TAMK University of Applied Sciences, Master's Degree
Degree Program in Information System Competence
Päivi Vuori

PROFESSIONAL MASTER THESIS

Development of Agile software production to the organisation

Supervisor M.Sc. Pekka Pöyry
Tampere 10/2009

Tampereen ammattikorkeakoulu, ylempi amk-tutkinto
Tietojärjestelmäosaamisen koulutusohjelma

Tekijä	Päivi Vuori
Työn nimi	Agile ohjelmistotuotannon kehittäminen organisaatiolle
Sivumäärä	53
Valmistumisaika	10/2009
Työn ohjaaja	FM Pekka Pöyry

TIIVISTELMÄ

Ohjelmistotuotantoprosessi kuuluu osaksi jokaisen ohjelmistoalan yrityksen käyttämiin prosessimalleihin. Ohjelmistotuotannon kehittämisen tarkoituksena on parantaa olemassa olevaa tuotantoprosessia niin, että prosessi edesauttaa tuotteen tekemistä ja yrityksen kustannuksia sallitulla tasolla.

Työskentelen Nokia Siemens Networks BSS (Business Support Systems) RD SV Verification -organisaatiossa, joka käyttää ohjelmistotuotannossaan ketterää kehitysmenetelmää (Agile). Organisaatiossani tuli esille ajatus siitä, kuinka toimitettavan tuotteen laatua ja toimintatapoja pystyttäisiin parantamaan. Sen seurauksena päätettiin keskittyä kehittämään olemassa olevan tuotteen prosessia ja hyödyntää saatuja tutkimustuloksia. Kun kehitetään tuotteen tuotantoprosessia, niin se koskettaa aina koko yritystä, ei ainoastaan organisaatiota, johon kehittäminen kohdistuu.

Tämän opinnäytetyön tavoitteena oli tuoda esille prosessista ne kehitettävät kohteet, jotka nousivat esille haastatteluissa ja tarkastella kehittämiskohteiden onnistumista. Haastattelujen pääasiallinen tarkoitus oli tuoda esille tekijöiden näkemyksiä omasta tekemisestään ja käyttää saatuja tuloksia apuna kehitettäessä tuotteen tuotantoprosessia. Kyselyn tarkoituksena oli tarkastella kehittämisjakson lopussa kehitystehtävien onnistumista ja kerätä jatkotoimenpiteitä varten ne kohteet, joihin olisi keskityttävä tulevaisuudessa.

Alan kirjallisuuden avulla haettiin kokonaiskuvaa ohjelmistotuotannosta ja eri ohjelmatuotantomenetelmistä. Kyseisiä menetelmiä vertailtiin keskenään ja niistä haettiin sellaisia kehitysmalleja, jotka tukisivat olemassa olevan prosessin kehittämistä ja antaisivat ideoita kehittämiselle. Haastattelutuloksista saatiin kerättyä prosessin kehittämistä varten kehitettäviä kohteita. Kehittämisjakson lopussa tehtiin kysely. Kyselytuloksilla mitattiin, kuinka hyvin kehityskohteiden kehittäminen oli onnistunut. Kirjallisuudesta oli myös hyötyä, kun haettiin analysointimetodeja haastattelutulosten analysointeihin. Lopputyössä käytetty teoria on saatu lähdeteoksista ja sitä on sovellettu tekijän näkemyksen mukaisesti kyseiseen työhön.

Tutkimuksen tuloksena saatiin kehitettyä olemassa olevaa prosessia eteenpäin, vaikkakin yrityksen sisäiset muutokset toivat lisähaasteita prosessin kehittämiselle. Tulevaisuudessa prosessin kehitystyö tulee jatkumaan muualla ja näistä tuloksista saa hyvän lähtökohdan seuraavaa kehitysprosessia silmällä pitäen.

Avainsanat Ohjelmistotuotanto, ohjelmistotuotannon prosessit, ketterät ohjelmistotuotantomenetelmät, ohjelmiston testaus

TAMK University of Applied Sciences, Master's Degree
Degree Program in Information System Competence

Writer	Päivi Vuori
Thesis	Development of Agile software production to the organisation
Pages	53
Graduation time	10/2009
Thesis Supervisor	M.Sc Pekka Pöyry

ABSTRACT

The software production process belongs partly to every software company consumed process models. The meaning of development of software production is to improve existing product process that enhances the product working and diminishes the company's expenses on the allowed level.

Nokia Siemens Networks BSS (Business Support Systems) RD SV Verification organization uses software production by Agile. An idea about how the product quality could be improved by the product line came up. It was decided to begin developing the product process and exploit the research results. When developing the product process, it affects the whole company, not only the organization that the developing is done in.

The purpose of this final thesis was to bring out development targets that came up from the interview and to consider how well development targets went. The interviews' main meaning was to bring out the employees' point of view of their own working and to use the received results to help develop the product production process. The purpose of the inquiry was to dissect in the end of the development phase the succeed of development tasks and collect next targets for further development.

Literature was used to get to know software production and different production methods. The methods were compared with each and development models were collected, which supported for the existing process developing and gave ideas for the development process. From the interview results were collected development targets for developing the process. In the end of the development phase the inquiry was made. The inquiry results measured how well succeeded the development targets were. Literature was a benefit as a search for the analysis method for the analysis of interview and inquiry results. The thesis' theory is from the source books and it is adapted to this work according to the writer's view.

The results of the research got to existing process go forward although company's internal changes brought more challenges to the process developing. In the future, the processes development work will continue somewhere and these results are giving a good starting point for the next development process.

Keywords Software production, software production processes, agile software development methods, software testing

Table of contents

Terms and abbreviations

Agile	Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams.
Likert scale	A Likert scale is a psychometric scale commonly used in questionnaires, and is the most widely used scale in survey research. When responding to a Likert questionnaire item, respondents specify their level of agreement to a statement.
Product backlog	A prioritized list of high level requirements.
Scrum	It is an iterative incremental framework for managing complex work commonly used with agile software development.
Scrum Master	The person responsible for the Scrum process, making sure it is used correctly and maximizes its benefits.
Sprint	A time period (typically between 2 weeks and 1 month) in which development occurs on a set of backlog items that the team has committed to.
Sprint backlog	A list of tasks to be completed during the sprint.
Team	A cross-functional group of people responsible for managing itself to develop the product.

1 Introduction

Nokia Siemens Networks is a leading global enabler of communications services. The company's research and development teams are present in all of the world's technology hotspots. The major sites are Finland, Germany, China, India and the U.S. This thesis is made for the BSS RD SV Verification product line in Finland. The BSS RD SV Verification product line makes system integration testing for Charge @ Once business.

Testing and coding are an important part of the software development process. The targets of testing are to show how a product is working and to also confirm software quality. The main goal for testing is to find and remove defects in the software as early as possible and this way improve the product's quality. Poor quality of software gives a big number of problems and delays for a product. This means that good quality saves company's money. Different quality standards are helping in this work, but how the employees of a company adapt the software development process model to their daily work is a big question in the software development process.

The timetable of this thesis work was defined very tight. In the beginning it seemed that the time scale is feasible. During the year different changes happened in the organization, and therefore I had to find a new subject and work plan for the thesis. The first task of the plan was to specify what software development methods are and how they differ compared to one another. When methods were compared, it was searched for research result of differences of the plan-driven and agile methods. The second task was to read retrospective enquiries and write down, what problems there have been in sprints. Those problems are subjects for the interviews. The third task was to select interview groups out of organization employees and to discuss problems with them. The fourth task was to go through the interview material and write a summary report for a feedback session. The summary report presented what things motivate people in their work. The summary report was shown to the employees and people had a possibility to discuss their opinions during the meeting.

The basic target of the survey was to find, what positive things motivate people and what things have to be developed in the process. Those targets give the movement towards a better way to work. The fact that companies' tight business targets reflect to employees makes and the work pressure has become heavier. This is one reason why employees try to change their work habits. This thesis gives one point of view on what things are motivated in their work.

The purpose of the final thesis was to find development targets from the product development process and to investigate how to integrate the development ideas into daily work. The thesis consists of three parts. The second chapter describes research and software development methods and gives an exposition of the theory behind this thesis. The purpose of the chapter is to give the reader an understanding of the research and software development methods on a general level. The third chapter deals with the results of the interviews and inquiries, and the results are compared to the results obtained by Ketvell's in his thesis (Ketvell 2009).

2 Definition of research and software development methods

Before seeking for a method could begin, it had to be thought what was the investigation question for this work, therefore it was researched what tools are available for this work. On top of other things, it came up a question of how to set people to develop processes together. This makes it possible to approach the development target from the employees' point of view.

The purpose of this development task was to develop the existing process. In the process development, literary research materials were used to help developing the process, the employee's opinions of the development targets were collected and the progress of the targets was followed. This is a good opportunity to learn to understand other methods because Nokia Siemens Networks has used agile software development, especially Scrum method.

The first development task was to research what methods and techniques exist. Before starting the research some questions on the target of the work had to be asked:

- What are the techniques which give the best way to get answers?
- What kinds of materials are the most important answers for development work?
- How can it be used in the work?
- How could the analysis results be used in the development work?
- What kinds of analyse methods give the best result to work?

After a long consideration it was concluded to use enquiries or interviews. Enquiries and interviews belong to qualitative research methods. Qualitative research methods give a good possibility to search the opinion of people. It was decided to use interview as a method because with it you can collect research material for development work. (Hirsijärvi 2007, 119–161)

The interview method is chosen for the following reasons:

- As I want to collect research material by interviews the project participants are seen as subjects in the survey situation. Interviewees are given the possibility to discuss things freely. The interviewees bring in their own opinions and are active participants in the survey. This means that an interviewee has to tell things that are important in his/her work.
- It was taken into account that interviewees can give conflicting and inexact answers.
- During the interview, the interviewer can ask for clarification of the interviewee's answers.

During the survey there is a risk that respondents give answers which the interviewer is waiting for. However, it is necessary to have the interviews, because it is good to write down the employees' opinions of their work. Some of them have been working several years and they have work experience in different projects. As the process is developed, it is very important to take their opinions into consideration. (Järvenpää 2006, 1-45)

The theme interview was chosen as the interview type. The theme interview is a typical interview type, when the interviewer wants all the interviewees to know the interview subject, but does not want to give the questions beforehand. There were four interview groups and the interviewees were divided into those four groups according to their task description. One of the groups consisted of managers, and the three other groups consisted of implementation, testing, and laboratory personnel. The subject of the interview was to identify the good and the bad things in the project work. The respondents were meant to use their own experience in their answers.

Before interviews, retrospective enquiries were gone through. Those enquiries tell how sprints have gone and what problem fields there are. Black spot areas are collected into a problem list. The problem list shows what things to discuss with the interviewees. The list is used as a theme frame for interview, see figure 1.

