



Agile Software Development of Embedded Systems

Version : 1.0
Date : 2005.12.13
Pages : 29

Authors

Minna Pikkarainen

Status

Final

Confidentiality

Public

Agile Deliverable D.5.2.5

F-Secure Agile Assessment Experience Report of Three Agile Trials

Abstract

The purpose of this document is to present experiences and results of Agile Assessment at F-Secure in three agile software development and one plan-driven comparison project.



ITEA

INFORMATION TECHNOLOGY
FOR EUROPEAN ADVANCEMENT

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

CHANGE LOG

Vers.	Date	Author	Description
0.1	4.3.05	Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko	First Full Draft Created
0.2	13.6.05	Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko	Review together with the whole assessment team
0.3	16.6.05	Minna Pikkarainen	Some recommendations based on the meeting at F-Secure
0.4	19.10.05	Minna Pikkarainen	Review with Jari Still, Changes needed to change the document from confidential to consortium level
1.0	13.12.05	Minna Pikkarainen	Document sent to Consortium

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

TABLE OF CONTENTS

CHANGE LOG	2
1 Introduction	4
2 Background Theory	4
2.1 An Agile Assessment Theory	4
2.2 Success and Failure Factors in agile software development.....	5
3 Research Context.....	7
3.1 Description of the Company	7
3.2 Agile Assessment Environment	7
4 Progression of the Agile Assessment in F-Secure Corporation.....	11
4.1 Focus Definition	11
4.2 Data Collection.....	11
5 Empirical Evaluation of the Agile Assessment results.....	15
5.1 When to use agile methods in the F-Secure projects?	15
5.2 Content suggestion for the agile F-Secure Process model.....	18
6 Improvement idea mapping with agile based solutions	22
6.1 Possible agile practices for the traditional FPRP	25
7 Summary.....	27
References	29

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

1 Introduction

The purpose of this document is to describe the progression and the key results of an F-Secure Agile Assessment, which was performed for four selected case projects in the project called Agile ITEA. The aim of the agile assessment was to find the best agile practices for the F-Secure context and, thus, to support the F-Secure level agile product development model creation alongside with the traditional F-Secure Product Realization Process (FPRP) (common processes at F-Secure). Another purpose was to compare the efficiency of the traditional and agile software developments as well as to establish how the agile practices could be utilized also in the traditional software development projects.

The report is composed as follow, the section 2 presents background and theoretical assumptions for the agile assessment. Third section provides the progression of the agile assessment. Section 4 described the main results of the agile assessment analysis. The last section concludes the document with final remarks and outlines future actions.

2 Background Theory

2.1 An Agile Assessment Theory

Agile Assessment in F-Secure case was based on Agile Assessment method including the following steps. 1.) Goal Definition; 2.) Agility Evaluation; 3.) Data Collection Planning; 4.) Data Collection; 5.) Analysis and Improvement Idea Mapping with the agile based solutions 6.) Final Workshops (Figure 1).

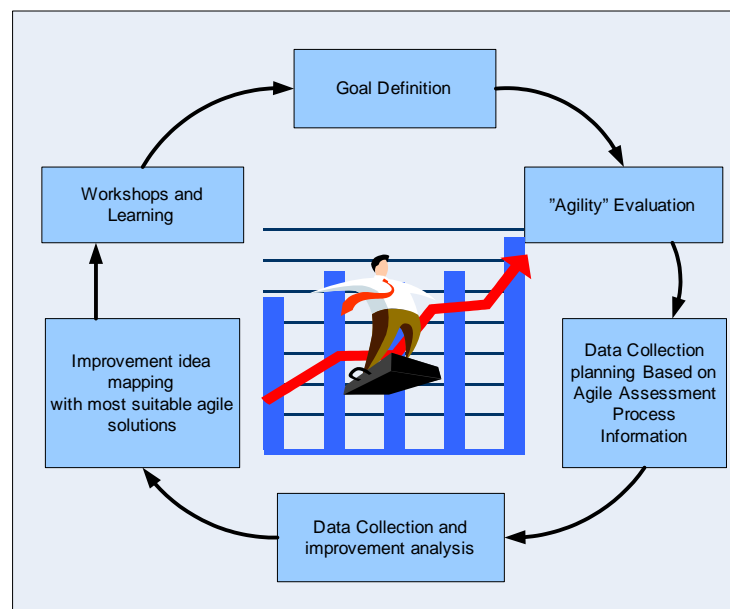


Figure 1. An Agile Assessment Framework

In the goal definition step, the focus is set for adopting agile methods. The agility evaluation is a starting point for a deeper evaluation giving an appraisal whether the agile methods would be suitable and effective in the selected projects. The agile assessment data can be collected using interviews, agile assessment workshops, observing in the organization improvement meetings and from the existing

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

documentation of iterative SPI actions (from PIWs) and improvement ideas (from project post-mortem). Also, various project metrics (e.g. effort and defect) data can be utilized in the analysis.

The agile assessment workshops are conducted in order to 1) collect the strengths and improvements from the different software development phases and 2) discuss the possibilities of increasing the agility together with the project stakeholders, and therefore, 3) to support the project and organizational learning between the different projects and the organizational level agile software development model development. A facilitator in the agile assessment is an agile assessor or assessors), who is aware of the available methods as well as the assessment method. (Pikkarainen, Salo et al. 2005).

Additional information of the Agile Assessment framework is described in the Agile Deployment Model and Agile Assessment Framework documents <http://www.agile-itea.org/intern/WP4>.

2.2 Success and Failure Factors in agile software development

Every software development project is always unique. Differences bring-up from the project environment e.g. domain, customer, development tools. Success of the software development can also be considered based on the factors that are affecting on how successful or challenging the software development project actually was in the different organizational contexts. The software development success factors can be characterized in many ways (Abrahamsson 2000). One example of this kind of characterization is to deal the success to 1) project efficiency, 2) usability issues, 3) business success and 4) direct operational success (Abrahamsson 2000). The focus of this document, however, is to discuss of the software development success especially from the agile software development viewpoint.

