NTNU – Trondheim
Norwegian University of
Science and Technology

Agile Security Requirements

A master study into their application

Anders Nordli Knudsen

Master of Science in Computer Science
Submission date: June 2014
Supervisor: Guttorm Sindre, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Abstract

Agile is the contemporary development practice of choice but security has been claimed as a challenge for it. This thesis investigates whether agile methods can be used for security-critical software and if the reason why the majority of Norwegian companies deviate from the agile methodology in their development is linked to security, by looking at the security requirements. A questionnaire and interviews of Norwegian companies were undertaken, and while the questionnaire did not yield any results the data from the interview indicate that the reasons for not conforming to the methodology appear to be related to security work and assurance. Agile is implied by the limited sample size to not only be useable for security-critical software but may be the best option in projects with uncertainty to the system and changing security requirements.

Sammendrag

Smidige utviklingsmetoder har blitt standard praksis for programvareutvikling i dag, men har møtt anklagelser for å ikke lage sikre programmer. Denne masteroppgaven tar for seg spørsmålet om smidig utvikling kan brukes for å utvikle sikkerhetskritisk programvare og om grunnen til at mesteparten av norske bedrifter gjør avvik fra de smidige metodene er på grunn av sikkerhet, ved å se på behandlingen av sikkerhetskrav. For å undersøke dette ble det gjort en spørreundersøkelse og intervjuer av norske bedrifter. Selv om det ikke kom noen resultater gjennom spørreundersøkelsen, kom det fram av intervjuene at avvik var grunnet sikkerhet og da spesielt måtene det ble forsikret om at sikkerhetskravene var gjennomført. Smidig utvikling er implisert av det begrensete utvalget å kunne brukes i utvikling av sikkerhetskritisk programvare, spesielt der det er stor usikkerhet til hvordan systemet skal være eller sikkerhetskravene endres fortløpende.

Acknowledgements

I would like to thank first and foremost my supervisor, Guttorm Sindre, for tireless patience, support and guidance throughout the work with the master thesis.

IKT-Norge for their help in sending out the questionnaire. While it did not go the way I had hoped, I am grateful nonetheless for the assistance.

The five interviewees at the four companies for letting me come and ask questions about their security measures and work, for not only trusting enough to speak of security but also welcoming the chance. A thank you goes especially to the first interviewed for stepping up and helping me reach out to other companies. It was a much needed reassurance after the results of the questionnaire.

Table of Contents

Section 1: Introduction

Iterative development has been known since long before agile was coined as a term in software development through the creation of the agile manifesto by Beck et al (2001). Today it is widely recognized as the thing to do and many developers embrace the methodology as it is something the customer expects. There are some challenges inherited to the methodologies in agile, among them security. A majority of the established security assurance methods fall outside agile thinking (Beznosov and Kruchten 2004). Critique has also been made that agile does not make secure code due to a lack of security expertise on the field (Sullivan 2010) and agile methods are seen as inadequate by theoretical analyses (Bartsch 2011). After the revelations of Edward Snowden and the more recent Heartbleed vulnerability in OpenSSL (Heartbleed), security has received worldwide attention. This study sets out to see if there has been any changes and if agile is now more fit for security-heavy projects, taking advantage of the increased awareness surrounding the topic.

1.1 Background

In a pre-project study done ahead of this research (Knudsen 2013), investigation was done as to whether security meshed well with agile development methods. Through state-of-the-art research in published scientific studies and a case study at a Norwegian company, the pre-project study sought to answer three research questions:

- Could agile development methods be used for security-critical software?
- How did the use of agile development method affect the security requirements?
- How was agile development done with security-heavy projects in practice?

While enough information was gathered to form an answer to the second research question, research into agile development for security in practice led to a discovery that made answering the first impossible in the given timeframe. Of the three practical examples studied, only one stayed true to the key principles and methods of the chosen agile development method and only did so through thorough reworking of their security development approach (O'Boyle and Eng 2013). Both the NASA development team (Trimble and Webster 2012) and the team in the case study had their own variation of an agile development method fitted their needs. Furthermore a published study of Norwegian companies' use of agile development methods came out towards the end of the pre-project study. The published study had 81% of the 552 companies responding saying they did use an agile development method, but the majority of these respondents also deviated from the method's key values and principles (Brevik and Grønli 2013). Following the assumption that at least some of these would be working on security-critical software, any answer for the first research question without further research into the reasons for

deviating from agile principles would be flawed.

From a paper by Beznosov and Kruchten (2004) conventional security assurance methods could be seen divided into four groups of methods, two of them which were either innate in agile or independent of any development methods. The third group they wrote could be feasibly covered through automated tools. Methods from the fourth group, consisting of half of all the security assurance methods, were deemed incompatible for agile development methods. In one of the two proposed strategies for these mismatching assurance methods, they suggested developing new agile-friendly security assurance methods but could not offer any of their own. This was ten years ago.

1.2 Study focus and Research Questions

This study will focus on getting a more definitive answer through investigating the reasons of deviation in Norwegian companies, with a focus on the security requirements and assurance of these in agile development.

Research questions for this study are:

RQ1: How is security affected in deviations of agile development? Are the reasons for nonconformity security related?

RQ2: Can agile development methods be used for security-critical software?

The rest of the document is structured as follows: Section 2 explains the research methods used in the study. Section 3 holds the results gathered through the research methods. Section 4 provides discussion of the results. Section 5 is the conclusions from the study.

Section 2: Research Methods

2.1 Choice of Strategy and Methods

In selecting the research strategies for answering the research questions, the six strategies detailed in *Researching Information Systems and Computing* (Oates 2006 p. 35) were studied. RQ1 focuses on how the security requirements are affected when developed with a deviating agile method. It also seeks to discover if the reason for the deviation is security related. As such, survey is an eligible strategy because of its use to form generalized conclusions about how security is affected (Oates 2006 p.104). On the other hand, the depth to answer if the reasons are security related may be lacking, as the focus is on getting a larger coverage over going into depth about the researched topic (Oates 2006 p.105). It also only provides answers to how it is right now and requires the replier to both know enough of the topic to answer and being honest when answering the survey. The design and creation strategy is unsuitable for the first research question since it is not about creating a new artefact. An experimentation strategy is not viable as the research question is no hypothesis that is to be proven or disproved. If it was, it would have required a test group to run through a project. While agile methods lead to shorter development cycles it would take at least a month to run through an iteration of the project. As experiments should be repeated to be certain that outside factors do not affect the result (Oates 2006 p.127-128), an experimentation strategy is unfeasible due to time limitations, lack of participant group and the difficulty of removing outside factors on a method that is based on continuous improvement both of the project and its members. Case studies focus on the case under investigation in its real world application, focusing on the depth instead of breadth and exploring all the factors of the case (Oates 2006 p.141-142). For RQ1 it is a viable strategy as the question seeks answers to the reasons behind the method deviation and how security is affected. However, the answers coming from such a strategy cannot be used to make a generalization for other environments and have less grounding to answer the second part of the question. It may also be hard to find a project where it is known that the developers use a deviation of an agile method. For action research to be a good strategy, the research question should be focused on improving an already existing method. Ethnography deals with understanding cultures or groups of people (Oates 2006 p. 35), which is unsuitable for RQ1.

RQ2 seeks an answer to whether agile development methods can be used for security-critical software. A yes or no answer is desired, so the generalization possibility of a survey is viable but comes with the cost of possibly overseeing the reasons to the answer (Oates 2006 p.104). Like the first research question, both the design and creation strategy as well as the experimentation strategy are unsuitable. Design and creation because the thesis is not about the construction of an artefact and experimentation due to time limitation, lack of participation group and removing outside factors in the experiment. A case study strategy is advantageous for this research question due to appropriateness for situations where the researcher has little control and exploration of alternative meanings and explanations, according to Oates (2006 p.150). He also makes note that it is sometimes perceived to be leading towards

generalization with poor credibility, time-consuming to set up and the presence of the researcher may affect the results (Oates 2006 p.150 - 151). In opposition to the survey strategy, a case study will give a lot of depth but lack the width to form a strong generalized answer to the research question. Action research is unsuitable because of its focus on improvement of a method in practice, as described in the advantages of such a research by Oates (2006 p.168). There is no research group for this study and conducting it by the researcher alone, while being agile, can lead to self-delusion as warned by Oates (2006 p.160). The result of such a study strategy is also questionable, as some IT researchers do not accept it as a strategy or criticize it for lack of rigor and inability to establish cause and effect outcomes that can be generalized (Oates 2006 p.168). Likewise is ethnography not a good strategy, as it focuses on people and the research question is about a method.