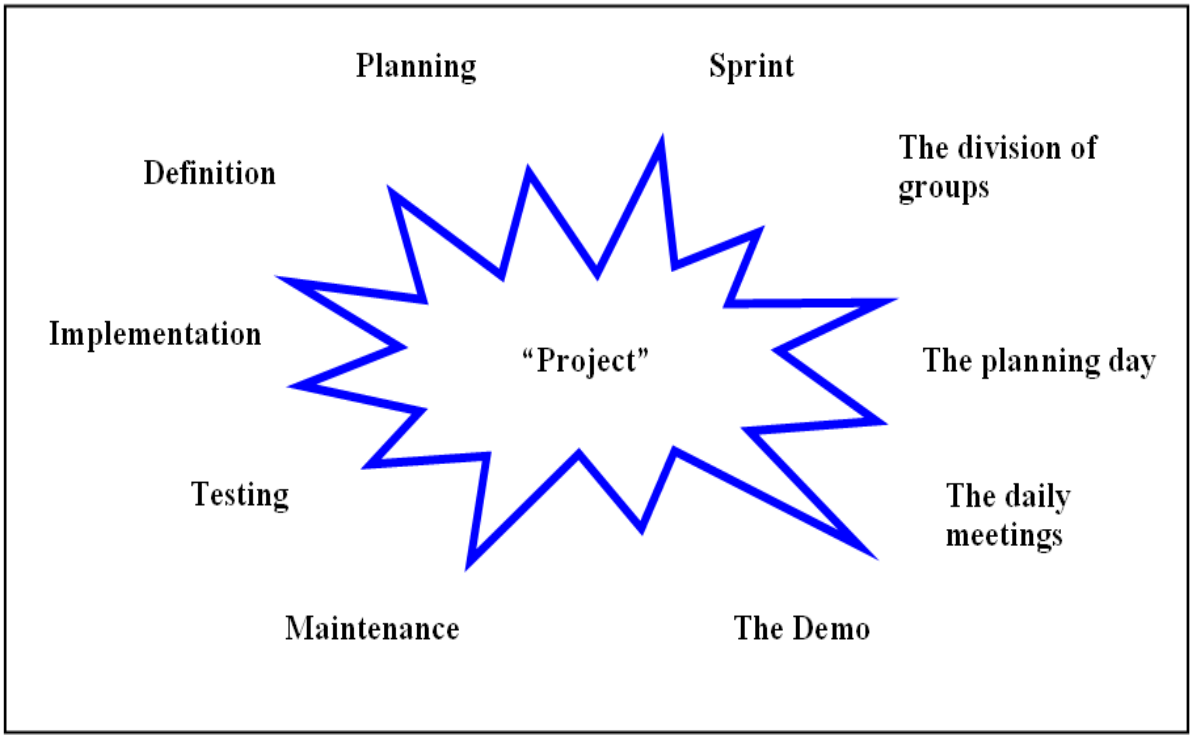


Figure 1 A theme frame for interview (appendice 1)

During the interview the theme frame was meant to lead the discussion to the right way. If the interviewees didn't follow a subject or they didn't say anything, the interviewer could take subjects out of the theme frame table.

One problem is how an interview result is verified. The inquiry method seems to be a good way. Inquiry can define any process that has the aim of augmenting knowledge, resolving doubt, or solving a problem. A theory of inquiry is an account of the various types of inquiry and a treatment of the ways that each type of inquiry achieves its aim. An inquiry is easy to make, because the number of people is small and they are near to the interviewer. The appendices include the original materials from the interview and the inquiry results. As the research method became clear, it was time to familiarize oneself with software development models. It was gone through what models are and how they differ from each other.

2. 1 Software engineering

As the research method was searched, the next task of the development work was to research what software methods exist. The main target was to find what kind of process cycles there are and what are the best parts in those processes.

Software engineering is a name for a work that produces computer programs and computer software. Ian Sommerville (Sommerville 2004, 9) has defined the software engineering term. Software engineering is an engineering discipline that includes all aspects of software production. Software engineers should adopt a systematic and organised approach to their work and use appropriate tools and techniques depending on the problem to be solved, the development constraints, and the resources available. (Sommerville 2004, 1-31)

The software development belongs to different quality systems that are used particularly in the documentation work. The software life cycle consists of different phases, which are for example requirement specification, software design, implementation, and testing as maintenance phases. Requirement phase includes an analysis of requirements for a hardware and software project. The existing customers and software and hardware requirements are taken into consideration. The Analysis isn’t committed to how those targets are realized. A customer requirement can be for example what software language should be used. System development phases are described in a general level in figure 2.

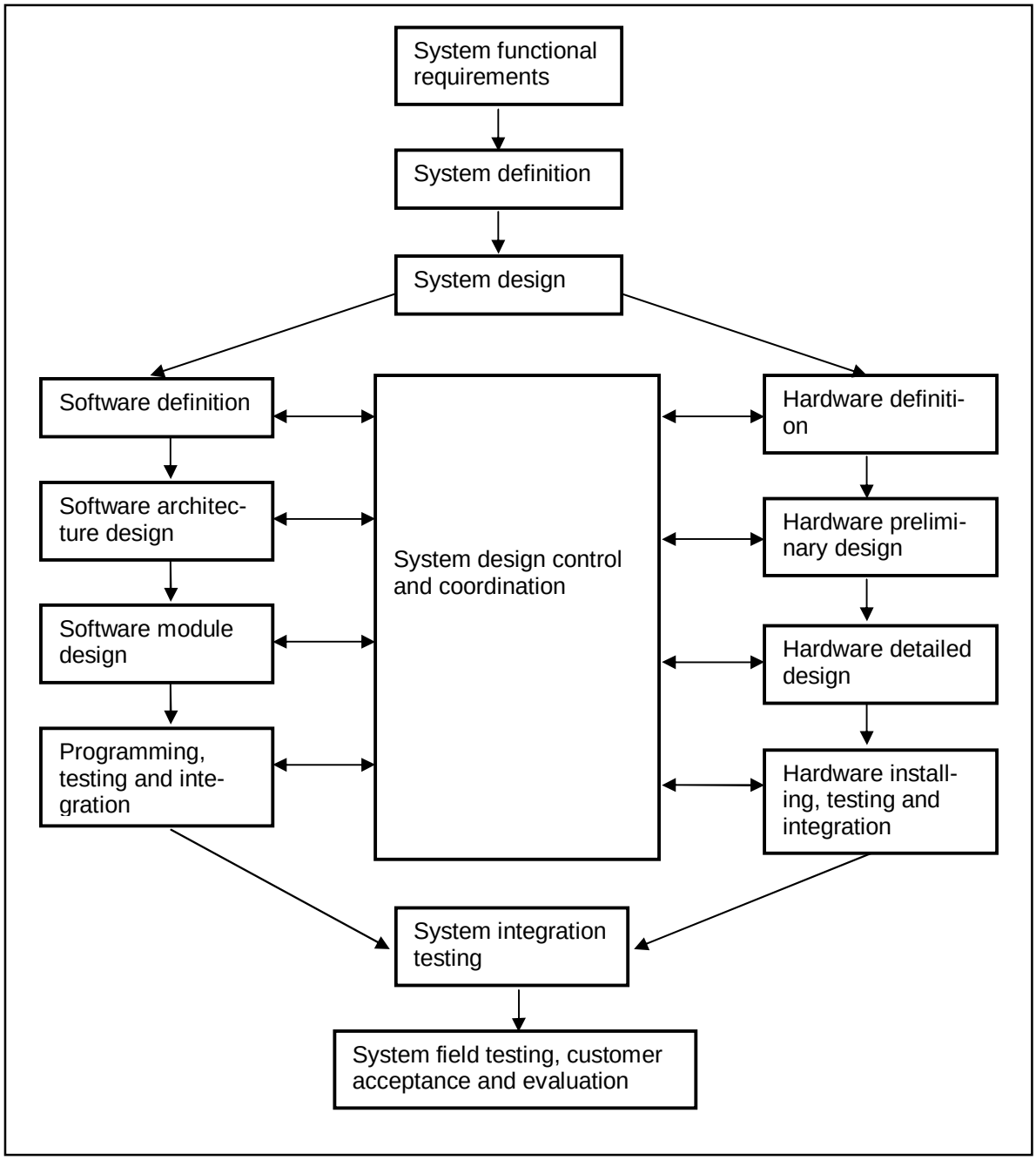


Figure 2 Phases of system development project (Modified from Haikala 2004, 49)

Software and hardware design phases investigate what functionalities are demanded and how those will be implemented in the product. The documents of the phases show the software and hardware technical architectures. A document describes for example software or hardware components, structures, architectures, and software languages. The programming phase contains coding of software by the chosen software

language. This phase's output is complete software, but there might be some functionality faults. The hardware installing phase is done at the same time. All hardware softwares are installed and unit elements are connected together. (Pol 2002, 19)

Myers (Myers 2004) has defined testing shortly. Testing is the process of executing a program with the intent of finding errors. The meaning of testing is to find all effective faults in the software, see how the software works in a faulty situation and how it works according to the requirement specification. Therefore testing is an extremely creative and intellectually challenging task. In the testing phase belongs the planning of test, testing, and documentation of the test. The test environment is prepared, test cases are run, and the result of test cases is analysed and a test report is written. (Myers 2004, 6)

Myers (Myers 2004, 14) has defined software testing principles into 9 categories:

1. A necessary part of a test case is a definition of the expected output or result.
2. A programmer should avoid attempting to test his or her own program.
3. A programming organization should not test its own programs.
4. Thoroughly inspect the results of each test.
5. Test cases must be written for input conditions that are invalid and unexpected, as well as for those that are valid and expected.
6. Examining a program to see if it does not do what it is supposed to do is only half the battle; the other half is seeing whether the program does what it is not supposed to do.
7. Avoid throwaway test cases unless the program is truly a throwaway program.
8. Do not plan a testing effort under the tacit assumption that no errors will be found.
9. The probability of the existence of more errors in a section of a program is proportional to the number of errors already found in that section.

All those principles belong to the testing process of software and hardware projects. Before the project is in its end, software and hardware are connected together. This system integration phase is important to be tested, because in a new system you have

to find interface errors. The major motive of testing is to get an almost faultless product to the market.

When programming and testing phases are done, the final phase can begin and the product is ready to be delivered. Before that, it has to be collected all software and hardware information for documentation. There are also information about the newest software and the license versions. After the customer delivery, the software enters the maintenance phase. It means actions that the customer needs to be satisfied with the software quality. The customer environment isn't always worked correctly although the software program is of very high quality and thus customers find faults in their system. The software supplier has to correct those faults. This means that every software correction have been tested before the new software version is sent to the customer. It is very important that the process is run in a controlled way.

2. 2 Software development models

The software engineering chapter described all phases that belong to the production of software. Software production can be divided roughly into two different development method groups. This final thesis introduces some of those methods. The first group describes plan-driven methods where tasks, milestones, requirements, and architecture designs are planned and documented very exactly. Software development is based on life cycle that begins from the requirement specification and ends in maintenance delivery. The idea of plan-driven development methods is linear and sequential. The second group is based on communication between people and the repeat of operation. These methods are called agile software development methods. Agile methods underline that fine working software is an indicator of progression. The goals of this method are decreased risks and a shared software development in short cycles. Documents aren't equally important than in plan-driven methods.