Software development success in organization can be based on the efficient software development methods (agile or traditional) and their correct use in the suitable situations. Also the user involvement, executive management support and clear statement of requirements in Table 1 can be seen as a critical success factors for the different software projects (Reel 1999). Success can, however, turn to failure if technologies like development tools and platform are not suitable for their purposes in the current projects. Another failure factors could be, for example, lack of user inputs, incomplete requirements and changing requirements and specifications (Reel 1999). These are the factors that are tried to solve with the efficient use of agile software development methods, practices and tools.

Table 1. Success and challenge factors in Software Development projects (Reel 1999).

Project Success Factors of Responses		Project Challenged Factors of Responses	
User Involvement	15.9%	Lack of user inputs	12.8%
Executive Management Support	13.9%	Incomplete requirements & Specifications	12.3%
Clear Statement of Requirements	13.0%	Changing requirements& Specifications	11.8
Proper Planning	9.6%	Lack of Executive Support	7.5%
Realistic Expectations	8.2%	Technology Incompetence	7.0%

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

Smaller project milestones	0-7.7%	Lack of Resources	6.4%
Competent Staff	7.2%	Unrealistic Expectations	5.9%
Ownership	5.3%	Unclear Objectives	5.3%
Clear vision & Objectives	2.9%	Unrealistic Time Frames	4.3%
Hard Working, Focused staff	2.4%	-	-
Other	13.9%	-	-

Based on Reel analysis of the critical success factors in Table 1 it can be concluded that:

- Research above shows that at least seven of ten signs of software project failures are determined before a design is developed or a line of code is written
- User involvement is the most important factor in the software projects
- Requirements management is also significant factor for the project failure or success
- Importance of technology competence is not significant success factor or it is included competent staff-factor, but incompetence of technology is quite critical failure factor

In Table 2 the described success factors are categorized with the four key areas which are involvement of the stakeholders, requirement and feature management, project management and proficiency of development team. The categorization of the success factors are used in the agile assessment results categorization in section 6.

Table 2. Categorized success factors

Categorized factors	Success factor	Challenged factor
Stakeholders involvement	<ul style="list-style-type: none"> • User involvement • Executive Management Support 	<ul style="list-style-type: none"> • Lack of user inputs • Project managers don't understand user's needs • Business needs change • Sponsorship is lost • Managers ignore best practices and lessons learned
Requirements/feature management	<ul style="list-style-type: none"> • Clear Statement of Requirements 	<ul style="list-style-type: none"> • Incomplete requirements & Specifications • Changing requirements & Specifications • Project changes are managed poorly
Project management	<ul style="list-style-type: none"> • Proper Planning • Smaller project milestones 	<ul style="list-style-type: none"> • Lack of Resources • Unrealistic Expectations • Unclear Objectives • Unrealistic Time Frames • Managers ignore best practices and lessons learned • The project's scope is ill-defined • Deadlines are unrealistic
Proficiency of development team	<ul style="list-style-type: none"> • Competent Staff • Hard Working, Focused staff 	<ul style="list-style-type: none"> • Technology Incompetence • Resistance of the developers • The project lacks people with appropriate skills

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

3 Research Context

3.1 Description of the Company

The case study of this research was conducted at F-Secure Corporation, an organization which is developing antivirus, network encryption, desktop firewall with intrusion prevention, anti-spam and parental control products to protect individuals and businesses against computer viruses and other threats spreading through the Internet and mobile networks.

3.2 Agile Assessment Environment

Agile Assessment results at F-Secure were based on two Scrum, one Mobile-D and one traditional software development project experiences.

Scrum 1 project is focused on developing a security system management tool. The project team consists of six persons including four software engineers, a Scrum master/project manager and the Quality Engineer. The core of the Scrum 1 project (developers) works in an open office space. The project had been going on for over a year and the Scrum method itself is used in the project about five months until now conducting three Sprints (Figure 2).

Another Scrum project is a research project, which aim is for developing new innovations for supporting the F-Secure product development. The idea of the Scrum method piloting was based on Scrum 1 project excellent Scrum method utilizing experiences. The project team of the Scrum 2 project consists of four persons. One of them works as a Scrum master (i.e., project manager) and the other three project members are software engineers who are focused on the research work including also the implementation as well as the antivirus software unit testing activities. The project team has been operated about four months and the Scrum method had been successfully utilized in the project about for three months until now.

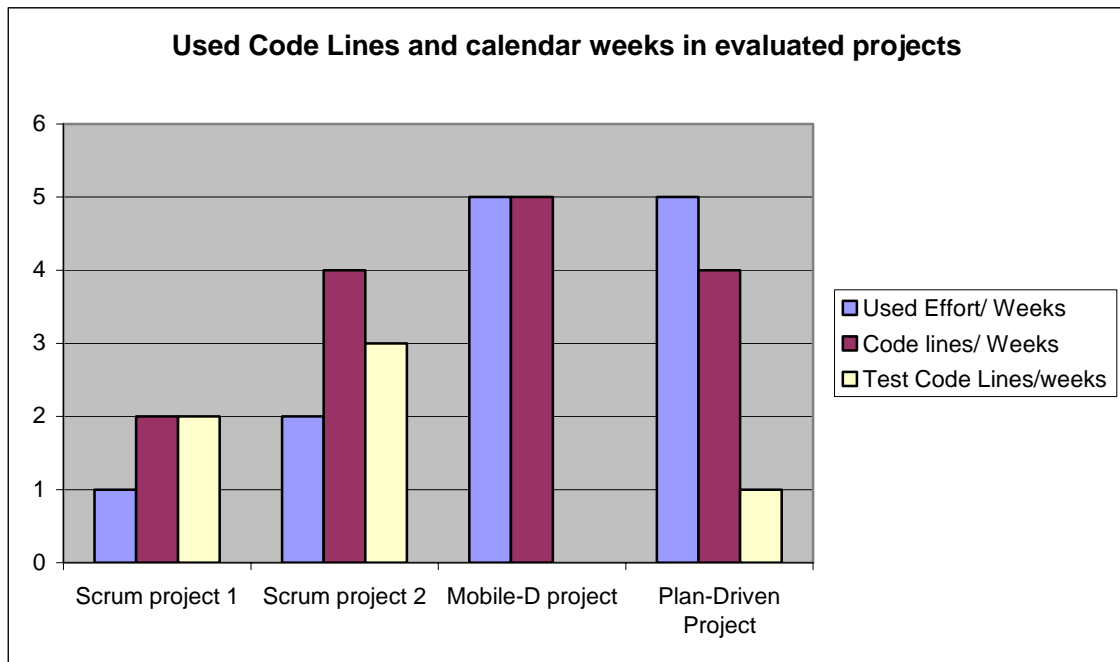
The goal of the *Mobile-D project* was to develop a mobile security application. The core of the case project consisted of four software developers and two quality engineers who worked in an open office space. The Mobile-D project team conducted five software development iterations. The team leader of the project was an expert of the Mobile-D process and was provided by the research organization. Thus, the team had constant support and coaching available on adopting the new agile process model and its practices and tools. (Pikkarainen, Salo et al. 2005).