Considering the strategy advantages and disadvantages, there are two viable strategies for both research questions. Survey and case study. The time allotted to this thesis is twenty weeks so the use of two strategies would be too time consuming, especially considering case studies' nature of providing a lot of information about specific instances that must be processed and time used for setting it all up. RQ1 needs a grounded answer that a case study is useful for, whereas RQ2 seeks a generalized statement that was also an unanswered part of the pre-project study. As the pre-project study was a case study and out of a personal desire from the researcher to contribute more data for the research field, it was decided to use a survey strategy for this master's thesis on both research questions.

Four different data generation methods are mentioned by Oates (2006 p.36) to produce empirical data. Interview, observation, questionnaire and documents. The data produced is either qualitative or quantitative. It has already been mentioned that a grounded answer is called for in RQ1 without the specific need for a wide generalization. As such qualitative data is sought. Interviews are described by Oates (2006 p. 187) to be suitable data generation methods when detailed information is sought, the questions are complex or open-ended or the issues investigated are sensitive or hard to observe. Matters of security are a sensitive topic that many may not wish to part with openly, so an interview is a viable data generation method. The other side of it is not many may wish to be interviewed or reveal too much about their own practices. Likewise does observation face some of the same issues as not only is the researcher there to learn how security is done, they are also watching as the work being conducted or discussed. Depending on the type of observation used, some disadvantages may lie in the restrictions of studying behavior without getting to know the reasons or lack of reliability due to the research depending highly upon the researcher conducting the observation (Oates 2006 p. 214-215). On the other side, it is described to provide a rich insight into the social setting, and what people really do (Oates 2006 p. 214). While easy to set up, it may be difficult to get the reasoning behind the decisions carried out in practice with observations alone. In RQ1's case, trying to extract information about how security is affected in an observation alone may be difficult and more so whether the reasons for method deviation is security related without asking for it specifically. Questionnaires are best suited to situations where the researcher wants to obtain data from a large number of people, get brief and uncontroversial information and have the data standardized (Oates 2006 p. 220). It is also economical in that it costs very little in materials and time compared to other data generation methods (Oates 2006 p. 229). Some of the

disadvantages lie in frustration that predefined answers do not match the responder's experience, the answers may be biased in the researcher's view or the responder may not answer truthfully (Oates 2006 p. 230). A response rate of thirty percent would be considered good and ten percent not unexpected for questionnaires (Oates 2006 p. 99), so getting a large enough sampling frame is important. For RQ1 the responses from a questionnaire may be too shallow to provide a good answer without relying on open questions, which there should not be too many of in a questionnaire because they take longer to answer and increases the chance that the responder will not complete it. Questionnaires will however give a strong indication to whether nonconformity is security-related, the second part of RQ1. Documents as a data generation method have the advantages of being cheap, easy to obtain and work based on them can be checked by other researchers as most of them remain permanently available (Oates 2006 p. 241). More sensitive and confidential documents are harder to be permitted access to, and may be protected by law from viewing (Oates 2006 p. 236). How security is handled in a company may be too sensitive for a researcher to gain access to through documents, making other data generation methods such as interviews be less intrusive and more readily agreeable for the respondents. Research articles based on agile and security requirements is thus a better approach in the data generation method but from experience in the pre-project study, the actual number of good articles here including both security and agile development is in the minority.

In RQ2's case, the data collected must be viable to form a definite yes or no with enough support to form a reason to the answer. While interviews can provide the depth and reasoning, it has the drawback of needing a lot of time and effort that usually make them unsuitable for drawing generalized conclusions, since it would a large number of interviews to get a big enough sample size (Oates 2006 p. 199). Observations fall in the same boat where the number required and only watching behavior and actions will not provide either reasoning or answer to the research question. A questionnaire on the other hand would be a very viable data generation method as long as the questions can form the reason for whether agile methodology can be used for security-critical applications, with reasoning beyond only a statistical number or the idea that if the majority does it then it must be right. With a large enough sample size, it can be used to make a generalized conclusion. The sample size for documents is, as has been mentioned, too low to be of good use for RQ2. Research articles may help contribute to the answer but is not enough to form data on its own.

Considering that neither observation nor documents contribute to the research questions the same way as the other two, the choice of data generation method falls on interview or questionnaire. Due to the number required for a good sample size and the warning that interviews are time-consuming for the researcher (Oates 2006 p. 198), questionnaire was chosen as the main data collection method since a part of RQ1 and the whole of RQ2 ask for a generalized answer. The need for more than just quantitative data, especially in RQ1, means that a part of the questionnaire must contain open questions so the respondents can answer with their own thoughts and experiences. Because questionnaires have the advantage of being low cost in materials and time (Oates 2006 p. 229), interviews were added as a second data generation method to produce more data on the field and increase the research quality through method triangulation (Oates 2006 p. 37). Tertiary the findings of the pre-project study and other

research articles will be used to compare with the results of the other methods, but will not be among the data generation methods of this thesis.

2.2 Conducting the Survey

To answer the research questions the methods used must be able to generate data on the following:

- If the respondents use agile development methodology in projects with high security requirements (RQ2).
- Why they use/do not use agile methods in high security projects (RQ2).
- If any changes has been made to the agile methods used and what they are. Especially if they are related to security (RQ1, RQ2).
- How work with security requirements change based on the security level of the project (RQ1, RQ2).

Indirectly related to the topic, information about what agile methodology was used and how they assured the security requirements had been met were also sought.

It has already been established that the data generation methods of questionnaire and interviews will be used for this thesis. How they were used to generate data is written in the two subsections below.

2.2.1 Conducting the Questionnaire

Some warnings regarding questionnaires are getting only ten percent response rate on a questionnaire is not uncommon (Oates 2006 p. 99),that unsolicited questionnaires are regarded as junk mail by many people (Oates 2006 p. 102) and spam filters may prevent bulk emails from even reaching the recipients (Oates 2006 p. 102). As such an interest group for the Norwegian information and communication industry, IKT-Norge, was approached with the plea for assistance. They were willing to send out the questionnaire alongside their organization email, provided it was not too long and sensitive and passed a review of their own. It was believed that through them, the response rate would be significantly higher and reach a larger number of people than had the researcher tried to construct an own email list. While the sample size of possible respondents is unknown, it should be possible through this probability sampling technique of Norwegian ICT companies to get a sample of respondents indicative of the overall population (Oates 2006 p. 96) and so enable generalization for the research questions.

The questionnaire was built through online survey software (SurveyMonkey). Because of the limitations of ten questions and a hundred respondents in their free model, a monthly subscription was purchased

to grant unlimited questions and up to a thousand and more respondents. Considering Brevik and Grønli(2013) had over five hundred respondents to their own research, it was felt as a liability to put the limit to only a hundred even if the chance for so many answers could be low. It was considered to additionally put the questionnaire up online on an international and a Norwegian forum to generate more data. It was elected not because not only would this add to the cost of running the questionnaire by possibly going past the monthly subscription answers, this form of non-probabilistic sampling would only provide a weak basis for generalization (Oates 2006 p.96). It would also cast doubt to the reliability of the answers as it is unknown how much experience the respondents on the forums have with the topics of the thesis. There is also the possibility that the forum respondent also had answered the questionnaire sent through IKT-Norge.

The first two questions of the questionnaire were made to expedite completion for those that that did not use an agile methodology in development. To pick up whether it was related to security, the recipients were asked to explain why they were not using it. No mention of security was made here to avoid leading on a reason tied to it. The next questions were tied to determining whether the respondents were using a deviation of agile security methodology and the rest of the questionnaire were Likert scales (Oates 2006 p. 223-224) on differences between security requirements and other system requirements. A textbox was left for explanation to the differences if the respondents wished to express it further. The Likert scales revolved around the agile principles of welcoming change, delivery of working software, process improvement and customer collaboration; as these principles were considered the most important for the thesis out of the twelve principles mentioned on the agile manifesto page (Beck et al. 2001).

Prior to the questionnaire being sent out, the questions were worked out in conjunction with the master thesis supervisor. The questionnaire was sent first to four colleagues as a pilot test, with three respondents and two having suggestions for improvements. After finalizing the form, the link of it was sent to IKT-Norge for approval and subsequently being sent in their email letter.

The email address of the researcher was added to the questionnaire if the recipients had any questions or volunteered for an interview to speak on the subject. It was explicitly stated that responses would be anonymous and even if the companies were to contact the researcher, there would be no way to tie the email to any specific response.

See appendix 1 for the questionnaire.

2.2.2 Conducting the Interviews

The questions made for the interviews were derived from the questions used in the questionnaire but ordered differently in different themes. First were questions about the choice and use of agile methodology, then the interviewees were asked to relate their experiences comparing security requirements or high security projects with more ordinary ones. Last security assurance related to

security requirements was touched upon before an open question at the end. Most of the questions were also open-ended to require more than a yes or no answer.