2. 2. 1 The waterfall model

Winston W. Royce proposed a new process model for the creation of software in 1970. The development of waterfall model is seen flowing equally downwards through the phases of requirements analysis, definition, design, implementation, integration, and maintenance phases (see figure 3). Software is often developed using this model. (Haikala 2004, 36)

The requirement phase defines the general requirements for a new system. Those requirements are called customer requirements that tell what a customer needs and wants from a new system. The customer requirements aren't committed to how the system is going to be implemented. The software definition phase collects functional, technical, and non-functional requirements for the software. The software design phase answers to the question;" How is the software going to be made". The software is designed in a technical level. This means that the technical architecture document defines permanent information, storage solutions, and the user interface.

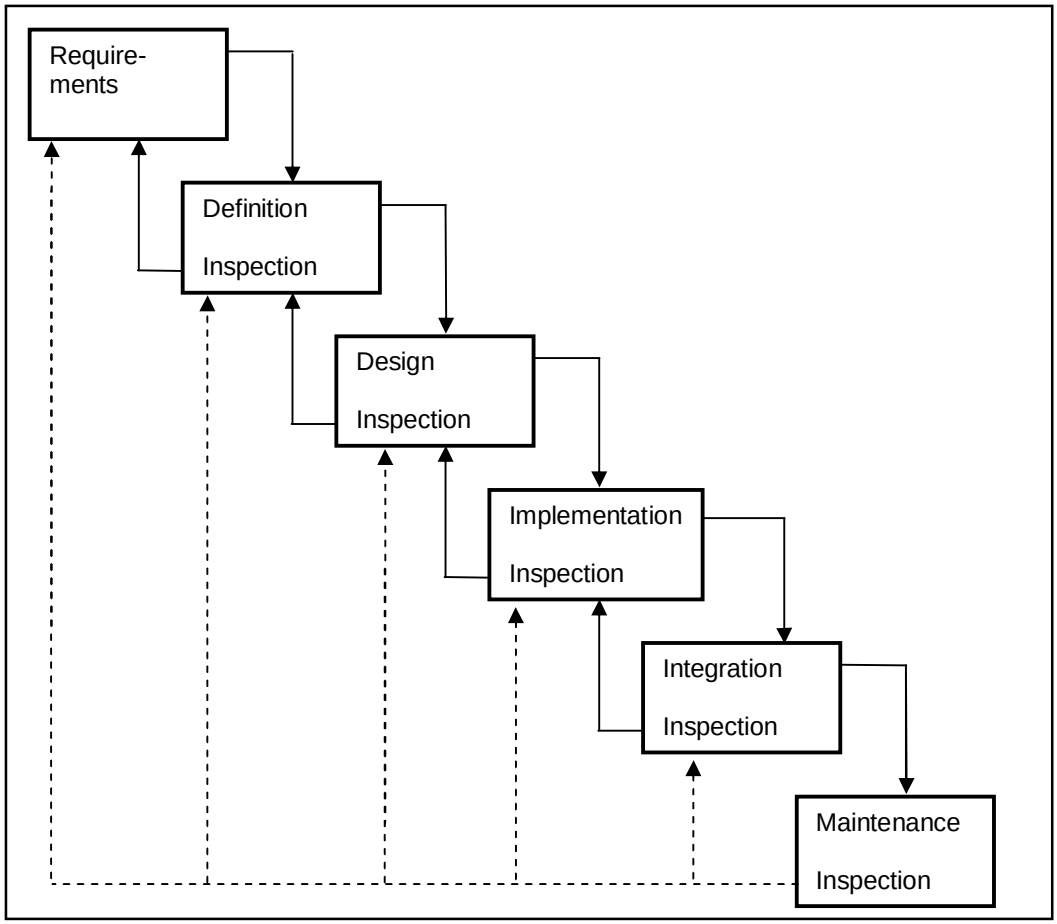


Figure 3 Waterfall model (Modified from Haikala 2004, 36)

In the implementation phase the software is coded in a chosen software language. This software code is based on requirements that are defined before the implementation phase. The integration phase connects software and hardware together. Integration testing is performed; its meaning is to find connection problems between both of the elements. The final phase is the maintenance phase where the product is delivered and customer corrections are made.

The biggest challenge of the waterfall model is that how can all the product tasks be completed at the same time. If one of the tasks is delayed, it affects the next phases and the product delivery time is moved forwards. If the product doesn't fit the customer's own view of the requirements, it means that the product development cycle has to be restarted. This brings more cost to the product, and therefore the software development process often includes iterative planning and execution that are

performed in small parts and repetitiously. The software is often developed incrementally, i.e. the software size is grown towards the final product. This development custom has created several iterative process models, such as Rational Unified Process, Spiral, and Agile.

2. 2. 2 The spiral model

Boehm (Boehm 1998, 61-72) has represented the spiral model in 1988. Its idea is based on the waterfall model. As the waterfall and spiral models' differencies are compared, it can be seen that the idea of the spiral mode is to control risks. If the spiral is compared to other process models, it is seen that risk control is an essential part of this model. A good thing is that it can be used all the time during the software life cycle. The purpose of a software project is to make a product using the cycle way, and every cycle is developed with iteration phases (see figure 4).

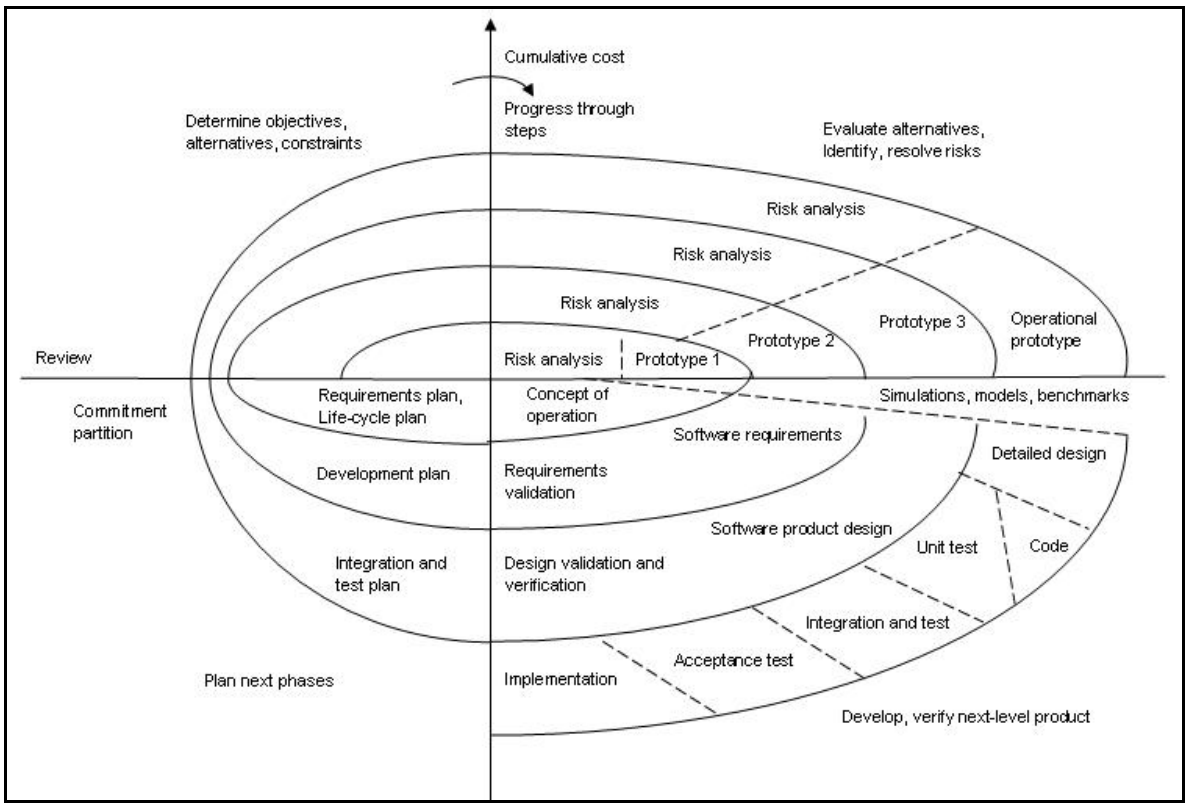


Figure 4 Spiral model of the software process (Boehm 1998, 64)

The first circle in the spiral describes that risks analysis is done for the prototype. The first version of the software is planned and executed and prototype 1 is ready. The second circle describes that operation and requirements plans are made for prototype 1. After that risks analysis and the next version of prototype are made. In the third cycle are collected software requirements and then those are validated. Then a development plan and again risk analysis are made for prototype 3. In the fourth cycle, software product design is done and it is also validated and verificated. After that, integration, test plan and risk analysis are done for the product. In the last cycle the final product to the customer is made. It can be said that every circle makes reformations in the software. The number of cycles isn't restricted and they tell how large the product is. The cost of a project can be roughly measured by how many and far the circles are from the centre. This information can be used when making a project plan.

2. 2. 3 IBM Rational Unified Process model

The Rational Unified Process aka RUP model is based on the spiral model. Software development has gone ahead several iterations where every iteration cycle constitutes its own development loop. Iteration means that the product development is grown incrementally from iteration to iteration. (Gornik. 2003)

The goal of the development is:

- The development teams work closely with customers.
- Every team member has access to the same knowledge base. It doesn't matter if the member is working with requirements, design or test.

The Rational Unified Process is based on nine core process and supporting work-flows. Those workflows show, what tasks have to be done in the different phases. The process is described in two dimensions. The horizontal axis shows time and dynamic aspect of the process. The vertical axis is described in terms of activities, workers and

workflows. One of the workflow tasks can make into four phases: inception, elaboration, construction, and transition (see figure 5). (Gornik. 2003, 3)