Traditional project is focused on developing a new version of virus protection system. Project was very large and critical, because it is pointed to consumer markets and amount of potential consumers users is huge. The project team consist of 55 persons software engineers, a project manager and quality engineers.

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05



Efforts/ weeks	Code Lines/weeks	Test Code Lines/ weeks
0-100	1	1
100-200	2	2
200-300	3	3
300-400	4	4
400-500	5	5

Figure 2. Effort/ Code lines used in the evaluated project

The code lines/ weeks in Figure 2 was highest in Mobile-D project but it has also used most resources in the short time. Both Scrum projects were created more test code than the plan-driven project. This may have direct affects on the product quality. The Scrum 2 was a research project which purpose was to test and create new technological innovations for F-Secure product development.

In addition, all of the evaluated the project included different stakeholders such as the organization management, customers and Quality Engineers (QE). However, the Mobile-D project also had the VTT Technical Research Centre of Finland support including an exterior facilitator for post-iteration workshops (Salo, Kolehmainen et al. 2004) organizing and result analysing. All of the evaluated agile projects were also located in the same site which enabled to the project team a good possibilities for the daily, face to face communication as expected based on the agile software development principles (Beck, Beedle et al. 2001) and methods [Scrum (Schwaber 1995) and Mobile-D (Abrahamsson, Hanhineva et al. 2004)].

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

Table 3. Communication and location of the evaluated projects

Location&Communications:	Scrum 1	Scrum 2	Mobile-D	Traditional
Customer location: on-site (100%) / on-site (50%) / off-site:	50	100	100	50
Customer communication: face to face daily / weekly / monthly:	face to face daily + monthly (extended set of customers)	on-site (100%)	on-site (100%)	Weekly Communication with business unit

In Scrum 2 and Mobile-D projects the customers were available on-site in the same department, but not constantly working in the same office-space with the developers as suggested in Extreme Programming (XP) (Beck 2000). However, in the Scrum 1 project have 50% on-site customer, which might complicate the Scrum method use possibilities and therefore to require more work from the project management (Table 2). In traditional project, instead, uses the traditional development methods having a weekly communication with the business unit (product manager).

Agility of the evaluated projects were assessed using the Boehm and Turner agility dimensions where the(Boehm and Turner 2003) 1) **Size** means The Number of people in the team, 2) **Criticality**: Products safety criticality, 3) **Dynamism**: Degree of requirements and change in technology, 4) **Personnel** the skill and experience of the team and 5) **Culture**, the agile project culture that should support the developers' freedom to make the technical solutions so that the development team can achieve a high motivation (Boehm and Turner 2003). In figure 2. each axis is labelled based on the agility dimensions. When a project's data points for each factor are joined and the resulting shape is located directly around the centre, this suggests the high agility (using of agile methods). Shapes that gather distinctly toward the periphery suggest lower agility (using of plan-driven methodologies). More varied shapes suggest the use of a hybrid method of both agile and plan-driven practices. In table 5 the agility dimensions of the Scrum 1, Scrum 2, Mobile-D and traditional projects are analysed using Boehm and Turner agility dimensions. The data for this analysis is based on a) the query of the needed metrics (Table 7) which was fulfilled by the each project manager and b) focused in additional phone and e-mail discussions with the project managers and VTT Researchers.