The interviews were structured in a semi-structured fashion, allowing the themes and questions that were identical for all interviews but also giving the possibility of asking questions about issues raised by the interviewee (Oates 2006 p. 188). Oates (2006 p. 188) also warns that such interviews are not useful for research conclusions that draw conclusions for the population, because the same topics may not be addressed or the number of cases are few. Primarily the interviews are intended to provide an in-depth answer to RQ1 while contributing to the conclusions. There is a chance that none of the interviewees use a deviation of an agile methodology for their development projects but as the majority of Norwegian companies were determined to do such a deviation by Brevik and Grønli (2013) when agile methodology was in use and that half the security assurance methods fell outside the agile methodology (Beznosov and Kruchten 2004), the researcher is confident at least one case will provide answers.

Due to time and cost of travel as well as money already being spent on the questionnaire subscription, local IT companies in the Trondheim region were approached for interviews. As the interviews were intended to answer in-depth instead of forming generalization for the research question, the non-probabilistic approach of purposive sampling (Oates 2006 p. 98) was taken where the researcher picked which companies to approach. As an unforeseen advantage, one of the interviewees took it upon themselves to suggest and approach more companies on behalf of the researcher for interview opportunities, creating a light snowball effect to the sampling (Oates 2006 p. 98) and bolstering confidence after the results of the questionnaire. Because the researcher waited to see if the recipients of the questionnaire would want to invite for interviews, the rest were done later in the research period and so only a total of four interviews were had that could be arranged through time and interest from the companies approached.

An audio recorder was used for all four interviews. All interviewees gave express permission for the researcher to use the device when asked at the beginning of the interview. They were all also informed that the audio recording would be deleted at the hand-in of the master thesis.

Three out of four interviewees were informed that the findings would be kept anonymous and they had the right not to answer any questions they did not wish to. The missing was a mistake made by the researcher in conducting the interview, but anonymity is still kept and none were pressured into providing an answer.

The interviews were carried out at the interviewee's work place, at an office room where the interviewer and interviewee sat at opposite sides of the table. Such a seating is not ideal as it would suggest confrontation (Oates 2006 p. 191) but it fell naturally in place at each interview. Only the fourth had a natural seating that allowed a more ninety-degree angle between interviewer and interviewee. Introduction and brief small talk were done at the beginning of each to ease any tension (Oates 2006 p. 191) and an open body-language was kept throughout the interview by the interviewer (arms at the sides of the chair and never crossed, back straight, legs apart). Active listening through eye-contact and

mirroring was also done to set the interviewee at ease and questions were kept in a neutral tone not to lead the interviewees on, as recommended by Oates (2006 p. 192-193).

The interviews were ended with an open question asking the interviewee to raise any points not addressed and thanking them for their time as advised by Oates (2006 p. 193).

See appendix 2 for the interview questions translated from Norwegian. Follow-up questions as a part of the semi-structured interview have not been added there.

Section 3: Results

3.1 Results from Questionnaire

After three weeks without any responses to the questionnaire or any emails regarding the same, an email was sent IKT-Norge politely asking if the questionnaire had been sent. Someone delegated to add it to their email letter had forgotten, for which they apologized deeply and promised to send it as soon as possible.

Once two more weeks had passed without a reply to the questionnaire, another email was sent asking the same. This time they informed that the questionnaire had been sent out the same day they discovered they had forgotten. The questionnaire had been in circulation since the 23rd of April.

Unfortunately, even now at the end of the research, not a single reply whether by response to the questionnaire or email to the researcher has been received as a result of this data generation method. The only brief and unhelpful response to it was made by the researcher, testing that the questionnaire was indeed operational prior to sending an email to IKT-Norge the first time around.

3.2 Interviews

A total of four interviews were done at different organizations within Trondheim. The data collected from the interviews have been separated into four themes. Below is a brief overview of what the role and organization of the different interviewees.

Interview #	Role	Organization
1	Test-developer	Large company
2	Manager	Consultancy firm
3	Manager Developer	Consultancy firm
4	Security Expert	Large Company

3.2.1 Agile Method

The first interviewee worked as a tester and developer at a larger company in Norway. In their project they used the standard Scrum with two weeks' sprints consisting of eight days of work, one day for demo and one day of planning the next sprint. They also held daily scrum meetings (standups) and had designated roles within the project team such as product owner and scrum master. The members of the project team were split into three different roles: testing, development and public relations/architecture/project leader. Same methodology was used for projects with high security requirements, with the mention that all of their projects have high security requirements and the current one higher than ordinary. The high demand came as a result of moving sensitive data from an internal secure network into the cloud and back again, while making an index in the cloud for the information.

While having only been working there since the beginning of the year, the interviewee believed that they always use Scrum without thinking about changing or experimenting with other agile methods. In mention of security requirements, the interviewee said:

> ***"Use it [Scrum] always. Whether you have security requirements or not, we have not considered to change methodology for that reason."***

At previous workplaces, the interviewee mentioned they had used Kanban and Scrum and earlier still had begun using agile towards the end after the waterfall methodology. For the current company it was believed to be no contemplation for anything else than Scrum and that you had to use an agile methodology because that is how the world is now.

> ***"I feel it is more fun at Scrum. That you get out results."***

Interviewee did not believe there had been any changes made to the methodology to fit it to development but rather the other way around. Development had been fit to the methodology itself to allow for shorter iterations. While the method was not believed to be an advantage related to security, with Scrum there was the benefit that you could always see that you are making what you want to make. You know all the time where you are in the development process. For them the methodology affected how they worked with security, in that they added security gradually over the course of the project.

The major challenge with the agile method was said to be getting the customer involved in the project. Usually you would want highly competent customers involved in the project but if they were highly competent, they usually did not have time to be involved. An example for such customers where you could expect difficulties in getting help was the government or other officials. For their current project

they had an internal customer, a project manager from the company itself. That was also thought to be a possible weakness for their method. Due to the short iterations and to have a demo to show towards the end, there was a risk that they would not get the security requirements fulfilled on what they made because they had to make insecure versions to show off.

Something that could go outside their method was a lot of spec work having to be done at the beginning of the project, where they had to make plans for security for longer durations than a single sprint. Because of the high security demands, they had to take the security of the whole project into consideration.

In interviewee #2's division in Trondheim they used agile development for the majority of their projects. Agile development was defined as projects wherein there was no set scope, time or budget; but continuous assessment and an iterative development with something to ship at the end. Within those boundaries, it was said that the majority of what they did was agile but some projects had more locked frames than others. While the different projects used different methods, most of their developers had a basic education in Scrum and had the methodology as a starting point for adaptation to the different projects depending on the project's requirements. Their project teams ranged between two or three up to seven or eight people, with some projects distributed to other divisions while others were done in-house. There was also a difference in the amount of customer collaboration in the different projects. All the variations posed different requirements of the methodology but most started with Scrum wherein there is a backlog with prioritized tasks taken from the top for development to be done within the week. The sprints were ranging in length from one week up to four. They did not follow Scrum by the book but ended up with a mix between Scrum and Kanban adapted to each project.

They used to follow Scrum more by the book back in 2010 when working on a project with high security requirements. To accomplish it they added Microsoft Security Development Lifecycle for Agile Development (Sullivan 2010) as a component and starting point, especially the first year the project was established. Because of the high focus on security, one member of the four developing the project had it as a special responsibility. One of the activities mentioned that was especially done was threat modeling of the user stories, where they for each had to consider what to keep in mind on matters of security. Because security was a very important feature for the customer, it was not hard to get it prioritized in the backlog for the project.

> ***"Often you experience that non-functional requirements get down-prioritized. Things like performance, security, robustness. Less mature customers or customers who doesn't have a focus on that, have a tendency to like all the time want a new functionality and then it can be difficult for us as producer to balance and say that now maybe we must prioritize things you may not see at once, but must be there."***

In their project with high security requirements the customer was so focused on security that the team did not need to sell it in, and instead could work on it from day one. The security work was adapted to their method. Another security measure they do in their projects is a security review by an internal and professional group focused on security. They spend a day on the review going through the project and looking for known vulnerabilities. On the high security project an external review was also done by an expert company, which expressed that the reviewers could not find much to mention. To accomplish this level of security they had followed the guidelines in Microsoft Security Development Lifecycle, had a member on the team focused especially on the security and had a customer who knew well the importance of security. Due to the high focus, the development team was always attentive in learning the security mechanisms in the frameworks they utilized. For instance they used a framework from Microsoft called MVC where most standard security mechanisms are default. You really had to try hard to not get escaped input, and cross-site-request-forgery was easy to deny. Things that used to be a hassle could be removed by the use of the framework, so they paid attention to what security functionality was offered by it.