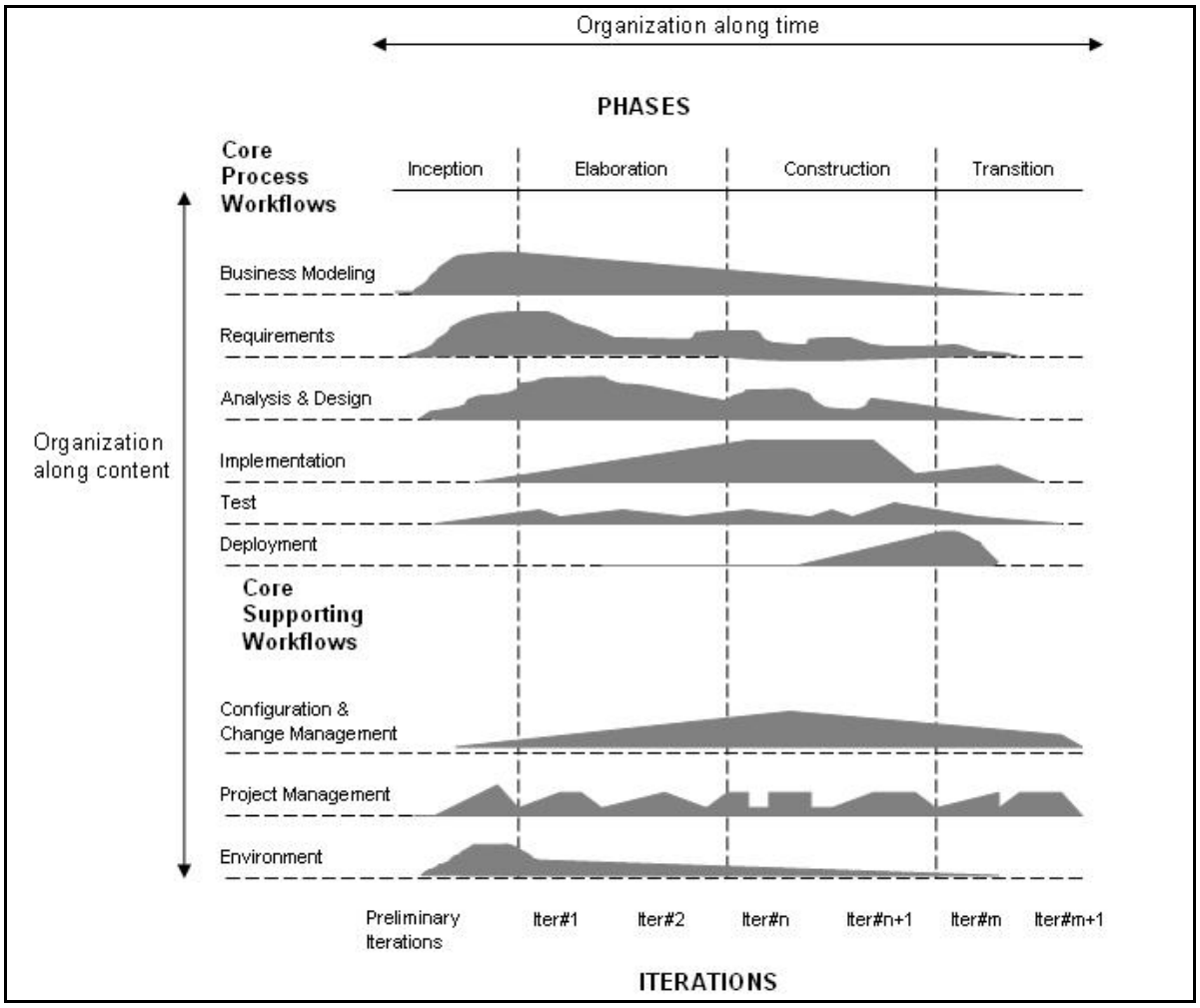


Figure 5 Structure of core workflows (Gornik. 2003, 3)

The inception phase establishes the business case for the system and predefines project targets. The phase involves identifying all use cases and including success criteria, assessment of risks, estimate of the needed resources, as a phase plan shows major milestone dates. Figure 6 shows iteration cycles for the project.

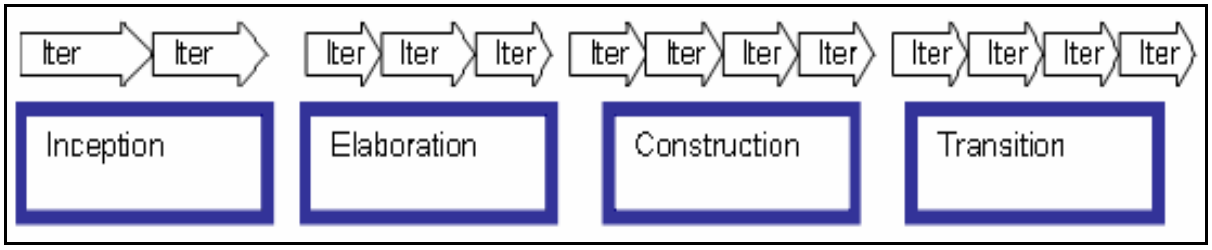


Figure 6 Iteration cycles of phases (Modified from Gornik. 2003, 4-6)

In the elaboration phase it is analyzed the risk elements of the project. A project plan is developed and decisions are made of the system architecture. A prototype is made that depends on the scope, size and risk of the project. The construction phase develops and integrates all remaining components and application features into the product. The software product is integrated on platforms. Every iteration cycle produces a new beta version of the product. The transition phase gives the product to the end user. The phase includes beta releases, bug fixes, trained and supported users as reacted to the user feedback. Information on how satisfied the users are with the product is collected. (Gornik. 2003, 4-6)

2. 2. 4 Agile

The development of agile software is based on four general principles of operation and basic values that every model follows. Those principles are represented by Agile Software Development Manifesto.

The Manifesto announces four principles which are:

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

(Beck, Beedle, Bennekum and Cockburn 2001)

The method is based on software development of iterative, incremental, testing automation, modification control, communicate and high teamwork. The method requires of the customer a strong participation in the project. The differencies of methods are how they separate the software life cycle and how exactly documents are written. Agile development methods are for example Crystal Methods, Scrum, Feature Driven Development (FDD), Adaptive Software Development (ASD) and Extreme Programming (XP). (Holopainen 2003, 1-17)

Every iteration cycle is a small software project which includes new requirement tasks. The cycle takes time from one to four weeks and it develops a new functionality to the product. In the every iteration can't add a new functionality to the product, but the software is always worked after the cycle. Before a new cycle has started, the project revalues its priorities and decides the content of the next iteration cycle. The purpose of the agile software development is to allocate resources where it best benefits the project. This way the software project is able to control risks. The first is to make things that are the most important and that put the functionalities into a risk. The efficiency of the method is speed, when the project gets direct feedback from customers. The feedback information gives possibilities to react fastly and to control changes in the product. The next chapters discuss the Scrum method because Nokia Siemens Networks has used it in its projects. (Leffingwell 2007, 115-133)

Scrum has been developed for managing the systems development process. It hasn't been defined any specific software development techniques for the implementation phase. Scrum concentrates on how team members should work in order to produce the system flexibly in a constantly changing environment. The product development involves several environmental and technical variables, for example requirements, time frame, resources and technology which are changed during the process. This means that the development process responds to changes, which unpredictable, complex and requiring flexibility of the system development process is demanding. The scrum process includes pregame, development and postgame phases, see figure 7. (Schwaber 2003)

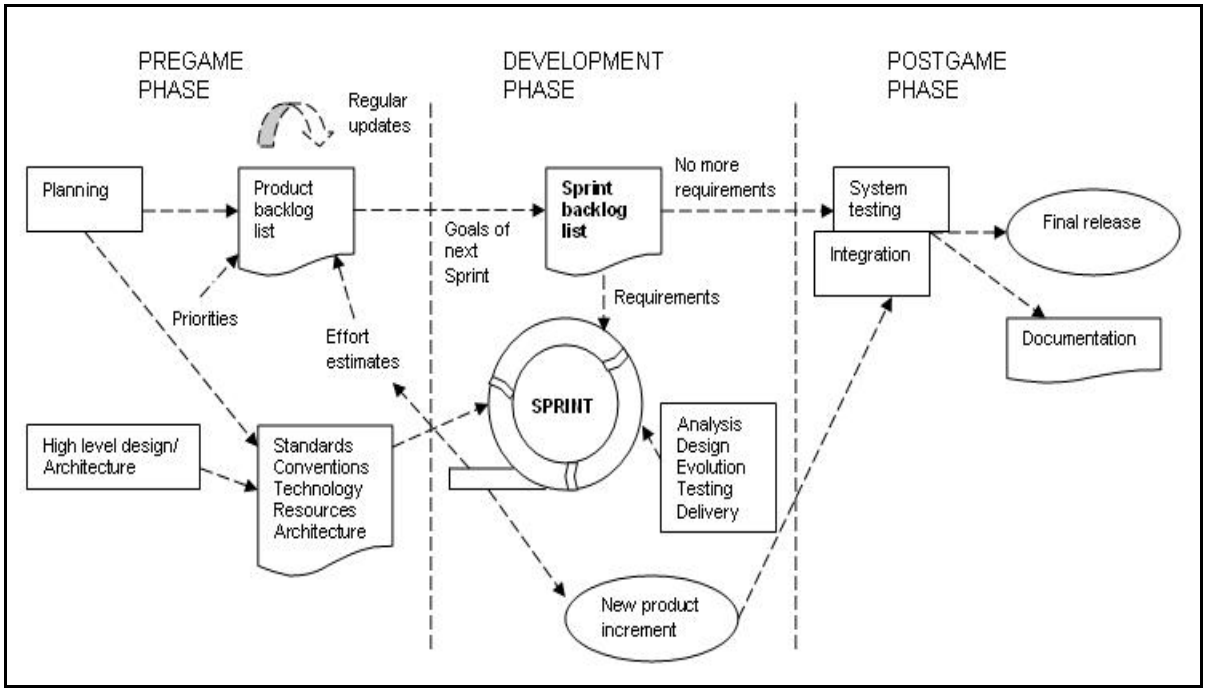


Figure 7 A Scrum process (Abrahamsson 2002, 28)

The pregame phase includes planning and high level design or architecture phases. Planning includes the definition of the system developed. A product backlog list is created containing all requirements. The requirements are from customer, sales, marketing division and customer support and software developers. The requirement list is prioritized for implementation. The product backlog list is constantly updated with new requirements and with more accurate estimations and new priority orders. Planning includes also the definition of a project team, tools and other resources, risk assessment and controlling issues, training needs and verification management approval. The each iteration updates the product backlog reviewed by teams. (Schwaber 2002, 72)

High level design includes system architecture which is planned based on current items in the product backlog. If the system has changed, those changes are needed for implementing the backlog items. Items are identified along with the problems they may cause. A design review meeting is held to go over the proposals for the implementation, and decisions are made on basis of this review.

The development phase is handled as a black box where the unpredictable is expected. The different environmental and technical variables are identified in the pregame phase, which may be changed during the process. They are observed and controlled through various scrum practices during the sprint of the development phase. The product development is made in the sprint. Sprints are iteratived cycles where the functionality is developed or enhanced to produce new increments. Sprint backlog is consisted of task that team has been devised for a sprint. Those tasks are worked to transform selected product backlog into the sprint goal. During the sprint, team members make analysis, design, evolution, testing and delivery of the product. At the end of development phase new product is given to integration and a new effort estimate to backlog items is created. (Boehm 2002, 64-69)

The postgame phase includes the closure of release. This phase is entered when an agreement has been made that the environmental variables such as the requirements are completed. The system is now ready for the release and preparation for the postgame phase is done, including the tasks such as the integration, system testing and documentation of the product. (Boehm 2002, 64-69)

2. 3 Software development by Agile or plan-driven methods

Agile and plan-driven software development methods are opposite to each other. Both sides think that theirs is the best way to make development work. When we compare plan-driven methods with Agile, we can see the following differences.

Agile methods are focused more to the planning process than the resulting documentation. This means that these methods often appear fewer plans oriented than they really are. Plan-driven methods have focused on major milestones and their overall content, rather than on micro milestones locked into an ironbound contract. Traditional methods like water fall is using extensive planning, codified processes, and rigorous reuse to make development an efficient and predictable activity that gradually matures toward perfection. (Boehm 2002, 64-69)

If agile methods are compared to the traditional waterfall process, the process has the following advantages:

- Risks are mitigated earlier.
- Change is more manageable.
- Iterative process has higher level of reuse.
- The project team can learn along the way.
- Better overall quality.