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

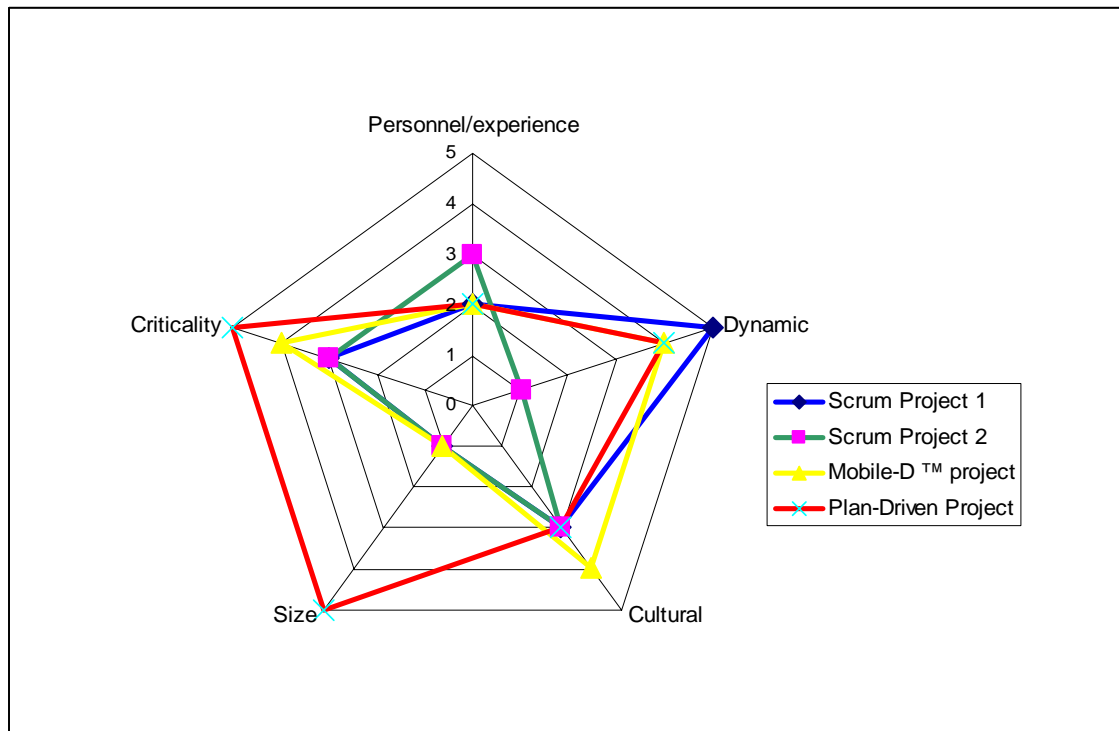


Figure 3. Agility of the evaluated projects

Table 4. Agility of the evaluated projects

	Personnel/experience	Dynamic	Cultural	Size	Criticality
Scrum 1	2	5	3	1	3
Scrum 2	3	1	3	1	3
Mobile-D	2	4	4	1	4
Traditional	2	4	3	5	5

According to this Boehm and Turner analysis (in Figure 3), it can be disclosed that the agile projects have a quite same experience level. All of them have a very small team (under 10 persons). Dynamism in Scrum 1 and Mobile-D projects were high but well handled because of the well focused requirements in the iteration planning meetings. Only, in Scrum 2 project, conducting to the nature of the research work, had had many changes also during the agile software development. In Addition, all of evaluated projects were critical from the F-Secure Corporation viewpoint. Instead, the traditional project which is a large, more complex project which is developed with the minor time using the major effort has clearly lower level agility also according to the Boehm and Turner analysis. In fact, the traditional project has a similar culture and a lot of changes in the detailed level requirements, at the assessment moment, its size and criticality supported the of traditional SW development methods.

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

4 Progression of the Agile Assessment in F-Secure Corporation

4.1 Focus Definition

The Agile Assessment focus at F-Secure was defined in the one two hour meeting together with the VTT Agile Assessors and F-Secure management. The aim of the Agile Assessment was to support the F-Secure purpose to develop the agile F-Secure product development model alongside with the traditional mature F-Secure Product Realization Process (FPRP) (Figure 4).

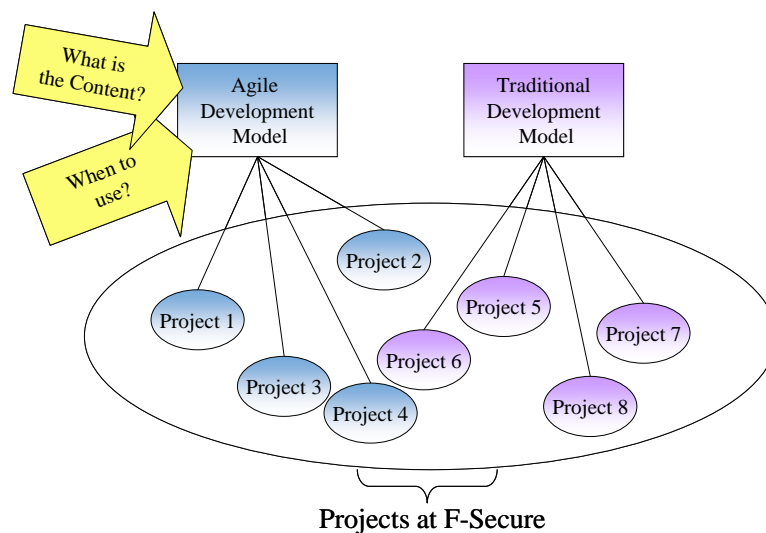


Figure 4. Agile Assessment Focus at F-Secure

Therefore, the goals of the agile assessment were dealt with the three main parts which were: 1) To define the agility needs in different project contexts (when to use the agile software development model), 2) To find the most suitable agile practices in the F-Secure (content of the agile development model) and 3) To compare the efficiency of the agile and traditional projects in order to bring some suitable agile practices also as a part of the FPRP.

4.2 Data Collection

The data for the F-Secure agile assessment was collected both using qualitative data (e.g. the agile assessment workshops, focused interviews, observation in the further project planning meeting) and quantitative data (e.g. project metrics). In addition, in the Mobile-D case, also, the validated post-iteration workshop and post-mortem data (worksheets and action point lists) was used. The aim of the agile assessment workshops and focused interviews were to clarify the qualitative data (e.g., customers, developers, managers and testers opinions) of each project phases behind the collected quantitative data (e.g. project effort and code line metrics).

In the Scrum projects the qualitative agile assessment data was collected in the three hours agile assessment workshops, which, actually, has many similarities between the

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

methodology shaping workshop described by Alistair Cockburn. At least the information shared during the workshop (e.g. history of the project, key things that were wrong, what is most relevant to keep in the next projects) are quite similar. The agile assessment workshop, however, was organized as a two main steps. The purpose of the first step was to define the success and failure factors of the agile software development at F-Secure based on this current project experiences. In this step, the participants first used about ten minutes for describing the agile software development success and failure factors in the yellow labels. After that, everyone put the labels in the worksheet (Figure 5) and clarified the key points behind the previously defined responses.



Figure 5. Example of the success and failure factor analysis

The aim of the second step was to analyse the project current state (strengths and improvements) for each development phase in the Scrum projects. Before the workshop, the agile assessor prepared the templates for each Scrum phases based on the collected metric data in that particular project. Thus, in the agile assessment workshop, each participant has possibility to fulfil the templates (example in Figure 6) for using about ten to fifteen minutes for the key strengths and improvements definition for each development phase. After that everyone described what they really meant with their responses.