The reason for choosing Scrum in their project was due to security being one of the non-functional requirements, the other requirements were equally important and to better be able to deliver the project when the specification of it was unclear. While the customer had an idea on how the system should be, there was much uncertainty related to its functionality. Were they to use a more traditional waterfall method the interviewee said:

> ***"We would not be able to design the whole solution ahead of time and then one considered the amount of security. When that is said we did do comparably much design on the security part early on precisely because it is a non-functional requirement. It is possible to think it a little bit through and build it without considering too much functionality."***

Interviewee mentioned how to do logging, how to make it anonymous, how to encrypt for databases and how to do authentication as examples of functionality they considered early on. This was all established from the beginning without the customer prioritizing it up or down. The project became agile because even if the security part was important, the other functionalities were equally uncertain so they could not design the system ahead of time.

The big difference for the method now is that since the project has moved from development to management and further development, they no longer spend time threat modeling everything. As a security review done externally resulted in everything looking good, they now focus on the baseline established so as long as everything new is established within the concepts set and do not introduce anything new, then security is kept within the solution. The established and reviewed parts are not touched. Any new functionality follows the same patterns as previous methods to mitigate OWASP threats. Because the development team is down to one man, there are no dedicated resources put for security and no Scrum standups. After the delivery of the project, it is more a matter of getting feedback

from the end-users and to see if there are any bugs to be fixed or new features desired added. So the big difference came to whether the project was in development or in management.

At the question whether there had been any challenges for projects of high security because of the agile development, the interviewee replied:

> ***"No, do not think that was the difficulty. What was more a challenge was maybe the lack of knowledge on methods for security because that was something the customer asked for. They wanted to know like how are you as producers going to have a work method that ensures we make a secure solution."***

Since this was something the project team had little knowledge of beyond the technical aspects, they had to learn how to add it to their development method.

The advantage in using agile methodology for the project was that there was no clear order on what was to be made. Nobody had a good overview of it all because of a large and complicated rule set behind the system to be developed. The project was founded upon the freedom not to have to know how everything fit together and having to estimate it all in advance, with a given timeframe and testing at the end. Instead the project demanded a lot of testing throughout development, as laws and regulations affected functionality and the interpretation of how it should function, which could be different between customer and developers. They would not have been able to work that way with a waterfall method, as the numerous bugs discovered would require them to go a long way back in the development method.

The third organization and second consultancy firm also used Scrum in nearly all their projects, in different shapes. While they did use the same in projects with higher security requirements, they rarely worked on projects with such a high demand for security. It did feature as a part of most projects and as an example given they found it relevant to secure web applications against ordinary OWASP threats. As such security is an element in what they do but rarely high on the project requirement list, as the customer is usually more focused on functionality. They use Scrum due to its adaptability in development, allowing the customer to change their mind.

> ***"It is so that agile development is seen as more efficient than traditional organization forms in a development project."***

They use different variations of agile development with Kanban being discussed. For them the question revolves around whether to use sprints or not, if it is important to have them or if you can work continuously instead. While they have adapted Scrum for each project, they have not made changes to the method. A challenge they have seen is that it is not easy to keep to locked sprints, it possibly being better to use continuous scrums instead of sprint iterations to add quick changes.

Challenges coming from the agile methodology for high security projects have never been an issue and they have not experienced the security requirements having been treated differently than others. The challenge lay in being allowed time to work on a non-functional requirement that is not so easy to see, as security is often one of the non-functional requirement similar to performance, availability and user-friendliness. It is a matter of getting time to work on it during the sprint, as often you work with functionality instead. The point is to keep within the absolute limits of security; you cannot let a security requirement remain in the backlog unimplemented.

The fourth company interviewed also made use of Scrum but the interviewee said that you never follow an agile standard to the letter. They worked on a large project with longer iterations, set up for agile methodology with user stories, prioritization of the work to be done and implementation of what is on the top of the list. The most important guidelines for architecture, design and solution was made from the beginning, and then agile development was done iteratively; with definition and prioritization of user-stories, implementation and testing of them. The most important elements were made first then the rest was built.

The only change for the methodology used was some adjustments to the daily meetings. They also used the same method on projects with higher security requirements.

For the security expert interviewed, it was important to set in security from the very beginning. Starting in the design phase of the project and ensuring the security requirements are in place from early development. It was also important that risk assessment had been done and there was good knowledge of security requirement mechanisms among the developers. But the implementation of security functionality and to implement software securely was prioritized in the same manner as the rest of the requirements.

> ***"That we should have a signed solution is prioritized in the product line on the same lines as other functional aspects of the code. So security functionality goes in the backlog and is prioritized equally to other requirements, but with the higher priorities on the top."***

It was said that you must have the security requirements in place and the developers must know what the code they are making should do and how critical for security the component made is in relation to others. Beyond that the same method is used. Security-requirements, -functionality and –testing are put into the product line alongside other tasks, even if they are done externally.

The role of the security expert for this project had been as an ad-hoc resource called upon when important security decisions had to be made. The interviewee had not been a part of the sprint planning meetings or set the security requirements early in the project, as they had only joined the project half a year ago.

When it came to challenges for the method, the interviewee had to answer in a more general manner:

> ***"And the challenge could be… it is always difficult in the interface, both between project team and between modules in the project. So that is the most difficult part from a security perspective. To get the interface to work. One knows well what happen in one's own module or component, but how it works together with other components is always what is difficult."***

To see the whole on an information security basis and how the information is to be handled from A to B in an information system was what was difficult. While not something each developer must have an overview of, someone on the project team must know it.

The biggest advantage of Scrum has been being able to plan to a certain level then letting it remain to the team, the priority mechanism in the project and communication with the customer to ensure the customer gets what they want. In earlier projects with the waterfall method the organizations hired to implement the system would get too stuck in what was planned and written in the requirements that it ended up with a system they did not really want. Another advantage lies in a better day for the developers because they have a schedule to keep. Knowing what they have committed themselves to do each month and the goals of the project. For the company it was also an internal competition between development teams on which manages to get done the most with good quality. A last advantage mentioned was getting a continuous motivation to get the work done and more awareness of what each team had developed through the status meetings.

On the security side the interviewee said:

> ***"That I will be careful about argumenting for or against. I cannot say if it has become better or worse by using an agile methodology from a security perspective. What I see from before my time is that a lot of good security measures have been done both on the module level related to crypto solutions and mechanisms, and in relation to for instance code review from a security perspective. That type of things. But then the question is if this had been done if one did not use an agile method, that I cannot say. So whether agile has led to more secure or less secure code is impossible to answer."***

In summary:

	Interview #1	Interview #2	Interview #3	Interview #4
Uses agile method?	Yes (Scrum)	Yes (Scrum)	Yes (Scrum)	Yes (Scrum)

Uses same method in high security projects?	Yes	Yes	Yes (does not have high security projects but would use same)	Yes
Reason:	How the world is.	Uncertain/changing requirements.	Adaptability, giving the customer what they want.	*See advantages.*
Method been changed?	No (but long security spec work outside)	Yes (method mix)	No.	Yes (daily meetings)
Challenges for the method?	Customer collaboration. Internal customer. Insecure demoes.	Lack of good security methods.	Time to work on non-functional requirements. Keeping within sprints.	Overview and interface between components.
Advantages from the method?	Project development awareness. Create what is wanted.	Adaptability.	None mentioned.	Customer gets what they want. Awareness and motivation for developer.

3.2.2 Differences on Agile principles

Interview #1

Customer collaboration

They have an internal customer that sets the requirements high for security, on the level of nobody gaining access to someone else's documents, not being able to fetch something you do not have access to, all communication being encrypted, only real user accounts allowed so it is known who it is, no logging of data in the system and other logs should not contain sensitive information. On the developer side it is another detail level, where it is decided how strong cipher to use and who to keep the access keys. The customer sets four or five security requirements but they are usually broad. There are also internal requirements from the company itself that has been heightened over the year, on matters of what to encrypt and where to encrypt it. The interviewee believed this had increased after Snowden.

> ***"[I] Believe that maybe internal requirements [for security] often are stricter than external requirements."***

On the detail level they have plenty and specific strict requirements from the company itself.

Openness to change in requirements

The strict high-level security requirements are locked and not something they mess around with. Functional requirements when it comes to that the project should be able to scale is not changed. They could probably change the details about the security requirements but the stricter ones have been reviewed so much by the company itself that it would take a lot before they dare change them.