The main difference is that agile methods get much of their agility by relying on the indirect knowledge included in the team, rather than writing the knowledge down in plans. When team knowledge is at an adequate level, there is also the risk that the team will make irrecoverable architectural mistakes because of unrecognized shortfalls in its indirect knowledge. Plan-driven methods reduce this risk by investing in lifecycle architectures and plans and using these to facilitate external expert reviews. Boehm (Boehm 2002) has compared agile and plan-driven methods see table 1.

Table 1 Agile is compared to plan-driven methods (Boehm 2002, 68)

Home-ground area	**Agile methods**	**Plan-driven methods**
Developers	Agile, knowledgeable, collocated and collaborative	Plan-oriented; adequate skills; access to external knowledge
Customers	Dedicated, knowledgeable, collocated, collaborative, representative and empowered	Access to knowledgeable, collaborative, representative and empowered customers
Requirements	Largely emergent; rapid change	Knowable early; largely stable
Architecture	Designed for current requirements	Designed for current and foreseeable requirements
Refactoring	Inexpensive	Expensive
Size	Smaller teams and products	Larger teams and products
Primary objective	Rapid value	High assurance

Briefly, agile fits best in small and middle size software products. Agile methods worst side is that employees have to communicate with each other. This is very difficult, if team members are located round the world far from others or the used language isn't the same or members aren't familiar with each other when they don't know their habits. The implementation of the product is difficult, if requirements aren't clear. Team members don't know what they have to do. Working is also difficult if product leaders have diverging views of a feasible product and how it should be done.

The inconvenience of plan-driven methods is to make modification to product during development of product life. Every change returns software development to beginning and this way is very expensive to the company. In plan-driven methods it is important that all have the same view of the product and requirements are very clear. If in the organization there are different views of the product, they make implementations which aren't compatible with each other. If incompatible software is found before the integration phase, whole process has to restart.

As the work process develops, it is recognised different phases in the process. Therefore it is concentrated on workings and people interaction that all components evolve at the same time and interactivity. An objective is to take into consideration all the factors in the development of the working process, which have an influence to fluency of working processes.

Developing a work process is based on an in depth studies or analysises components work, its interdependencies and needs of developing. In all work can be dissected basically target or result of work (what purpose is envoked), target of work (what is being done, handled or changed) and worker (who is working). When reviewing working process, it is investigated at the same time what are the benefits and weaknesses to a development target. What are development idea effects more widely for a work process? It is very useful, if people see how to make work in the development process and that they take in discussion things that improve working conditions. This is the reason why the next target of this thesis is to make interviews and collect employees' opinion on their working.

3 Results of interviews and inquiries

Before starting to analyze the results of the interviews and inquiries it is good to keep in mind the targets of the development tasks. The purpose of the final thesis was to find development targets from the product development process and to investigate how to integrate the development ideas into daily work. The next chapter describes the Scrum development method as Nokia Siemens Networks has used the Scrum method in project work.

3. 1 The product development uses Scrum

If the software development uses Scrum, it might get some people very confused, especially people, who are used to the waterfall method. An example makes it clear how this is working in real life. The product making begins in the pregame phase. This phase means that the product design begins. The software making has named architects and product management people participate in the product design. Those people start the high level design of the product planning. The product management has brought to the planning meeting those requirements that customers, sale and marketing agents are waiting from the new product. The architects discuss with product management how realistic those requirements are and are those possible to be realized with the current technology. Feasible features are gone through in several meetings and set requirements are cut to smaller implementation units.

Implementation units are prioritized and sorted into an implementation order so that every implementation part is made within its own schedule. As requirements are put into an order, it has to be noticed that the product has to be in a customer delivery state as soon as possible. All requirements aren't in the first delivery version, but product works according to the high level design. As all the requirements of a product are gone through, they are moved to the product backlog list. In the pregame phase it

is discussed what tools and resources are needed for the purpose of product implementation.

Next, the product is moved to the development phase. A development team is collected for the implementation of the product. The team includes software coding and testing people. The development of product continues in a planning day. The development team goes through the requirements on the product backlog list during the planning day and the content of requirements is discussed with architects and product management people. The purpose of the discussion is to make clear the content of requirements and consider how feasible those requirements are and what is the real delivery schedule. The development team collects requirements according to the priority from the product backlog to their own sprint backlog list. The sprint backlog list only includes requirements, which team members think to implement during a sprint. The requirements are cut to smaller units. If the development team is big enough, it can be divided and the splitted teams take parts of whole requirements for development. Parts of requirements can return back to the product backlog list, because those requirements need more extensive settling by architects and product management people.

After the planning day, sprint teams have a meeting by themselves and they vote a scrum master. The scrum master represents team members, take team’s problems forward and arrange daily meetings. During the first day meeting members choose tasks which they are going to make during the sprint from the sprint backlog list. The team members judge about their own tasks that how long it will take for the task to be done and write down estimates in the backlog list. During the sprint, the software development people make software and at the same time the testing people is testing software for that the faulty situations might be corrected before the end of the sprint.

The end of sprint is a demo that shows all the people where we are with the product. Every team shows what they have made during the sprint. In the next planning day it is discussed again what are the next tasks from the product backlog list and there are also those requirements from the last sprint that are needed for more information.

Again the development team collects requirements from the product backlog to their own sprint backlog list. These cycles go on until all requirements are done.

When the product is in a delivery state that the first customer version can be integrated, it is the beginning of the postgame phase. This means that for the software it is made a new product increment for the customer delivery. The customer delivery software includes requirements which make a whole feature. Unfinished features aren't included in the customer delivery version. In the postgame phase the product environment is combined into different software systems, i.e. software and hardware are integrated together. After that, on the new product environment system testing is made where it is checked that the whole product is working (how features are set). Testing results are taken to the pregame phase, so that the architects and product management get last moment information of the product functionality. The Information is used to develop new functionalities to make old features better.

As the product is integrated and the system tested, documents for customers and used internally in the company are written. If product development isn't continuous, this is the final version of the product. Often development is continued and the product teams get a new feature to be developed. This is the reason why postgame phase is ongoing and at the end of integration it is waited a new product increment version for the customer. At a general level it can be said that the purpose of the scrum is to give a frame to the project work. Therefore every company and organization adapt directions the best way they can see their own research and development work.

3. 2 The analysis of interview results

As software development methods were looked into, it was time to make interviews to have research results. After interviews it was researched what kind of analysis methods exist and what is fit for use in the research work of the interview material. During the sprints were taken development ideas of the working process. After about a year, it was time to look how well development targets had gone. The best way to

calculate development targets was to make inquiries to employees.

A factor analysis was seen the best way to analyze the interview results. Factor analyses can be differentiated in two ways. The first is an explorative factor analysis which is based on a theory research. This means that the researcher investigates the research material beforehand and doesn't limited those values or characters. The second is a confirmatory factor analysis. This model is based on the assumation that the researcher already has a theoretical understanding and knowledge about what kind of results the researcher is waiting from the research material. The confirmatory analysis gives a number to variable factors. Every factor has got a number and those numbers are calculated together. The purpose of the analysis is to give statistical parameters to the researcher who has to decide whether his expectation is right or not. This analysis of interview material is based on the confirmatory factor analysis. (FSD 2007)

During the interview it is written down to a Power Point slide what things group has seen important to pick up in the meeting. Those were separated as positive and negative things. At the end of the interview items were gone through and filled to complete the sentences. The purpose of the survey was to make at the same time the transcription of the interview material. When all interviews were finished, it was started to analyze the material using the factor analysis. The first was collected from interview material to a keyword list which is described a thing. After that a list is arranged as keywords. If the same keyword got more than one point so its number of word is increased numerically. Then it is listed keywords according to the value of number.

At the end of analysis was arranged a final meeting. This meeting's purpose was to go through what things interviewees considered the most important in their work. Before the meeting, it was collected seven most important things to a Power Point slide. At the meeting it was discussed commonly the summary slide items and everyone said their opinion on what they feel about those things. In the meeting took part people which weren't participating in the interviews. The purpose of the feedback meeting was commit the people to see what things are motivating them and at the same time it was

discussed with managers how this can be executed.

3. 3 The interview results

Interviews are given to information from it that how people are seen how a project ought to go through. One of reason why group interview is chosen was that it is given to fast result and participants could have been deepened own answers and others opinions by during interview. It is good thing because a group can be helped issues for example in memory things. The end of interviews was collected all material together and it was analyzed final result. As answers were analyzed, it is given to them points. Then there was collected seven most of important things which motivate people. Those seven things describe what it is common to all the interview groups. The following listing indicates the result of the opinion of workers, on how they see what things are motivating in their work.

The most motivating things were:

- Work is done in the smoothest and fastest way when team members are close to each other. Design, coding and testing are made in the same sprint and at the same time. Therefore requirements are made fast ready. Team members will be able to respond to fast changes and then it is sprung up “high working spirit”.
- When people work in the same group and near each other, they don’t feel that they are alone with their problems. Team spirit helps them, when they resolve their work problems.
- People can select such tasks from the backlog with which they can increase their competence area.
- Information is moving inside the group at the daily meetings.
- People see at the Demo how a thing is really working in the product. This thing is motivating in the work.
- Scrum master has a better opinion of the work if he is participating in the

work.

- People are motivated if they can install software flexibly and fast although server is like “empty table”. (Appendice 2, 1/3)

After that it was collected the most important things, which aren’t motivating people and what are block or decelerators in their works. Analysis was difficult to make, because there is a wide divergence of opinion on this issue. The opinion is depending on how it is given. Issues can be separated to six different categories which are team, meeting, demo, scrum master or manager, documentation and working. The most negative things were:

Team working

- Teams’ flow of information isn’t working, if teams are working in different cities.
- How tasks should be done, when teams have different view from things.
- As workers are too few, competence area is concentrated on one worker. This is a problem, if somebody is sick or leaving from company.
- If a team is too big, the team doesn’t care, if somebody is doing totally different works.
- As tasks changed, teams aren’t separated to small groups, but they continue making work in the same group as before. (Appendice 2, 2/3)

Meetings

- Meetings take too much time from real working. This isn’t touched to daily meetings.
- In daily meetings they have to be careful that it doesn’t come a hunting of profitability for tasks. It can be that the pressure of work is coming too heavy. Positive attitude has to be kept in the work. (Appendice 2, 2/3)

Demonstrations

- Preparing the demo is taking too much time from the sprint.
- The content of demo isn’t defined enough; the demo is trying to show the

whole functionality of the product. Sometimes the show is too simple. (Appendice 2, 2/3)

Scrum master or manager works

- Workload estimates are too optimistic for tasks considering both workers and managers.
- Planning day has to be clear and effective. Preliminary work before the planning day has to be done and an also new requirement for the next sprint has to be thought of.
- The content of maintenance work is changed during the sprint. E.g. it was said in the afternoon to do a task before evening, because it has to be sent to the customer that evening. An interruption of work was then ready for new requirements (time scale).
- The content of the sprint is planned too tight and therefore there wasn't enough time for competence transfer within the group. (Appendice 2, 3/3)

Documentation level

- There was poverty in the documentation at the beginning of the work. The code does not show how requirements have to work.
- The level of requirements was too high. The requirements don't always describe what is wanted by them. (Appendice 2, 3/3)

General working

- Test cases aren't held in a dynamic environment. The functionality of cases is broken time after time.
- The employees don't know what software packages are installed to customers. Software packages lists are late compared to the real situation.
- In the software packages there were too many different customer branches. Maintenance work is difficult to make if there are too many branches of customers. (Appendice 2, 3/3)

As results were analyzed afterwards, it was reviewed how the answers are allocating to survey questions. In survey it was seeked answers to questions; what are the weaknesses to software development method, what are the benefits to customers and employees, if the organization is using the method and how can teams improve? After the feedback meeting, teams take problem solution as a part of their sprint work. Before that it has to be thought what kind of an action time table is in this development task. This means that it was tried to increase the follow-up of the development process and on the other hand development tasks don't come too late.