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

Name and goal of the sprint 1

Your Role in the Sprint 1	Example of the used and possible new Agile Practices
Ideas for the future SW development:	

Figure 6. Example of the Workshop Template

Agile assessment data collection had some differences (in Table 5) between the selected projects. For example, in Scrum project 1 the data was collected using the agile assessment workshop including five participators (i.e., three software engineers, project manager and one quality engineer). In Scrum 2 and traditional projects the data collection mechanism was quite similar. The key differences in there were that the Scrum 2 workshop included a three software engineers and project manager (i.e. Scrum master) whereas the traditional project workshop included only managers (e.g. two product managers and one project managers) excluding the one quality engineer. In Mobile-D project, there was no the agile assessment workshop because the post-iteration workshop (worksheets and action point lists), post-mortem data (most suitable and unsuitable practices for the future agile based software development) as well as observation results of the second Mobile-D project meetings and focused interviews offered the needed, validated, data that could be utilized in the agile assessment analysis directly.

Table 5. Agile Assessment data collection in F-Secure

Project	Data Collection methods	Participators
Scrum 1	Agile Assessment workshop + project metrics	three software engineers, project manager and one quality engineer
Scrum 2	Agile Assessment workshop + project metrics	three software engineers and project manager
Mobile-D	Observation second Mobile-D project planning workshop, project metrics, post-iteration workshop	Post-iteration workshop: Mobile-D-project development team Other workshops: Mobile-D project

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

	and post-mortem results and one interview for the project manager	team and the other stakeholders
Traditional	Agile Assessment workshop + project metrics	two product managers, manager, project manager and quality engineer

The project metrics (e.g. effort data, number of code lines, criticality, number of changes, size and experience level of the project) were planned and sent to the project managers using the short e-mail query in Table 6, where the response were required before the agile assessment workshops. Therefore, the agile assessors had possibilities to use the collected metrics both in the agile assessment workshop planning as well as in the actual agile assessment analysis.

Table 6. Query of the needed metrics

Areas	Questions
Effort and code lines	M.1 Used Effort for each release M.2 Used Effort for each development phase (preparation, requirement definition, working, test cases, validation, project planning/management) M.3 Effort used in Test Case Creation M.4 Number of test code vs. implementation code for each project M.5 Number of test defects M.6 Number of 10000 lines of executable code
Project	Name of the evaluated project: Application domain(s) of the project: Application type: Project purpose: Main Phases:
Customer	Project type: new project from scratch / new version / enhancement / other: Criticality of the product (Loss due to impact of defects) [Low loss 1, 2, 3, 4, 5 High Loss]: Project size: Estimated/Actual Effort (days): Estimated: Actual: Number of resources: Developers: Testers: Architects: Others: Experience level: Developers: Testers: Architects: Others: Number of resources: 1-3 years: 4-8 years: >8 years: Domain expertise: Low / Medium / High: Language expertise: Low / Medium / High: Experience of proj. mgr: Low / Medium / High: Specialist available:
Location/communication	Team location: co-located (100 %) / co-located (50 %) / off-located: Team communication: face to face daily / weekly / monthly: Customer location: on-site (100%) / on-site (50%) / off-site: Customer communication: face to face daily / weekly / monthly: Dynamism (% of average number of changes): Marketing demands [Number of external changes from marketing]: Dynamism [Percentage or Number of change requirements/month]: Culture [Chaotic 1, 2, 3, 4, 5 Order]:
Other	Component reusable levels: 0-10 % /10-30 % / > 30 %

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

5 Empirical Evaluation of the Agile Assessment results

The purpose of this section is to compare and analyse the results of the Scrum, Mobile-D and traditional SW development projects evaluation. Other aims are to describe 1) when to use agile methods in F-Secure, 2) what would be the best agile practices for agile F-Secure product realization process model and 3) what agile practices could be used in the traditional future SW development projects at F-Secure.

5.1 When to use agile methods in the F-Secure projects?

In this document the analysis of each evaluated agile software development project at F-Secure has been presented using the success factors. According to the analysis of the Scrum projects experiences, the agile software development would offer the best solutions, especially, for the small teams including the social people and the team leader who is well aware of the agile based software development demands (training needs in F-Secure). On the other hand, the results support the assumption that the agile methods would be best for the projects that have large amount of change requests, for example, because of the research nature of the development work in Scrum project 2 (Table 7). In Mobile-D project, instead, the short iterations and a very efficient automated unit testing were disclosed as a one key for the project success. However, the daily communication, freedom of the developers and the documentation only based on the external needs can be seen as a success factors in both in Scrum and Mobile-D development (Table 7). However, one problem was the lack of definition of the external documentation needs during the agile software development projects.

Table 7. Success factors in the evaluated agile software development projects

Success Factors in Scrum development	Scrum 1	Scrum 2	Mobile-D
	Number of responses		
Communication (Daily meetings/wrap ups)	2	1	3
Freedom to developers, increased motivation	4	0	2
Documentation just based on needs	4	0	1
Quick feedback from Customer	3	0	0
Small group	1	2	0
Social peple/Good team spirit	1	1	0
Team Work	1	0	0
Good team leader	1	1	0
Adaptation of process	0	2	0
Short Iterations	0	0	7
Unit Testing	0	0	6
Information Radiator	0	0	2
Easier Planning (User stories and task cards)	0	0	2
Key Benefits of the agile based software development	Scrum 1	Scrum 2	Mobile-D
	Number of responses		
Better Quality (Less Bugs)	2	0	3
Productivity	2	0	1
Responce to change	2	0	1
Quick problem solution and decisions	3	0	0
Visibility	-0	0	1

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

Key benefits in Table 7 that could be achieved using the agile software development based on the agile assessment and Mobile-D project post-mortem workshops results would be 1) the better software quality because of the improved unit testing, 2) productivity because of the quick problem solutions, 3) team work, 4) increased motivation of the developers as well as 5) response to change because of the short development cycles and planning in the smaller pieces.