> ***"If we do not fulfill the security requirements then nobody can use it [the system]."***

Discussion of requirement change

The security requirements have always been at the foundation remaining stable, so they have not been up for discussion. They do not use a requirement specification but rather goals to be accomplished. The goals have been brought up more than security requirements at discussion meetings, especially if the goals had a time limit. If something was to happen in five seconds and it took five point three, it would usually be fine. On the other side if fifty documents were to be encrypted and two pass unencrypted, it would not be acceptable. It was not a case that as long as ninety-nine point nine percent was encrypted it would be fine. On the security side no lenience was given to the machine being busy or something only happening in that specific incidence.

Discussion of process improvement

They use the same process every second week with a retrospective after the sprint, where it is discussed what was done and what could be improved. This regardless to whether the project had high or low security requirements.

Difference in delivery of working software

Instead of mentioning any difference, interviewee explained their process of delivery. First the build is sent to a few thousand internal users and if it works for them it is expanded to a thousand more. Then a hundred thousand. If everything works well a small number of people in different countries and continents are given the build, a number that gradually increases. Those who receive the build get the new code every time there is an update. The end users usually receive the software at the end through CD or download, estimated to come every second year or so.

Interview #2

Customer collaboration

There was a major difference in customer collaboration depending on the size of the project and matureness of the customer. Professional customers with good knowledge of information technology would have requirements to security. In the project with high security spoken of in the interview, the customer had an employee whose sole role was focused on security. It fell naturally for the customer to have clear demands to security, but some may have been unrealistic or in a strict sense not necessary.

> ***"Sometimes it can be easy to like read about all the security mechanisms that exist and cross that we want this, this, this, this and this without considering the consequences and if it makes the security any better."***

At smaller customers they had experienced few if any requirements to security due to a lack of knowledge from the customers. It was then the company's responsibility to make the requirements and inform the customer that security needed to be prioritized, which touched upon the issue of discussing functional and non-functional requirements with the customer in agile development.

Occasionally they experienced problems with getting anonymous test data. Most the time the focus lies on security when the system is in production and it kept becoming an issue that the company had to write code to get the test data anonymous themselves. They would rather not have their test environments full of sensitive data.

Openness to requirement change

Interviewee said it depended on their contract. If it was a set price contract expecting them to deliver software to a given price, they had to handle a change request notice process for the changes to be done. This was especially the case for security requirements as any new additions could come with a huge cost. On their agile Scrum projects they are more open for changes and it becomes the company's responsibility to explain what the change will cost the customer, how important it is, if it can be done later and if there are simpler alternatives. On the same level as other functionality. The company keeps itself to a high level of professionalism for their projects, not allowing anything to pass without there being at least a minimum of security mechanisms if it is to be put up on the internet.

Discussion of requirement change

System requirements were said to be more likely discussed as the security requirements were often given ahead of time and were non-functional. They usually did not experience much politic of wishes from the customer when it came to non-functional requirements. Often they had to do standard solutions where they got far just by following good practices from ASP.net, not writing their own encryption, use standard Java.net, use SSL, test against OWASP top ten threats and general guidelines

when it came to web solutions. There was usually more discussion surrounding the functional requirements and they were open for change.

Discussion of process improvement

No difference between high or low security projects. In the past they followed Scrum a bit closer, considering the security part and discussing how it worked for their project. A lot of code review was done to begin with and checking implementation against recommended guideline. But this was all said to be done on the same lines as other functionality.

Difference in delivery of working software

Security played an important part in the difference of delivery. For projects without high requirements on security they delivered working software on each commit. In these projects the most important thing for the customer was to get ideas tested out and receive feedback from real end-users. What they called innovation projects. For those projects, it was difficult to know if what they made was right without testing it.

> ***"It was within the medical sector so you would believe there was an enormous security focus. But what we instead did was to design the solution in such a way that there were never any huge consequences if someone got ahold of the data because no personal data was used. So instead of operating with social security numbers and names we had series of numbers that could be used in an external system to connect to people if one wants, but one could never get hold of the data to compare and connect them to people. There was nothing to identify someone in the system."***

In such projects there was always a plan for how the security should function, especially authentication and integration to the back of the system. As long as they followed the architecture principles, security could be delayed so functional code could be delivered daily.

The high security project on the other hand was not put to test against real user before a long time had passed.

> ***"It has always been a problem with agile development that you should have shippable code but it ends up usually potentially shippable. In practice it is rarely set for production before one has a large chunk of the system done. It is often what is different between iterative and incremental development, that is iterative is the thought that you should like continuously improve the production set and incremental is that you only make first a little bit and then a little bit, then you set the whole part for production."***

For the high security project, it was more the case of being an incremental project where functionality was continuously developed but could not be put to production because the whole package was not there. Due to technical complexity they were early in a test environment and prioritized that. The thought was to grab hold of difficult tasks early and not just believe they would be able to manage it. Interviewee said that the reason for doing it like that was not only due to security but because the customer wanted everything tested completely in the test environment and get end-users to come in and test it there, not roll it out to real users.

Interview #3

Customer collaboration

In their case the customer has to set the requirements. In the cases where the customer did not have the required competence, they as the professional part had to contribute especially when it came to non-functional requirements the customer did not see.

Openness to change in requirements

Interviewees did not see a difference in openness but mentioned they usually did not work on projects with a high demand for security. No experience with any difference in their own projects but would maybe look differently on it had they done projects for the military or banks.

Discussion of requirement change

No difference. From the customer's side there may be someone with a focus on security, but most are concerned about functionality and it is also there changes are made most often. Security requirements are usually derived from a ruleset that details the security level, or arise as a result of threat modeling during the project. The security requirements begin as something abstract and then become more concrete, which fits agile. The challenge was keeping the view of the entire project and securing the transition between abstract to something specific, which was a danger in agile projects as one focuses more on implementation. Something the traditional waterfall method did better by settling everything in advance.

Discussion of process improvement

No difference. They treat security requirements the same way as other non-functional requirements.

Difference in delivery of working software

No difference as they treat the security requirements like other requirements. It was speculated that a higher requirement to security would require a part of the team to focus on security and it would have been a different dynamic for delivering software, but they have no experience with such situations.

Interview #4

Customer collaboration

In their case the project included several consultancy firms in addition to internal resources on the development side which all were a part of architecture and implementation. So there was only a subtle difference between customer and developer. It had been up to the project managers consisting of both interns and externs to both prioritize and lead the project, where the customer-consultant dialogue had been very close. Interviewee expressed difficulty in answering this question as the customer had been a part of development.

Openness to change in requirements

On the security side they followed the principles in the guidelines of The Norwegian Data Protection Authority for treatment of sensitive personal information, and how it is handled from security zone to security zone. On that part they were very strict and it would take a lot to be convinced the guidelines needed to be broken. The security requirements of the module were more open when it came to functionality.

> ***"Many claim from a research perspective that security requirements should tell you what to do but not how. That is, what is the requirement is important but it should be up to those who implement it to propose how it is best done. For instance a security requirement could say you should have crypto in the communication tunnel from A to B, and then it is up to the developer to say 'Okay, here we will use asymmetrical cryptography, here we will use symmetrical, we will use 256-bit keys,' all the detailed things."***

As long as the principles are kept and it satisfies the requirements of performance, the security requirements could be changed. A warning was here given that it did not take many seconds difference before it reverberated to other components of the system, slowing them down.

Discussion of requirement change

Interviewee did not have enough insight to answer.

Discussion of process improvement

As the interviewee joined the project late, they felt it was difficult to answer as most had already been established in earlier discussions. For the question, those early meetings would be the most important for influencing improvement to the process.

Difference in delivery of working software

The most important difference came to the way the software was tested. In security tests, deployment and how often code is sent from the project team to management. They did not deliver continuously to the end-user before it had been thoroughly tested. It was not like other agile projects, for instance a web application or an internet store where a new feature is made and deployed at once. Instead they made the code, presented it for their internal customer and did not implement it in the production environment before a larger release.

Difference in…	Interview #1	Interview #2	Interview #3	Interview #4
… Customer collaboration	No (internal customer)	Yes (varies).	No.	No (but project had overlapping customer-developer)
… Openness to change	Yes (high locked security)	No (for agile).	No.	Yes (security principles maintained)
… Requirement change discussion	Yes (less likely to adjust security)	Less for security.	No.	Unknown.
… Process improvement discussion	No.	No.	No.	Unknown.
… Working software delivery	Unknown.	Yes.	No.	Yes.

3.2.3 Requirement assurance

To ensure the security requirements had been met, the company in the first interview had an internal group reviewing security as a main contributor. As well, everything put in was coded in full and tested. Automated tests ran to ensure there were no access violations, wherein you had access to information you should not have.