3. 4 Inquiry result

Before the inquiries were made, it was decided what kind of question type is used. A questionnaire is a research instrument consisting of a series of questions and other prompts for the purpose of gathering information from respondents. A questionnaire consists of a number of questions that the respondent has to answer in a set format. A distinction is made between open-ended and closed-ended questions. An open-ended question asks the respondent to formulate his own answer, whereas a closed-ended question has the respondent pick an answer from a given number of options. The response options for a closed-ended question should be exhaustive and mutually exclusive.

Types of questions are:

- Contingency questions: A question that is answered only if the respondent gives a particular response to a previous question. This avoids asking questions of people that do not apply to them.
- Matrix questions: Identical response categories are assigned to multiple questions. The questions are placed one under the other, forming a matrix with response categories along the top and a list of questions down the side. This is an efficient use of page space and respondents' time.
- Closed ended questions: Respondents' answers are limited to a fixed set of responses. Most scales are closed ended. Other types of closed ended

questions include:

- Yes/no questions: The respondent answers with a “yes” or a “no”.
- Multiple choices: The respondent has several options from which to choose.
- Scaled questions: Responses are graded on a continuum (example: rate the appearance of the product on a scale from 1 to 10, with 10 being the most preferred appearance).

- Open ended questions: No options or predefined categories are suggested. The respondent supplies their own answer without being constrained by a fixed set of possible responses. Examples of types of open ended questions include:
- Completely unstructured: For example, “What is your opinion of questionnaires?”
- Word association: Words are presented and the respondent mentions the first word that comes to mind.
- Sentence completion: Respondents complete an incomplete sentence.
- Story completion: Respondents complete an incomplete story.
- Picture completion: Respondents fill in an empty conversation balloon.
- Thematic Apperception Test: Respondents explain a picture or make up a story about what they think is happening in the picture. (Hirsijärvi 2007, 186–212)

For the inquiry method is chosen questionnaire of scaled questions type, because researcher can drew up the list of questions and answers ready for use. In that case, analysis is much easier to do.

The inquiry was made to people who are managers, implement, testing and laboratory people. The answers came fourteen and everybody who participates in the inquiry responded to the inquiry. This inquiry used a Likert item type where a questionnaire was six-level format (appendice 3). A Likert item is simply a statement which the respondent is asked to evaluate according to any kind of subjective or objective criteria.

Table 2 the results of the inquiry (appendice 4)

Statements:	Strongly agree	Agree	Neither agree or disagree	Disagree	Strongly disagree	No opinion
The information is working well between different teams.		3	1	8	2	
Learning a new thing is easier than a year ago.		5	5	3		1
Competence transfer is easier than a year ago.		5	4	3	1	1
The number of meetings is suitable for working hours.	2	7		3	2	
The team's size is good compared to work tasks.	1	8	1	4		
Preparing the demo is taken into account well in the sprint timetable.		1	4	7	2	
Content of the demos are very clear and understandable.		2	1	5	6	
The planning days are effective and well-prepared.		1		4	9	
The documentation is extensive enough.		1	3	6	3	1
The requirements' level is clear and understandable in the product backlog.			2	12		
The test cases are working well between different sprints.		2	2	4	2	4
In the testing or development environments used software versions are well known.		5	2	5		2

Table 2 describes which development tasks went well and which need improvement. As a result, it can be stated that more attention needs to be paid to the interaction between teams. This is very important as employees are working in different towns and countries. When people work at several sites that, it is good to keep in mind that the flow of information is not working as well as when everybody work at the same site. This affects the closing of issues, which takes more time than normally. As people are not at the same site, they can not organize extempore workshops for problem solving or other information sharing meetings. This is one of the reasons why it is

difficult to share a common opinion with all the project teams, and it can be said that it can be impossible.

The next problem area is the demo. It seems that in the sprint planning, it is not taken into account how much time preparing the demo can take. It seems that the content of the demo is not always very clear and understandable to the employees. The third problem was the planning days. The statement value tells that the employees do not think that the planning day is effective. It can be read between the lines that they are feeling that those meetings are a waste of their working time. In my opinion, both problems could get fixed trough a refresher course about the Scrum working method. All teams' members and people who are working in the project should participate in the course. In the course, people would go through the working methods and try to decide common rules for the project work. This would commit both old and new employees to the work and targets of the project.

The fourth problem is that the documentation is not good enough. This should be taken up as one task in the sprint backlog, and all project people must be committed to do this task. The refresher course gives clear target why documents quality level must be good. This course gives to all employees same way of thinking from content of documents.

The fifth problem was that the requirements' level is not clear and understandable in the product backlog. It is very understandable that some of the requirements are raw tasks and their content is not clear. One of the reasons can be that the people are at different sites and information sharing is not very fluent. This puts pressure on the preliminary work for the planning day.

The sixth problem was that test cases were not working between different sprints. This causes new work tasks in the sprint, and it takes time to work a new functionality to the product. It is very difficult to avoid the problem because the work is done in the agile method and the product is made in the iterative way.

3. 5 Benchmarking for the research

The inquiry result is good compared to other company inquiry result, because it can be seen how different companies are working and it can be learned new things to one's own working process. To successfully undertake benchmarking, one company must be willing to learn from another company. Benchmarking is the process of comparing and measuring an organization's operations or its internal processes against those of a best in class performer from inside or outside its industry. Benchmarking involves finding the secrets for success for any given function or process so that a company can learn from the information and improve on it.

In this thesis, I compare my own research results to Antti Ketvell's (Ketvell 2009) results of his thesis about the position of human resources in a changing working environment a case study of implementation of SAP R/3 enterprise resource planning program in Citymarket Lt. I chose from the thesis results the benchmarking, because Ketvell's research approached to a research problem from employees' point of view and myself had the same point of view to my research result. Antti Ketvell has made a research on how the new operation control system's commissioning interacts to the employees of Citymarket Lt's transition process. Ketvell's research tells how employees regarded changes and how transition process succeeded. Ketvell considered his research results to the subject of theory. He tells that his transition process of research subject is a starting factor for old data processing systems, used software disassembled, seeking for cost efficiency and EU brought more external competitors. My research subject needs to develop the project inside process. The idea of development task was to seek subjects from the used process and correct them. (Ketvell 2009, 11-53)

As I made interview or inquiry, I noticed that employees felt that their opinion was the least taken into account when made changes to work processes. Ketvell (Ketvell 2009, 11-53) also noticed in his inquiry the same thing and therefore employees had it difficult to commit themselves to working on a new process. Although in my research subject, employees used the agile method, so they felt that their opportunity to

contribute to their own working is narrowed to minor. Employees felt frustrated, because instructions on how the product should be made came from elsewhere. People who decided those instructions don't necessarily internalize in the agile method and they are thinking strongly on the waterfall model way. This came out when I discussed with Finnish or German employees. It is usual that different work cultures and methods aren't settling down to other cultures. It is obvious that then dissatisfactions for the own work arises.

Employees' point of view has to be taken also into notice, when it is begun to make changes for process. It is obvious that how well it is planned shows how it is going to follow the plan through different process phases. Generally it is known that employees' effectiveness and self-respect keep steady, if work situation is unchangeable. Then it is better to make small changes to process and go wanted to the way. If there are made big changes in the work or the working environment, the current balance is shake and this affects the employees' work. What I noticed is that in transition states employees' self-respect goes down and same time goes down also work effectiveness. Sometimes effectiveness to process might be over a year. Different people feel changes in a different way although changes would be good things; people need time to dissolve the idea. At transition process, a part of the people reacted to changes with denial, defend, acceptance or embracement of the idea. When time is gone, people embrace the idea and can adapt the new idea for the old work model. I see that how well new work model is accepted, it depends on how employees are committed to new changes. At a general level it can be said that if the company arranges training programs and coaching of changes, it is easier to take the new process method into use. I see that continuous training and instructions in the organization make sure that change can be concretized and the goal of changes is clear to the employees. This is very important, because this way it can be made sure that he organization attain to goal of changes.

Ketvell (Ketvell 2009, 11-53) was asking in his inquiry how well it was noticed, the employees starting level for training. Most of employees answered that their computer knowledge isn't taken into notice. This is telling us how much in the higher level

management there are troubles listening to employees' point of view, how their work knowledge should develop and in what way. It is impossible to think that both parties can discuss openly together how work should intensify, if information flow goes to one way. It is obvious that both know what benefits the company, but the problem is that both sides have different opinions on how to proceed. I sensed that sometimes in change situations decision-making and flow of information happen still in old management systems from up to down. In modern day manager ship training it is taught that change situations at communications and decision-makings should notice that in the whole organization there is potential information and wisdom. I would see that the success of a change is management's concrete support and adequate resources given to the process. In accordance with my experience it would be important the clear communication with employees, that you know your colleagues otherwise than in the work role, confidence in colleagues, strong confidence in the manager, what kind of work experience the managers have, how clear is organization or team structure and how systematic changes execute. I see that those things give stability in transient states.

As I compare my research to Ketvell's (Ketvell 2009, 11-53), I see that one important object to succeed is who is participating in the planning. Improvement ideas were tried to bring into daily workday, in my research changes left from down to up level and high level managers didn't know or they couldn't take notice of those proposals for improvement. This is the reason why some changes were impossible to be realized in the sprint work and those subjects got negative feedback from inquiry. Ketvell's (Ketvell 2009, 11-53) research changes came from high level to down and therefore they didn't take employees in the planning. This gave negative feedback and results saw that staff opinion was missing in the planning. Employees were a close part of the execution phase, but they haven't been given the possibility to affect it. It can be challenging how to engage continuous interest in execution changes if all the time employees feel that they struggle with the fact that nobody notices their opinions.

Ketvell's (Ketvell 2009, 11-53) collected answers brought out that managers didn't know how much their subordinates had responsibilities. Managers haven't also known

that departments needed more people to do the work. Some answers told that continuous hurry had an effect to the quality of customer service. I also noticed the same thing in my work. It is very difficult to give the best service to customer if you have too much work going on. You are limited how many work tasks you can do at the same time. If tasks are too many, your concentration blows up in your face. In a wider perspective, is it can noticed that profit needs can drive employees to burn out. This isn't good for the company, because it loses money and competence. It is the reason why the right value of human resource estimates is an important value to company.