According to the agile project experiences in Table 8, the use of agile methods may be problematic in the situations where the project teams are 1) large, 2) complex, 3) distributed, 4) having many trainers or young, not so experienced, developers and if 5) the project had much dependencies with the external sources. Especially, the criteria of the external information needs (e.g. documentation demands) should be rather defined before the software development model selection. It may decrease the surprises (and failure factors) in the actual agile based development project. Agile projects have, however, achieved successful results also with the younger developers. However, the experiences are needed at least from the agile project management, process and architectural issues.

Table 8. Failure factors in agile development

Failure Factors in agile development	Scrum 1	Scrum 2	Mobile-D
	Number of responses		
Interfaces to other functions (e.g. documentation and marketing demands)	4	1	3
Validation by the QE team (e.g. responsibilities)	2	0	3
Knowledge Demands (Software Development Experience [process and technical])	2	0	0
Specific information needs from stakeholders	1	0	0
Prioritization with customer feedback	1	0	0
Agile Contracts (e.g. subcontractors)	1	0	0
Distributed development environment	0	1	0

As a conclusion of this analysis the several success factors and three key failure factors in figure 7 can be seen as a most important when selecting the right software development methods. In fact, based on the analysis the short iterations, unit testing and daily communication could be seen as minor requirements for the successful agile software development. On the other hand, it seems that the agile software development demands project team with high experience, responsibility definitions and trainings for the quality engineering team and very independent product (not the complex systems) without external documentation needs.

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

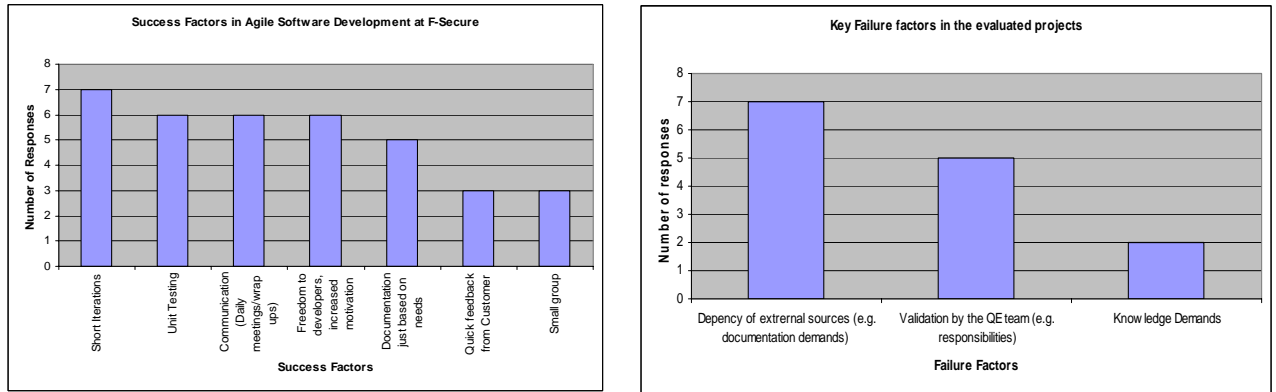


Figure 7. Conclusion of the success and failure factors

Compared to the literature in Table 9, the evaluated F-Secure projects (both agile and traditional) had many similarities and differences from their success factors. The main similarity comes from advantages of short iteration and strong involvements of the stakeholders.

Table 9. F-Secure success factors compared to the literature

Categorized factors	Success factor in research	Success factor in Agile project	Success factor in Traditional project
Stakeholders involvement	<ul style="list-style-type: none"> • User involvement • Executive Management Support 	<ul style="list-style-type: none"> • Communication • Quick feedback from customer • Teamwork • Short iterations 	
Requirements/feature management	<ul style="list-style-type: none"> • Clear Statement of Requirements 	<ul style="list-style-type: none"> • Easier planning (user stories and task cards) • Short iterations 	
Project management	<ul style="list-style-type: none"> • Proper Planning • Smaller project milestones 	<ul style="list-style-type: none"> • Documentation just based on needs • Small group • Easier planning (user stories and task cards) • Short iterations 	<ul style="list-style-type: none"> • Good co-operation SPS unit and testing teams • Good communication between project roles
Proficiency of development team	<ul style="list-style-type: none"> • Competent Staff • Hard Working • Focused staff 	<ul style="list-style-type: none"> • Freedom to developers, increased motivation • Good team spirit 	<ul style="list-style-type: none"> • Committed individuals
Categorized factors	Failure factors in research	Failure factor in Agile project	Failure factor in Traditional project
Stakeholders involvement	<ul style="list-style-type: none"> • Lack of user inputs • Project managers don't understand user's needs • Business needs change • Sponsorship is lost • Managers ignore best practices and lessons learned 	<ul style="list-style-type: none"> • Validation by the QE team (e.g. responsibilities) • Lack of commitment of company • Prioritization with customer feedback • Specific information needs from stakeholders 	<ul style="list-style-type: none"> • Not enough project and architecture management
Requirements/feature management	<ul style="list-style-type: none"> • Incomplete requirements & Specifications • Changing requirements & Specifications • Project changes are managed poorly 	<ul style="list-style-type: none"> • No good for totally new product • Specific information needs from stakeholders 	<ul style="list-style-type: none"> • Too many 'moving parts' in one projects • Too large scope for the first version of a new application
Project management	<ul style="list-style-type: none"> • Lack of Resources • Unrealistic Expectations • Unclear Objectives • Unrealistic Time Frames • Managers ignore best practices and lessons learned • The project's scope is ill-defined • Deadlines are unrealistic 	<ul style="list-style-type: none"> • Distributed development environment 	<ul style="list-style-type: none"> • Lot's of effort in coordinating sub-contractor • Failure to estimate and prevent major risks related to other projects and subcontractor • Too many 'moving parts' in one projects • Big amount of maintenance work from previous release • Different quality practices in different project where people have to participate at the same

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

			time=>confusion and demotivation <ul style="list-style-type: none"> • Not enough project and architecture management • Too large scope for the first version of a new application
Proficiency of development team	<ul style="list-style-type: none"> • Technology Incompetence • Users are resistant • The project lacks people with appropriate skills 	<ul style="list-style-type: none"> • Knowledge Demands 	<ul style="list-style-type: none"> •

As a conclusion of the evaluation in Table 9 it can be noticed that 1) agile project have only one project management specific problem whereas traditional project has several, 2.) Agile project has lack of user input even if Agile methods should offer excellent way to give feedback 3) Feature prioritization is also a problem in the Agile projects and 4) Agile project has lack of company commitment.