They did not use an external part to test the security but did it internally. The interviewee did not know if other internal groups were used as well but mentioned it was possible they had other company divisions in USA to test security. Penetration testing had been done in a different company's project the interviewee had worked on, where one division often tested another's project.

In interviewee 2's case, four concrete measures had been done to ensure the security requirements had been met. First, a developer had had security as their responsibility. Second, they did a code review where one step was the security aspect, in which they talked through how the code was made and how it had been secured. The third was the use of an internal security expert to run security testing for a day. Fourth was an external company hired to test the security. They also built automated testes.

> ***"In ASP.Net if you are to post a form from a website to a controller you have to declare the method with HTTPPost which says it can receive a post request, much like a notation in Java. Then you have something called… another notation which says this post needs to have a cross-site request forgery token to avoid cross-site request forgery attacks… and we had as policy that all posts needed to have that unless they were written on one or another exception list, that we have a very specific reason to post without them. But it could happen that a developer forgot to put that attribute on. So we made an automated unit test which scanned all the controllers, found methods with HTTPPost, and checked if it also had the cross-site request forgery attribute, if not the test failed. So if someone had forgotten it then the whole build collapsed and you got a message at once that either you had to add it or you had to add it to an exception list. You had then documented that you had taken an explicit choice."***

The company tried to get as much as possible automated related to security testing. In addition an internal group of five to seven people would do security review for projects for a day's duration. This group focused only on this security work and worked outside other projects. It was unknown whether they used an external part to test the system. Such was usually done on a case to case basis, on initiative from the customer.

Third company mentioned having good routines to check the security requirements, with tests, inspections, code reviews and automated testing depending on the situation and project. They did not use external parts to test the security, but the developer interviewed mentioned having been a part of projects where they had hired in an external part for penetration testing. It was also something recommended to do.

Interviewee 4 said that assurance for the security requirements was done through having requirements, module descriptions and update module descriptions which reflect what had been implemented as the most important process. To have the solution documented, which was also the most difficult part. The security expert used the documentation as a means of verification when approaching the developers and asking what they had actually made. In addition they had security tests and extern parts that did security review based on the documentation.

They also used neutral third parties for code revision and penetration testing of the solution.

	Interview #1	Interview #2	Interview #3	Interview #4
Assurance methods	Internal group code review, automated tests,	Security expert, code review, testing day.	Tests, code review, good routines. Occasionally automated tests.	Requirements, documentation.
Uses external part to test security?	No (internal, possibly other divisions)	Yes (customer initiative, internal as well)	No.	Yes.

3.2.4 Topics brought up

Interview #1

The advantage of agile is that you adapt to the world swifter. The drawback is that in practice development is done less agile, in that you use a process on the outside that takes longer time. Security was also not put in to the additions to begin with, as it gets harder to code and test with the security added. So such usually came afterwards. Another advantage lies in being more conscious to changes but in practice the security requirements are perhaps the least agile thing they work on, since they are not as open to change. Some security related tasks are drawn out of the sprint, such as the early specification phase, discussions with other groups, meetings and review. Or the tasks get moved from sprint to sprint. It is something they experiment on and could likely fit into an agile method.

"If we had had an external customer and we had only a demo of something that worked and said 'No, now we will only put on security'. What you see as the difference is if the grade is green instead of yellow, right? And then we spend months on that. It would be a lot more difficult to… [convince the customer that it was important]."

Interview #2

One difficulty lay in test data. In their experience the customer is sloppy when it comes to test data and good procedures to make test data anonymous. Another challenge is the prioritization of functional and non-functional requirements, and depending on customer matureness it can be difficult to prioritize security.

As a subjective view from the interviewee, the focus security has had in mass media the last two years has made it easier for security to be prioritized, that the customer has been more alert when there has been a lot of writings about security and wanted to know if it affected them on meetings. In the past security revolved around windows updates whereas now there are databases and passwords on the loose, stock companies vulnerable and focus on security events in the newspapers. So it has become easier to prioritize security and made it less something they had to sell the customer.

To have safe defaults in the development frameworks has been very useful and external reports have complimented that the company followed good practice, because the framework laid the groundwork.

There are continuously new challenges related to security but also new technology that makes it easier to withstand most attacks. However, new attack vectors keep cropping up. You are never done working with security.

New developers may not understand fully the frameworks and that can be a challenge. If you do not understand how it works, it is easy to make mistakes. It is important to know how the technology do security.

Interview #3

Security has to be tested from end to end in the whole solution. It is hard to make automated unit tests to secure some components when you may have to put them together with others to see that it works. This is a usual challenge for automated tests. Security tests also face the challenge that security functionality is not visible, so there is a danger of it being removed when code is changed. The tests need to be able to pick that up. It is also difficult to set up a test environment for penetration testing as it assumes you have a large amount of mechanisms in place, something you usually do not have until the program is set for production and then it is beyond the agile process anyway. Either you get a long loop

outside the agile process or you are already done with the project. Some functionality you cannot test within the agile method, such as performance at a hundred thousand users.

Test data has always been a problem as in most cases you hold data you are not allowed to handle, so it becomes a security issue in treating sensitive data as nearly all consumer data is sensitive. Such as handling social security numbers outside secure environments. Laws are occasionally bypassed in such cases.

In response to a question about how hard it is to convince customers of the need for security, they responded that it is never considered nice by the customer to pay for non-functional requirements and that is where the problem lies. Usually persuasion happens on two stages where first revolves around expressing awareness and need, the other about getting finance to do the changes as the customers may have a hard time considering it necessary. It is usually here, they say, that more compromises are done than on the functional level.

Interview #4

It is an eternal question for discussion what is best between traditional and agile development on a security perspective. A topic which would need more research to determine.

There had been no problems with test data, but what they used depended on how it was to be tested and if it was sent out to a third party. They used their own data for internal testing. Different practice was used at third parties, depending on the test environments they used.

> ***"Information is information and, it should be secured in the same way regardless if it is for test or development."***

It has not been difficult to get test data on their own solutions. They take a sample of the data available and perform procedures so they can be tested in a good manner.

Section 4: Discussion

It is unclear why the questionnaire did not receive any results. IKT-Norge was approached both to be able to spread the questionnaire to a far greater amount of companies and to have a professional backing making it more likely the companies would respond. It was expected to not get a huge amount of respondents due to the expectance of between ten to thirty percent answers and security being a sensitive issue. Receiving zero responses was shocking.

While it is unknown the exact number of companies the questionnaire reached, it is fair to assume the number is in the hundreds since IKT-Norge is an interest group for those who work with ICT as a business. Likely their newsletter reached one or more employees at each of the companies that may not be able to respond to the questions posed by the questionnaire or simply skip it entirely, but it should not be an entirely too big leap of logic that there would be some company wherein the questionnaire would be passed on to someone who could answer. While the questionnaire contained some open textboxes for questions that would increase the time for answering it, in itself the questionnaire was short enough that it should not take too long to answer. Reviews alongside the supervisor, the pilot study and IKT-Norge's own brief look would have weeded out that danger. The issue of security sensitiveness when it comes to a questionnaire is a more difficult matter to assess but there was nothing too specific there that should raise an issue. Whether SurveyMonkey is regarded in a bad light for the business world in Norway is unknown but could also pose a reason to the non-existent responses. To put it simply, the researcher has no idea why it did not produce any results and another round would likely have resulted in the same without knowing why there were no answers in the first place. At least from the assumption that if everything was fine, there would have been at least one respondent due to the sample size.

Concluding that Norwegian companies are not willing to share information regarding security experiences would be easy but both the research of the pre-project study and the openness in the interviews speak otherwise. It is likely as a result that it is easier to trust someone met in person if the security topic was the issue.

Something that was noticeable in the interviews when asking about the differences in agile principles was that the questions were sometimes not answered in full. Not everyone compared high security projects with other projects when it came to the different principles, so the questions themselves may have been too broad and would be better split up. While they were in a different form in the questionnaire with Likert scales, it could be the questions were too unclear

To answer the first research question it must be determined whether any of the four companies are deviating from the standard methodology. It is important to keep in mind that most agile methodology are standard methods to be shaped as the development team requires and pose guidelines to follow. For instance a very strict stand on an agile method is that it should produce no documentation at all, as agile is meant to avoid long sessions of planning and instead allow the developers to begin to develop from the get-go. In some cases this may not be possible at all. As it is mentioned in the manifesto:

> "[… we have come to value] Working software over comprehensive documentation…
>
> That is, while there is value in the items on the right, we value the items on the left more" (Beck, K. et al. 2001)

Nowhere does it say no documentation at all. So that alone is not enough to call it deviating.