Internal communication is a very important part of manager duties. Good communication gives tools to employees' motivation and commitment to work tasks and company. I see that internal communication is one of the subjects which have an effect to the company's external image. Therefore it should be remembered when making plans for changes communication. In the communication plan it is taken into notice how employees and interest groups get understanding of the change and get all the information and know-how on working for the change in the organization. As communication plan is executed, it is remembered that the information must be logical and correct. If it is inconsistent, it gives possibilities to rumors and it dilutes success in changes. It is general that management of company explains the less of communication with the fact they can't talk about unfinished things. I have seen that although communication things are unfinished or negative things, it has to tell truthfully to the employees. Employees feel like the information gives safe feelings and cut off extra rumors. Negative rumors move so easy e.g. if the employees are situated different places. In long distance, co-operation is important for that both parties can trust in each other, because without that working is difficult. In this situation a deciding factor is that the information is moving regularly and the participants understand each other. If information isn't moving inside the company, work community change's more concrete effect is uncertainty about future on personal and organizational level. In a situation like this people are taking into consideration theirs own destiny and leave the teamwork as the second thing.

By planning and education it has been tried to develop the work compatible with peo-

ple. But always the time scale for the project is very tight and it is easy to think that people can fix work organizing deficiencies and processes working in the same time than they are doing regular work. Although education is necessary in working life, it has been criticized a lot of during last years. It is not very profitably that one person of the work community has leaved to study for co-operation development. As a person came back from studying and he/she brings new things to team, it is usually that he has returned to old routines and attitudes quite fast. Therefore it is important, that the whole team is committed into taking new manners for daily working.

Every time when it is made changes into the process, it is like iterative development work, because development of process hasn’t become ready. When change process plans are made, it must be remembered that employees go through denial, defend, accept, assume and expansion phases before changes are taken into work. It is very challenging to make process changes, because people felt and reacted in a different way to those changes. Therefore in this research work, employees take development tasks at the same time in the sprint work.

4 Summary

It is normal that in a company it isn't wanted to make changes in their process habits. The reasons of change come usually from outside the company. The biggest outside reasons are the change of customer needs. The company has to respond to customers' needs for the company to keep market share, their customers and to overtake satisfying profits. Internal changes are a consequence of employees, a factor of production or cash flow changing. The company needs to prepare for upcoming changes in many different ways. One way is to make changes to the product process and therefore this thesis meaning was bringing out one view from employees. The purposes of the development work at the beginning was to develop the existing process and improve the working of the process and are the same time get better productivity to the product. The company's internal changes brought additional inconvenience to the development during the project time. Those changes complicated the realization of the development work. The development work continued although the product changed several times. It is seen that those actions make a better way to existing the process.

It does not have to be a specialist to understand that the IT field has quality problems. There were projects which were going well and perhaps even those that were complete earlier than the project milestone. Then there were lots of projects that did not progress as in the beginning it was planned and this brings development pressure in the project. At first in the traditional software development model, the product is defined and then the code is being made and after that it is tested. Therefore it is very difficult to produce software effectively in a moving and complicate technology environment and therefore project timetables become delayed. There is also the risk that the product is already partly of or fully outdated for customers' requirements. Therefore the traditional waterfall mode is not suitable for new product development, because almost always in the development of a new product's initial phase, the product's essential requirements and operating environment are not known very well. It is the risk that in the beginning the written requirements are being taken tightly and that leads to bad situations. In the real life, product requirements move in conjunction with the project and become clear and this gives pressure to the product process.

As the organization wants to move from the traditional process mode to the agile mode, it was very challenging to every team member. In the change, it is likely that unsuccessful actions bring negative feelings to all participants. Therefore at the beginning it would be good to take only the most important basic ideas for the process. This way, it does not give the most optimate results than the pure agile method, but it produces however better results than the traditional working platform. In this way you can give positive feelings and sooth possible transfer for a defined agile method in the future. Before the organization transfers to use agile methods, it must be verified that all the organization gets at least basic level training on the used method.

In the research work it was searched for new development ideas and points of view, and in the work theories were used. The literature part gave a good overview to the development methods. The literature gave examples on how different development methods should be working in theory. This gave a good opportunity to compare theory to the real life and think over the differences in my mind. Those theory parts gave the basis to the real development work. During on the development process it was tried to take in the best way development tasks partly to the product process. Sometimes the development tasks precede the biggest steps forward, but occasionally it was taken several steps backward. This happened especially when the products changed in the organization. The consequence of products' change was that it took time before the teams internalized the new product and could continue to the development process.

In the development of the product process, it occurred to me that somewhere in agile product process it would be good to sometimes take waterfall model in use in the process. In the initial phase of a new product the requirements are unclear and therefore it must get as soon as possible those requirements clear. The waterfall model offers possibilities do the work so that situations do not change so fast than in agile models. This makes easier to do the product when employees are all over the world and communication is difficult between teams. This way leaves time to the teams to communicate with each other and get problems solved and requirements clear before the growth of the product speeds up rapidly and requirements become more complex.

When the employees are begin to think their own work input and what things are essentials in their own job. It is a good thing, because the organization rises up the question: could we do our work better? Often we do work according to general habits because somebody has told us how you have to do and we do not discuss what habits are important to the product. Often those actions are already taken in the progress although changes would not go through so well. There is much strength in the internal change of an organization. Internal changes go through better when the employees may participate in finding development targets and develop those targets. This helps the employees to be better committed to the upcoming changes. The common development process gives to different teams' solidarity and expression of emotion although work tasks and teams' locations are in different places. The development process gives the team the opportunity to change point of views and find common line to making the product. Experiences of different team members bring extensive field experience in the software development work. Doing development work team members have to analyze own and another employee's work and find the best practices for the project. This gives the employees a feeling that they own the product and this gives motivation to the product and process developing. The opposite of this is, that employees' views are different from targets and from what is the right way of development. One problem is how to avoid conflicts between employees and keep harmony in the team. It is always a risk that all team members cannot say their opinion. It is a problem that strong people push through their views and thus the communication is not equal, which weight out opinions.

Looking at the development process from the outside of an organization it can be said that if an organization has begun to develop its own process, it brings market benefits to the company compared to other companies. The Market benefit can mean that the organization will come more cost-effective in a particular time slot. This means that the company saves costs and product's price comes competitive compared to the others companies' products. As the company's employees are committed to the product, their investment to the product development grows. Then the employees begin to seek for information outside the company. As the employees develop their own knowledge, the

company gets new information for product development and this gives better benefits in the market.

It can be regarded as external threats all the things that the team members did not have any influence on. The threats of the organization's developing can be organizational structures changes, employee changes, resulting in a loss of knowledge. It affects also to how long team members are working in the project. It also affects to members' feelings and how are motivated to work. Team members cannot straight affect to the company's strategy changes or product importance changes, but those are affecting their motivation.

Perhaps in the future, the development process of the product will continue. I do not anymore continue with this product, because the responsibility for product process development was transferred to another organization. I would raise further action to the things that came out from the inquiry research: communication between different teams and employees, preparing for the demo and the planning day, lack of the documentation and contents, the level of requirements to clear and the test cases usability between the sprints. Those development tasks are a good starting point for the next development process and the same can follow how new research and development teams feel their working with the product.

As in every development working, this was own time target and it was written in the situation of the moment. In the real, the development process is a continuing process, which never ends. Therefore I hope that this thesis raises questions and gives a new point of view to the reader. I also hope that the interviews and inquiry results give information to employee on how to make the development process plan for his own organization.

5 List of references

Abrahamsson, Pekka, Salo, Outi, Ronkainen Jussi and Warsta Juhani 2002. Agile software development methods: Review and analysis. [online][Referenced 22.01.2008]. http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf

Beck, K., Beedle, M., Bennekum, A. and Cockburn, A. 2001. Manifesto for Agile Software Development. [online][Referenced 14.01.2008]. http://www.agilemanifesto.org/.

Boehm, Barry 2002. Get ready for agile methods, with care. Computer. Volume 35.Issue 1. Jan. Page(s):64 – 69. Digital Object Identifier 10.1109/2.976920

Boehm, Barry W. 1998. A Spiral Model of Software Development and Enhancement. Page(s):61 – 72. [Online][Referenced 6.8.2008]. http://www.cs.usu.edu/~supratik/CS%205370/r5061.pdf.

FSD KvantiMOTV 2007. Tutkimusprosessi. [online][Referenced 02.09.2008]. http://www.fsd.uta.fi/menetelmaopetus/tutkimus/prosessi.html

Gornik, D. 2003. IBM Rational Unified Process: Best Practices for Software Development Teams. [online][Referenced 16.01.2008]. ftp://ftp.software.ibm.com/software/rational/web/whitepapers/2003/rup_bestpractices.pdf.

Haikala, Ilkka and Märijärvi, Jukka 2004. Ohjelmistotuotanto. The tenth edition. Helsinki: Talentum Media Oy.

Hirsijärvi, Sirkka, Remes, Pirkko, Sajavaara Paula 2007. Tutki ja kirjoita. The thirteenth edition Kustannusosakeyhtiö Tammi:Helsinki.

Holopainen, Heini ja Hamina – Mäki, Eija 2003. TietoEnator Oyj/ Public & Healthcare/dGov: TE Object,“Perinteinen”oliomenetelmä ketteräksi. [online][Referenced 17.01.2008]. http://www.pcuf.fi/sytyke/syysseminaarit/risteily2003/AgileSytyke2003.pdf.

Järvenpää, Eila 02.02.2006. Teknillinen korkeakouluTuotantotalouden osasto, LAADULLINEN TUTKIMUS. [online][Referenced 02.09.2008]. http://www.cs.tut.fi/~ihtesem/k2007/materiaali/luento4.pdf.

Ketvell, Antti. Tampereen ammattikorkeakoulu: Liiketalous, Henkilöstön asema muutoksessa. [online][Referenced 02.02.2009]. https://oa.doria.fi/handle/10024/4677.

Leffingwell, Dean 2007. Scaling software Agility. Boston: Addison-Wesley.

Myers, Glenford J. 2004. The art of software testing. The second edition. New Jersey: John Wiley & Sons.

Pol, Teunissen and Van Veenendaal 2002. Software testing : a guide to the TMap approach. Harlow : Addison-Wesley.