As a conclusion of this agile project success and failure factor analysis the key challenges of the use of agile software development methods and techniques are listed. According to the analysis the key business reasons why to use agile methods would be:

- + Response to changes
- + Need to get products to the marketing faster
- + Technological challenges
- + Need for daily visibility

Instead the key challenges for the F-Secure agile software development will be:

- High size of the project team
- New software developers
- Distributed development environment
- Complex product/ Product dependencies
- Documentation dependencies (communication between the other projects)
- Teamleader knowledge of the agile methods (should be high)
- Customer possibilities to participate in the project activities (should be high)
- Marketing involvement (should be high)
- Unstable resources
- QE team possibilities to participate in daily meetings
- Project manager possibilities to participate in daily meetings

5.2 Content suggestion for the agile F-Secure Process model

In the agile software development projects the different agile practices and techniques were used. Some of the used practices and techniques were similar in the all of the evaluated projects. However, the efficiency how they were used in the projects was different in some situations. In Table 10 the used practices are compared between the Scrum and Mobile-D projects at F-Secure.

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

Table 10. Comparison of the used agile practices (++++ Excellent practice, +++ Good practice in project, ++ Good but some problems were appeared during the project, + could be used in some situations or change, - Problematic in some situations, N/U Not used in the project)

Practice/ tool	Scrum 1	Scrum 2	Mobile-D
Iteration Planning	++++	++	++++
Short Iterations	+	++	++++
Frequent Delivery	++++	+++	+++
Information Radiators	++	++	++
Product Backlogs	++	+	++
Daily meetings/wrap ups	++	+++	+++
Pair Programming	+	++	-
Open Working Space	++++	+++	-
Automated Unit testing	+	+	++++
System testing by the QE team	+	N/U	+

Short iterations and iteration planning

Short iterations are one basic condition for the successful agile project, especially if the purpose is to get the solution for an external change flow in the software development projects. The short iterations enabled the easier task planning and effort estimations. Therefore, it was analysed as a one of the most suitable practices in Mobile-D project post-mortem meeting. In the Scrum projects, in the Scrum project 2 the requirements and working tasks was analysed weekly. Scrum project 1, however, had one month iterations. In this case, one comment of the developers was that the project might need the shorter development cycles. In the agile software development, one iteration length can be from one week to two months and the time used in the iteration planning is depending on the iteration length (Cockburn 2004).

Iteration planning was quite efficient in all agile software development projects at F-Secure. In Scrum 2 project there may be need to give more responsibility for the developers in the requirements prioritization. In the Mobile-D project there would be need more requirements preparation before the actual planning day. The purpose of this preparation time increasing would be to decrease time that the whole project team including the stakeholders are using with the actual iteration planning activities.

Frequent delivery

Without the frequent delivery the project may miss the realm problem with integration and development (Cockburn 2004). In the Scrum projects, the sprint results were presented to the customers after the each sprint. This was seen as a good mechanism to collect feedback of the customers and to make sure that the product is fulfilling the customer demands. In the Scrum project 2, however, one problem was the customer demands (re-scheduling) increasing because of the quick results of the research project. In Mobile-D case the releases were build bi-weekly which according to the post-mortem workshop was suitable for that particular project situation.

Agile software development tools

According to the comparison in Table 10 it can be seen that both information radiator and product backlogs were useful tools in the evaluated agile software development

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

projects. The good information radiator should be large, easily visible, understood, changing periodically and is easy to keep up to date (Cockburn 2004). According to the analysis it seems that the information radiators in F-Secure project fulfilled its purposes. However, the key problem was the laborious metrics collection and, in Mobile-D case, the problem was resolved by documenting the same information also in the electronic format.

While making the agile assessment for the different projects it has been seen that there are, actually, many different ways to use the product backlogs in the organizations. Some people has an opinion that the product backlog is just a list of the previous tasks, another people thinks that it is list of the overall requirements and at the same time some organizations are using the product backlog for maintaining the errors appeared from the product after the actual software development. In all cases, the basic problem was the maintainability of the product backlog. This was the most problematic especially, in the situations where the project had a lot of requirement changes (e.g. Scrum project 2).

Face to face Communication

Osmotic communication (Cockburn 2004) is typically accomplished in agile software development projects by seating the project team in the same room. The purpose of this kind of communication is to make the costs of communications low and the feedback rate high so that the needed knowledge is disseminated quickly which, thus, enables more efficient errors correction (Cockburn 2004). In F-Secure, the open office space was seen as a efficient way of improving the communication between the development team. However, in the Mobile-D project the Open office space was seen more disadvantageous than useful and therefore it also decided to revise in the next Mobile-D project. Somehow, it seems that the use of the open office space as well as the osmotic communication is much depending of the personality and background experiences of the software developers.

Daily stand up meetings (Schwaber 1995) in Scrum projects and the wrap up:s in Mobile-D project, were seen as a good practices also from the F-Secure viewpoint. Usually, the daily meeting is a short meeting which purpose is to answer to the following questions: 1) What did I work on yesterday, 2) what do I plan on working today and 3) what is getting in my way (Cockburn 2004). In F-Secure, the daily meetings were disclosed as an effective way for disseminating information and, therefore, to make the problem solution mechanism more efficient in all of the evaluated projects. It should be, however, careful in agile software development projects that the daily meetings are used to identify not discuss of the problems (Cockburn 2004). For example, in Scrum 1 project, the daily meetings were very efficient at the beginning, but took too long time at the end of the project because of discussion of technical problem solutions during the meetings. This problem was appeared when the complexity of the product was increased. In this case, the problem solution should be dealt right after the stand up meeting with only those people that have to be there.