In the first interviewed company's situation, the agile methodology had shaped how they approached development. Instead of fitting the method to them, they fit themselves to the method. There were different roles, a sprint process, daily standups and reviews at the end. Software was developed with a demo at the end of the sprint. Some notes of concern method-wise came from the mention that security work was drawn out of the sprints. There was a lot of specification work done at the beginning of the project due to the high security, work with security tended to be delayed from sprint to sprint to allow for demoes that were insecure, meetings and discussions and code review were done outside in a separate process. While work with specification ahead of the project is, as has already been argued, not enough to call a deviation and is a useful technique security-wise; leaving security parts of the user story in the backlog over time and taking security work out of the sprint itself is breaking the method. The sprint planning meeting is the place where the security work to be done is discussed and set for the iteration, it would seem that the first company take that out of the process. If it is so, the company is deviating from the Scrum method. This is similar to another security firm's experience incorporating security into Scrum, where they had to reshape their approach to fit the Scrum methodology (O'Boyle, R. and Eng, C. 2013). Related to RQ1 this appears to be nonconformity done for reasons of security, but as said by the interviewee they could likely fit it into their agile method. Because of the high security requirements coming from both the company itself and the internal customer, in this case it would seem security is strengthened as it appears to be zero tolerance for mistakes, even if the iteration demoes are insecure. The results given the end-users are so thoroughly tested and reviewed that it should not be a problem.

The second company explicitly stated they did not follow Scrum by the book but had it as a starting point and adapted it for their different projects, ending up with something close to Scrum and Kanban. So there is no question that they have deviated from the method. Interestingly, they followed Scrum more by the book when working on a high security project and had added on a security component that could be fit into the methodology easily. This as a result of the customer wanting to know how their development method would assure security was implemented well. It would seem related to RQ1 that a more by the book development was done specifically due to the security need. Once the development had been done and the project entered a management phase, they drew back on both allocated

resources and method use as the foundation had been laid and anything new would fall within the tested and proven methods or processes in the software. For RQ1 this means that the deviation is not done as a cause of security but development phase change. It does not affect security in any noticeable manner, as any new security features fall under the same secure patterns tested.

The second consultancy firm, the third company interviewed, had made some adaptations fitting the method to their development but stated they had done no changes to the method itself. They fit in security just like any other non-functional requirement into their agile process. One thing to remark related to this is that they did not work on projects with high security requirements. Judging from the rest that was told about their method, it would seem they keep it agile without deviating and so is not relevant for the first research question.

As for the fourth company, the interviewee said that one never follows an agile method to the letter but could only come up with the daily meetings as a change to the methods in Scrum. It is unknown in what way or how the daily meetings have been changed. What is known is that they use a third party for penetration testing and code review. While arguably highly beneficial on a security perspective, it relies on someone outside the development team or customer to both work and report back their findings of the tests outside the sprints themselves. Penetration testing or external code review could then be considered as assurance methods falling outside the agile methodology, causing deviations for reasons of security. It is also a reason why such a question was added the questionnaire. The conclusion that such assurance methods fell outside the agile method was had by Beznosov and Kruchten (2004). Again, such a deviation from a security point is a beneficial thing and contributes to keeping the security requirements and is in this case the clearest reason for the deviation. Everything else beyond the unknown daily meetings keep within the method.

When it comes to the second research question, it falls to consider how the principles are affected in the agile method when working on high security projects. Of the four companies interviewed only the first, second and fourth work on projects with high security requirements.

Customer collaboration seems to be in two of three cases unaffected when working on high security projects. Both the two unaffected had an internal customer, so that may be the reason for it. For the one affected the reason seems to be due to customer matureness and knowledge related to security. When the customer knows little, it becomes more an issue for the consultancy firm as a professional part to provide the security requirements. This was also true for the third company interviewed, even if they did not work on high security projects. The other side of it is that the customer may demand more from the company. As customer collaboration is a big part of agile development and security can be seen as any other requirement, the fact that the customer is less or more involved on security requirements does not seem remarkable for the research question on its own as this could be true for other requirements as well depending on the customer.

Regarding openness to change of requirements, the first and the fourth company both had either set requirements or principles outside the team that were more or less set in stone. The more functional security requirements could very well be adjusted. In the case of the second company, openness to

change in security requirements and other functional requirements were the very reasons for choosing an agile approach.

The two interviewees of the high security projects that could answer whether there was a difference in the discussion of requirements both replied security requirements were discussed in a lesser degree than other system requirements. This was due to most security requirements either being strict or having been set at the beginning of the project, not being something the customer usually would change. As the customer is more concerned about functionality, that security requirements face little change is not necessarily a bad thing for agile.

Both the two that could respond regarding discussions about process improvement with a high security project and the third company expressed no change whatsoever when it came to discussing how to make the process better in working with security requirements. The requirements were treated the same as the system requirements or the non-functional requirements.

What seems to be affected most of the agile principles investigated is the delivery of working software. While there is no lower security project to compare with for the first interview, it was mentioned that security was delayed to be able to present demoes at the end of every sprint in high security projects. The second company would deliver software at every commit in low security projects, relying instead on obfuscating the information so that extracting data would be to no gain. In their high security project it was the customer's wish they kept most updates in their test environment. For the fourth company, in contrast to a web application where a new feature and updates would be rolled out immediately, new functionality would not reach the end users before thorough testing and presentation, as a part of a larger release. Summed up high security projects seem to be slower in shipping software, due to the need of assurance.

It would seem security is the result and benefit of deviating from an agile methodology judging by the two high security projects that did so. However, the second company kept more true to the agile methodology because of a security reason and the deviation happened when it would not affect security itself. More research would be required to know whether this is true on a larger scale. From the results of the pre-project study, the findings seemed to indicate that the deviation in method was a cause of development convenience more than a security reason.

An argument supporting agile development methods for security-critical software is that following a more traditional model when the security requirements are unknown or liable to change throughout development is impossible without resorting to costly steps back in development, as mentioned by the second company.

One issue comes from the lack of overview one has in agile development. While guidelines and the idea of the system are in place and for each sprint it is known what is to be done, it is harder to know how everything connects. On a security perspective you need to have the overview to secure the transition from one end to another. This is much easier in traditional methods as everything is planned out in advance.

As is the case of interviews, it is possible that the interviewees answered in a way to present their company in the best possible light or differently due to the researcher's role as a student (Oates 2006 p.189). Attempts were made to mitigate only positive responses by asking specifically if there had been challenges as well in their experience. It could also be the case that asking someone else in the company would give different answers. Another researcher may find more elaborate or different views expressed if the same interviews were done again. It is the researcher's opinion that all the interviewees were honest and forthcoming in the answers given, showing no signs that they were ill at ease during the interview but rather welcomed the chance and enjoyed to talk about their experience with agile security. Promising anonymity likely helped mitigate reasons for the interviewee to not answer the questions completely and while most of the interviewees were told in advance they could refrain from answering, none took the privilege. Only when something was unknown did an interviewee not provide an answer and specifically told this was the case, as can be seen in the results of interview #4. Use of mirroring throughout the interviews may have encouraged certain topics or themes more than others, but the researcher will argue that alongside an open body language it also just as likely encouraged more openness. The validity of the answers found in the research can be put to question, as a fifth interview could likely discover something entirely else than what was found in those made. Even if the company interviewed was in the same region as the rest. A good rule-of-thumb for small-scale research projects is to have a final sample of at least thirty for surveys according to Oates (2006 p. 100). The sample size of this thesis is far below that due to the lack of results from the questionnaire and the time it took to arrange interviews. For such reasons what was discovered may not be the norm in companies working with agile and so the results cannot be used to generalize for a larger population, both due to the low sample size and the non-probability approach of the interviews. The uncertainty factor is too high.

What could be interesting for future work is to ask specifically the companies if they had experienced any failures as a result of their methodology. While challenges were brought up during the interviews, none mentioned any specific failures they had had and such may be more telling for the study than successes.

Because all the information and quotes have been translated from Norwegian, there is a risk that some nuances have been lost.

Section 5: Conclusion

RQ1

How is security affected in deviations of agile development? Are the reasons for nonconformity security related?

Based on the interviews security is either more assured or unaffected in deviations of agile development. In two of three cases of nonconformity, the reason was due to security. Due to the low sample size of interviews, there could be a different answer entirely in a fourth deviation method. This is also based on local companies in the Trondheim region.

While this research question was not intended as an overall generalization of companies in Norway, further work could see if there are different reasons not only regarding security, but also between types of companies and countries. A larger sample size would also be able to answer the question more firmly if nonconformity is due to security reasons and if security is affected in other ways in such methods.

RQ2

Can agile development methods be used for security-critical software?