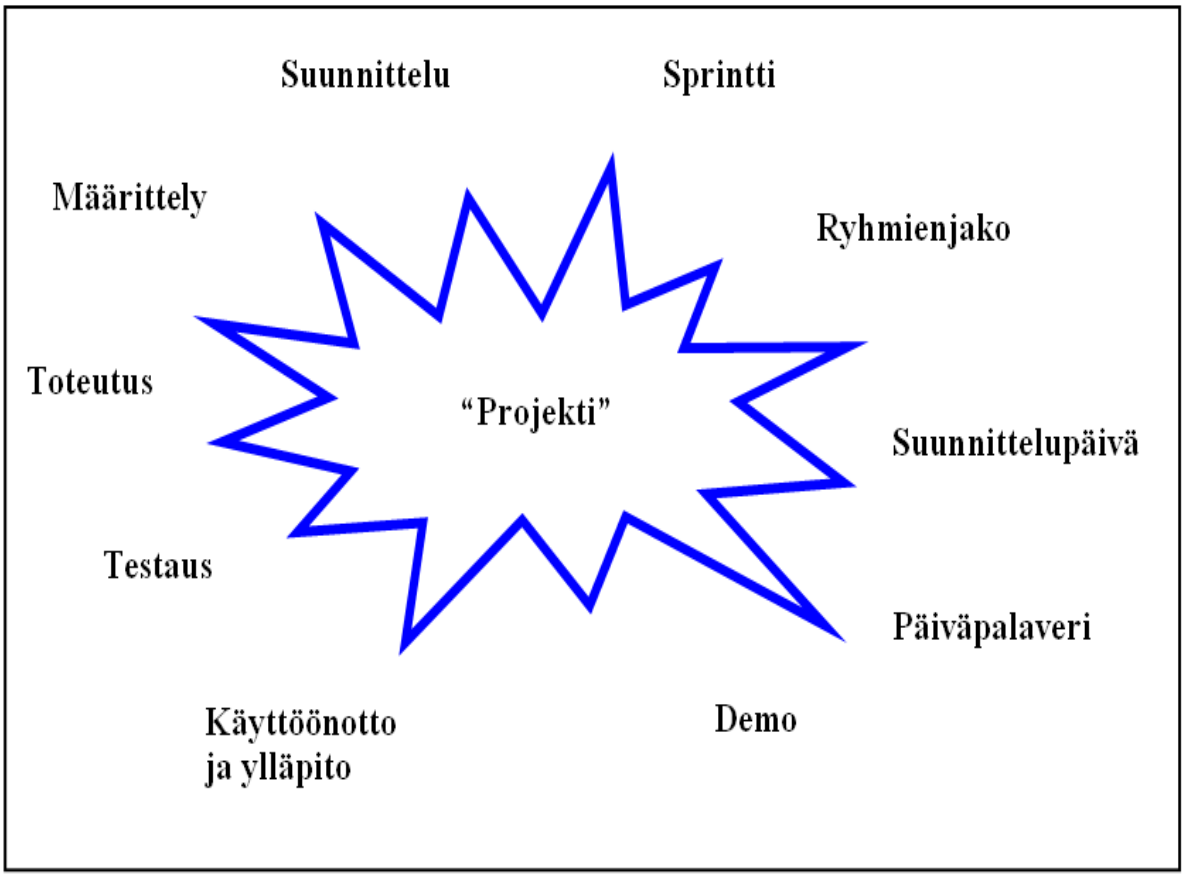
Schwaber, Ken and Beeble, Mike 2002. Agile Software Development with Scrum. Prentice Hall.

Schwaber, Ken 2003. Agile project management with scrum. Microsoft Press.

Sommerville, Ian 2004. An Introduction to Software Engineering. [online][Referenced 20.11.2008]. http://www.cs.st-andrews.ac.uk/~ifs/Books/SE7/Presentations/PDF/ch1.pdf

6 Appendices

Appendice 1: Ryhmähaastattelujen aihejako



Appendice 2: Ryhmähaastattelujen analysointitulokset (1/3)

Eniten motivoivat asiat

- Kun tiimin tekijät ovat lähekkäin, sujuu työ jouhevammin ja nopeammin. Vaatimukset saadaan nopeammin valmiiksi. Speksaus, koodaus ja testaus samassa sprintissä ja lomittain. Silloin pystytään reagoimaan nopeammin ja syntyy tekemisen meininki.
- Tekijät ovat samassa ryhmässä ja lähekkäin silloin ei tunne olevansa ongelmiensa kanssa yksin. Tiimin henkinen tuki auttaa ongelmia ratkoessaan.
- Backlog:ilta voi poimia sellaisia tehtäviä joilla pystyy laajentamaan omaa osaamistaan.
- Päiväpalavereissa tiedon siirtyminen tapahtuu ryhmän sisällä.
- Demoissa näki kuinka se asia oikeasti kokonaisuudessa toimi (motivointi).
- Scrummasterilla parempi näkemys työhön, kun hän osallistuu tekemiseen.
- Asennus onnistuu tyhjältä pöydältä joustavasti ja nopeasti.

Soc Classification level
2 © Nokia Siemens Networks Presentation / Author / Date
Nokia Siemens Networks

Appendice 2: Ryhmähaastattelujen analysointitulokset (2/3)

Haastateltavien huonoksi tuntemat asiat:

Tiimi:

- Tiimien yhteisten asioiden tiedonkulku ei aina toimi, jos tiimit työskentelevät eri kaupungeissa.
- Kuinka asioita pitäisi tehdä silloin, kun tiimien välillä on näkemyserot (Tampere/Helsinki). Näkemyseroasioissa tarvittaisiin erotuomari päättämään millä tavalla asiaa lähdetään toteuttamaan.
- Vastuu (kompetenssi) keskittyy vain yksiin käsiin, sairastumiset, lähdöt.
  - Osa hommista jakautui vain yhdelle tekijälle. Koodari koodaa ja testaaja testaa. Tekijöitä on liian vähän tehtävään työmäärään nähden.
- Tiimit ovat liian isoja, jos Scrum masterin on oltava täysin perillä mitä tiimissä tehdään. Tiimin sisällä ei kiinnosta, jos toinen tekee täysin erilaista hommaa.
- Tiimit eivät jakautuneet, kun tehtävät pilkkoutuivat tiettyihin osa-alueisiin. Tiimit jämähtivät paikoilleen ja eivät uudelleen organisoituneet tehtävien muuttuessa.

Nokia Siemens Networks

Haastateltavien huonoksi tuntemat asiat:

Palaveri:

- Liikaa palavereita. Palaverit vievät liikaa työaikaa oikealta tekemiseltä. Ei koske päiväpalavereita.
- Päiväpalaverissa on varottava liiallisen tuloksellisuuden hakua. Katsottava ettei yksilön paine työn tekemiselle tule liian raskaaksi. Positiivinen asenne on säilytettävä tekemisessä.

Demo:

- Demoon valmistautuminen vie liian suuren osan aikaa Sprintistä.
- Demoja ei rajattu tarpeeksi vaan yritettiin demota päästä päähän. Esitystä yksinkertaistettiin liikaa, jotta kuka tahansa voisi ymmärtää mitä tapahtuu.

Nokia Siemens Networks

Appendice 2: Ryhmähaastattelujen analysointitulokset (3/3)

Haastateltavien huonoksi tuntemat asiat:

Scrum master/Managerit:

- Työmääräarviot ovat liian optimistisia tehtäviin nähden, niin tekijöillä kuin managereilla.
- Planningday:t tehokkaammaksi. (Parempi esivalmistelu ennen Planning day:tä)
  - Planning day:hin mennessä ei oltu mietitty riittävästi vaatimuksia. Tuotteenhallinta ei aina tiennyt mitä halutaan tuotteelta.
- Maintenancen sisältö muuttui sprintin aikana. Iltapäivän aikana tultiin sanomaan, että illan aikana on saatava tietty asia tehtyä ja lähetettyä asiakkaalle. Työn keskeytys näkyi uusien vaatimuksien valmistumiseen (aikataulu).
- Sprintin sisältö on suunniteltu liian tiukaksi jolloin kompetenssin siirrolle ei ole aikaa ryhmän sisällä.

Nokia Siemens Networks

Haastateltavien huonoksi tuntemat asiat:

Dokumentointi:

- Alussa oli dokumentoinnin puute. Toimivuutta ei kirjattu ylös siitä kuinka asian kuuluisi toimia. Koodi ei kerro kuinka vaatimuksen kuuluisi toimia.
- Vaatimusten taso oli liian korkea. Aina ei saanut tarkennusta vaatimuksiin siitä mitä haluttiin. Loppu kädessä sprintin aikana piti päätellä mitä vaatimus tarkoittaa.

Työn tekeminen:

- Testicaset eivät kestänyt dynaamista ympäristöä. Case:n toiminnallisuus hajosi kerta toisensa jälkeen.
- Ei tiedetä mitä asiakkaalle on asennettu. Asiakkaille asennettujen softien listat laahasivat jäljessä.
- Vaatimuksien sisältöä ei aina ymmärretty ennen kuin ruvettiin toteuttamaan niitä.
- Liian monta asiakashaaraa softapaketeissa, vaikka pyrittiin pitämään geneeristä softaa. Niiden ylläpito oli hankalaa.

Nokia Siemens Networks

Appendice 3: Kyselylomake

Tämä kysely on jatkoa vuosi sitten olleille haastatteluille. Kyselyn tarkoituksena on saada selville se kuinka haastatteluissa olleet asiat ovat muuttuneet vuoden aikana. Mitä mieltä olet seuraavista väitteistä?

	Täysin samaa mieltä	**Jokseenkin samaa mieltä**	**Aivan sama**	**Jokseenkin eri mieltä**	**Täysin eri mieltä**	**En osaa sanoa**
Tiedonkulku toimii hyvin eri tiimien välillä.	1	2	3	4	5	6
Uuden asian opettelu on helpompaa kuin vuosi sitten.	1	2	3	4	5	6
Kompetenssin siirto on helpottunut viime vuoden aikana.	1	2	3	4	5	6
Palavereiden määrä on sopiva työaikaan nähden.	1	2	3	4	5	6
Tiimien koko on hyvä verrattuna niissä tehtäviin työtehtäviin.	1	2	3	4	5	6
Demoihin valmistuminen on otettu hyvin huomioon Sprintin aikataulussa.	1	2	3	4	5	6
Demot ovat sisällöltään selkeitä ja hyvin ymmärrettäviä.	1	2	3	4	5	6
Planningday:t ovat tehokkaita ja niihin on valmistauduttu hyvin.	1	2	3	4	5	6
Dokumentointi on tarpeeksi kattavaa.	1	2	3	4	5	6
Product backlogilla olevien vaatimusten taso on selkeä ja hyvin ymmärrettävää.	1	2	3	4	5	6
Testi casejen toiminta säilyy hyvin eri Sprinttien välillä.	1	2	3	4	5	6
Testaus-/ kehitysympäristössä käytettävien ohjelmistojen versiot on hyvin tiedossa.	1	2	3	4	5	6

Appendice 4: Kyselyn tulokset

Vastauksia 14 kpl.

	Täysin samaa mieltä	**Jokseenkin samaa mieltä**	**Aivan sama**	**Jokseenkin eri mieltä**	**Täysin eri mieltä**	**En osaa sanoa**
Tiedonkulku toimii hyvin eri tiimien välillä.		3	1	8	2	
Uuden asian opettelu on helpompaa kuin vuosi sitten.		5	5	3		1
Kompetenssin siirto on helpottunut viime vuoden aikana.		5	4	3	1	1
Palavereiden määrä on sopiva työaikaan nähden.	2	7		3	2	
Tiimien koko on hyvä verrattuna niissä tehtäviin työtehtäviin.	1	8	1	4		
Demoihin valmistuminen on otettu hyvin huomioon Sprintin aikataulussa.		1	4	7	2	
Demot ovat sisällöltään selkeitä ja hyvin ymmärrettäviä.		2	1	5	6	
Planningday:t ovat tehokkaita ja niihin on valmistauduttu hyvin.		1		4	9	
Dokumentointi on tarpeeksi kattavaa.		1	3	6	3	1
Product backlogilla olevien vaatimusten taso on selkeä ja hyvin ymmärrettävää.			2	12		
Testi casejen toiminta säilyy hyvin eri Sprinttien välillä.		2	2	4	2	4
Testaus-/ kehitysympäristössä käytettävien ohjelmistojen versiot on hyvin tiedossa.		5	2	5		2