauss, A. L. (1999). The discovery of grounded theory: Strategies for qualitative research. New Brunswick (USA) and London (UK): Aldine Transaction.

Lindvall, M., Muthig, D., Dagnino, A., Wallin, C., Stupperich, M., Kiefer, D., et al. (2004). Agile software development in large organizations. *IEEE Computer, 37*(12), 26-34.

Mann, C., & Maurer, F. (2005). A case study on the impact of Scrum on overtime and customer satisfaction. Paper presented at the *Proceedings of the Agile Development Conference (ADC'05).*

Marchenco, A., & Abrahamsson, P. (2008). Scrum in a multiproject environment: An ethnographically-inspired case study in the adoption challenges Paper presented at the *Agile Development Conference, AGILE 2008*.

Martin, R. C., & Martin, M. (2007). *Agile principles, patterns, and practices in C#.* Pearson Education, Inc.

Maurer, F., & Melnik, G. (2006). Agile methods: Moving towards the mainstream of the software industry. Paper presented at the *ICSE'06, Shanghai, China, May 20–28*.

Moe, N. B., Dingsøyr, T., & Dybå, T. (2009). Overcoming barriers to self-management in software teams. *IEEE Software*, 20-26.

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM, 48*(5), 72-78.

Schwaber, K., & Beedle, M. (2001). *Agile software development with Scrum*. Upper Saddle River, NJ, USA: Prentice Hall PTR.

Sommerville, I. (2007). *Software engineering* (8th ed.): Pearson Education Limited.

Srinivasan, J., & Lundquist, K. (2009). Using agile methods in software product development: A case study. Paper presented at the *Sixth International Conference on Information Technology: New Generations*.

Szalvay, V. (2004). *An introduction to agile software development*. Retrieved November, 2009 from http//www.danube.com/docs/Intro_to_Agile.pdf

Tichy, L., & Bascom, T. (2008). *The business end of IT project failure*.

Biographies

Elin Brekkan is an assistant professor at Bodø Graduate School of Business. Her research interests are: system development processes and methods and agile methods. She has previously worked in the IT-industry for several years, developing computer systems for the telecommunications industry. The last four years she has been teaching and doing research within computer science.

Eystein Mathisen is an assistant professor at Bodø Graduate School of Business. He has been teaching and doing research within computer science. Her research interests are: system development processes and methods and agile methods. In addition he has some years experience from the IT-industry, developing computer systems for the telecommunications industry.