The sample size in this thesis and the confined region the data was gathered from makes it impossible to answer this question with a generalization, even if the focus is on Norwegian companies alone. Results seem to indicate that yes, agile development methods can be used for security-critical software and may be the best option between that and traditional methods if the security and other requirements are unknown or liable to change. However many assurance methods fall outside the agile approach and can affect development in a non-agile way.

Future work could expand upon the sample size to answer the question in full, and compare the results to research done on traditional plan-driven software development methods and security-critical software to see which group of methods creates the most secure code. Or which group of methods to use based on project criteria.

References

Bartsch, S. (2011) 'Practitioners' Perspectives on Security in Agile Development', in *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on*. Vienna: p. 479-484

Beck, K. et al (2001) *Manifesto for Agile Software Development.* Available at: http://agilemanifesto.org/ (Accessed 30th of May 2014)

Beznosov, K. and Kruchten, P. (2004) 'Towards Agile Security Assurance', in *NSPW '04 Proceedings of the 2004 workshop on New security paradigms.* New York: ACM, p. 47-54

Brevik, E. and Grønli, T-M. (2013) 'The Scrum Illusion: Use of Software Development Methods in the Norwegian Itindustry', in Fallmyr, T. (ed.) *NOKOBIT 2013,* Norway: Akademika forlag, p. 15-28.

Heartbleed, 2014. Available at: http://heartbleed.com (Accessed 11th of May 2014)

Knudsen, Anders Nordli (2013) 'Agile Security Requirements', Pre-project for Master Thesis. Faculty of Information Technology, Mathematics and Electrical Engineering, Norwegian University of Science and Technology in Trondheim.

Oates, B. J. (2006) *Researching Information Systems and Computing.* London: Sage Publications Ltd.

O'Boyle, R. and Eng, C. (2013) *From the Trenches: Real-World Agile SDLC.* Available at: https://www.its.fh-muenster.de/owasp-appseceu13/rooms/Grosser_Saal/high_quality/OWASP-AppsecEU13-ChrisEngRyanOBoyle-FromtheTrenchesReal-WorldAgileSDLC_720p.mp4 (Accessed 13th of December 2013)

Sullivan, B. (2010) *Security Development Lifecycle for Agile Development.* Available at: http://www.blackhat.com/presentations/bh-dc-10/Sullivan_Bryan/BlackHat-DC-2010-Sullivan-SDL-Agile-wp.pdf (Accessed 10th of November 2013)

SurveyMonkey, Available at: https://www.surveymonkey.com/ (Accessed 4th of June 2014)

Trimble, J. and Webster, C. (2012) *Agile Development Methods for Space Operations*. Available at: http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20120013429_2012013093.pdf (Accessed 12th of December 2013)

Appendix 1: Questionnaire

Sikkerhet i smidig utvikling

Denne undersøkelsen er gjort i forbindelse med en masteroppgave om sikkerhetskrav i smidig utvikling. Alle svar er holdt anonyme. Om noe er uklart eller du har spørsmål, ta kontakt via mail til anders.nordli.knudsen@gmail.com.

Hvis dere er villige til å stille opp for intervju rundt dette temaet, send en mail til anders.nordli.knudsen@gmail.com.

Takk så mye for at dere svarer på denne undersøkelsen!

***1. Bruker dere smidig utvikling i deres prosjekter?**

- Ja
- Nei

Smidige utviklingsmetoder (agile development methods) kjennetegnes ved kortere utviklingssyklusser, mindre prosjektteam, tettere samarbeid med kunden gjennom prosjektet og iterativ utvikling. Noen kjente former for smidig utvikling er Scrum, Kanban, Feature Driven Development og Extreme Programming,

2. Om dere ikke bruker smidig utvikling, hva er grunnen til dette?

Om dere ikke bruker smidig utvikling kan dere trygt hoppe over resten av spørsmålene. Igjen, takk så mye for deltagelsen!

3. Er det blitt gjort endringer for strukturen til den smidige metoden dere benytter?

- Ja / Vi benytter en egen metode
- Nei
- Vet ikke

4. Om det ble gjort endringer, hva var grunnen til dette?

5. Bruker dere en ekstern part til å teste sikkerheten til systemet (f.eks. penetreringstesting)?

- Ja
- Nei

På de neste spørsmålene vil du bli bedt om å svare på deres smidige metodebruk. Spørsmålene er gruppert inn i et spørsmål for ordinære prosjekter, et spørsmål for prosjekter med høye sikkerhetskrav og et kommentarfelt som kan benyttes til å forklare eventuelle likheter eller forskjeller.

6. I hvor stor grad er dere åpne til kravendringer underveis i smidige prosjekter?

Kan ikke gjøre endringer	Kan endre noen krav tidlig i prosjektet	Kan endre viktige krav tidlig i prosjektet	Kan endre de fleste krav gjennom prosjektet	Kan endre alle krav gjennom prosjektet
○	○	○	○	○

7. I hvor stor grad er dere åpne til endringer av sikkerhetskrav underveis i smidige prosjekter med høye sikkerhetskrav?

Kan ikke gjøre endringer	Kan endre noen krav tidlig i prosjektet	Kan endre viktige krav tidlig i prosjektet	Kan endre de fleste krav gjennom prosjektet	Kan endre alle krav gjennom prosjektet
○	○	○	○	○

8. Kommentarer for spørsmål 6 og 7:

9. Hvor ofte leverer dere fungerende software til kunden i smidige prosjekter?

Sjeldnere enn en gang i måneden	En gang i måneden	Hver andre eller tredje uke	Hver uke	Oftere enn hver uke
○	○	○	○	○

10. Hvor ofte leverer dere fungerende software til kunden i smidige prosjekter med høye sikkerhetskrav?

Sjeldnere enn en gang i måneden	En gang i måneden	Hver andre eller tredje uke	Hver uke	Oftere enn hver uke
○	○	○	○	○

11. Kommentarer for spørsmål 9 og 10:

12. Hvor ofte har dere møter for å diskutere effektivisering av prosessen i smidige prosjekter?

Oftere enn en gang i uken	En gang i uken	En gang hver andre uke	En gang i måneden	Sjeldnere enn en gang i måneden
○	○	○	○	○

13. Hvor ofte har dere møter for å diskutere effektivisering av prosessen i smidige prosjekter med høye sikkerhetskrav?

Oftere enn en gang i uken	En gang i uken	En gang hver andre uke	En gang i måneden	Sjeldnere enn en gang i måneden
○	○	○	○	○

14. Kommentarer for spørsmål 12 og 13:

15. Hvor ofte har dere møter for å diskutere kravendringer i smidige prosjekter?

Oftere enn en gang i uken	En gang i uken	En gang hver andre uke	En gang i måneden	Sjeldnere enn en gang i måneden
○	○	○	○	○

16. Hvor ofte har dere møter for å diskutere sikkerhetskravendringer i smidige prosjekter med høye sikkerhetskrav?

Oftere enn en gang i uken	En gang i uken	En gang hver andre uke	En gang i måneden	Sjeldnere enn en gang i måneden
○	○	○	○	○

17. Kommentarer for spørsmål 15 og 16:

18. I hvor stor grad er kunden med på å lage kravene til systemet i smidige prosjekter?

Sjeldent eller ikke	I liten grad	Stort sett likt med prosjektgruppen	I større grad	Kunden bestemmer det meste eller alt
○	○	○	○	○

19. I hvor stor grad er kunden med på å lage sikkerhetskravene til systemet i smidige prosjekter med høye sikkerhetskrav?

Sjeldent eller ikke	I liten grad	Stort sett likt med prosjektgruppen	I større grad	Kunden bestemmer det meste eller alt
○	○	○	○	○

20. Kommentarer for spørsmål 18 og 19:

21. Noe ellers forbundet med deres smidige metodebruk og sikkerhetskrav som ikke har blitt tatt opp?

Ferdig

Appendix 2: Interview Questions

Do you utilize an agile development method in your company's projects?

(If so) Which one?

Do you use the same method in projects with high security requirements?

Why (not)?

Has the method been changed to accommodate development?

Has the development method had any challenges in projects with high security requirements?

Has the development method had any advantages in projects with high security requirements?

How often is the customer helping to create the requirements of the system, comparing system requirements and security requirements?

Are there any differences as to how open to changing requirements you are, between system requirements and security requirements?

Is there a difference between how often you discuss requirement changes, between system requirements and security requirements?

Is there a difference between how often you hold meetings to discuss process improvement, between projects with high security requirements and other projects?

Is there a difference in how often you deliver working software to the customer, comparing projects with high security requirements and other projects?

How do you ensure the security requirements have been met?

Are you using a third party to test the system?

Is there something else related to agile development and security we have not spoken of that you would like to mention?