According to the Scrum project experiences, the pair programming (Beck 2000) could be seen as a good practice in the complex coding and reviewing situations as well as learning mechanisms between the junior and senior designers. In Mobile-D project, it was however decided to revise because of the high resistance of some of the developers. In fact, the pair programming has caused much discussion between the software developers and researches all over the world. For example, the Jim Cobien agreed that

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

the full time pair programming would not be optimal amount of time for people spend together (Cockburn 2004). Thus, Alistair Cockburn actually suggests a side by side programming, where the developers are sitting side by side discussing of the problems but still doing their own work (Cockburn 2004).

Testing

In the Mobile-D project post-mortem analysis, the efficient unit testing was seen as a one of the key success factor in the agile software development projects. Automated unit testing was very efficient at Mobile-D project because of the good tool support and high technical experience. In Scrum 1 and Scrum 2 projects the automated testing, however, could be improved for example with the creation of the automated unit testing framework.

In the system testing the key problem was the unclear quality engineering responsibilities related to the agile software development. From the quality engineering (QE) team viewpoint there should be clearly defined requirements of the needed documentation and communication demands (e.g. participation in the daily meetings) between the QE and agile software development teams.

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

6 Improvement idea mapping with agile based solutions

As a conclusion of the agile practice analysis in Figure 10, it can be clarified that the agile software development in all agile projects at F-Secure included at least the following mandatory practices: 1. Agile or common process model selection, 2. Requirements preparation with customer, 3. Iteration Planning including the requirements prioritization, task and effort estimations, planning of the unit and system testing, 4. development based on the short iterations including the daily meetings, as automated unit testing as possible, continuous participative system testing. 5. Frequent delivery to the customer and therefore the customer feedback collection in the projects. Other practices like the open software spaces, side by side or pair programming were alternative practices that were successfully used in some situations (Figure 8).

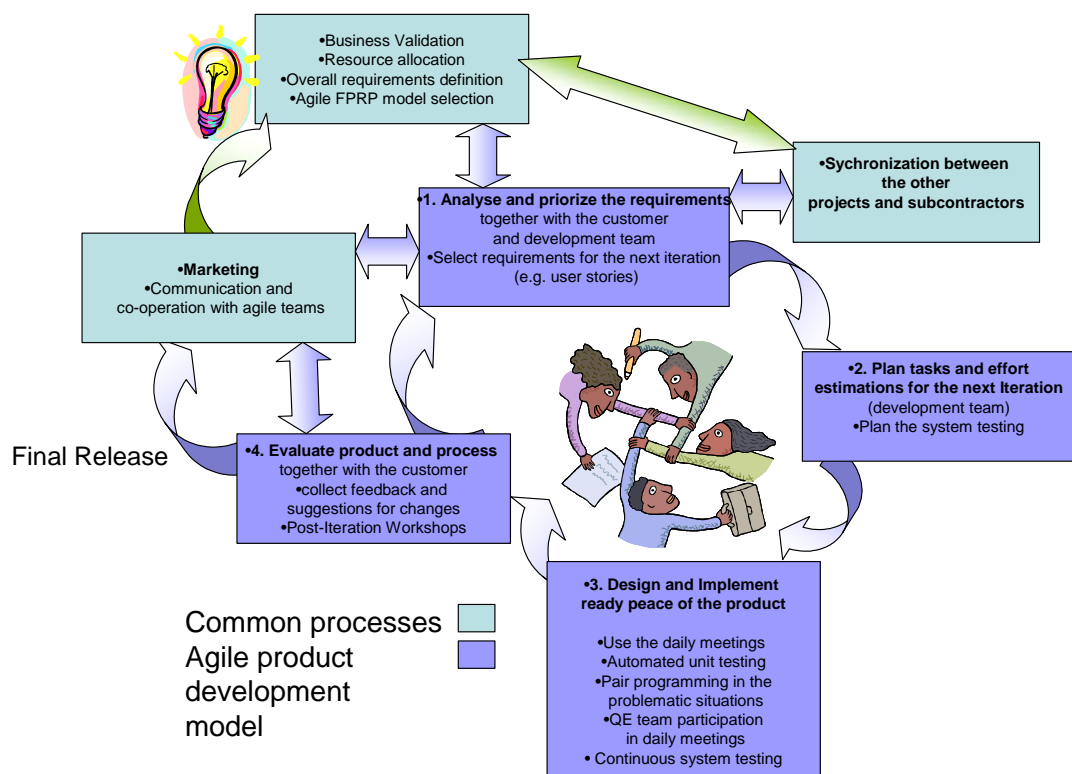


Figure 8. Possible Key steps in agile model at F-Secure

Key purpose in Agile Performance Assessment is to disclose the current weaknesses (i.e. improvement ideas), their solution proposals in agile software development projects as well as possible benefits and weaknesses of the needed changes (i.e. improvements) that could be done for the improvement of the current situation. One of the most interesting findings of the F-Secure agile performance assessment was that most of the problems that were disclosed from agile development projects were not actually indicated to the project team itself. They were more focused to the action of organizational level stakeholders like collaboration between another non-agile project teams, marketing department and business units as well as tool support that would be needed also for the agile software development (especially for requirements and project

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

management). In table 11 the key weaknesses, agile based solution proposals and expected benefits of the change are described.

Table 11. Summary of the improvement ideas in F-Secure

Manage Requirements			
Weaknesses	Agile Solution Proposals	Expected Benefits	Possible Problems
Collaboration between business units and agile projects	Workshop Techniques (e.g. Essential interaction design (Cockburn 2004) or Rapid7 (Kylmäkoski 2003))	Improved co-operations	People in management are often used to their way of action in project preparation phases
New Requirements and schedules after the rapid delivery	Clarification of an idea of the agile development should to the customers	Projects would have more time for agile	Customers still want to keep their viewpoint of the schedule and requirements
Difficulties to handle product requirements during the iteration preparations if all requirements are must	Tool and template support for product backlog and agile requirements management	Savings in project planning effort	Requirement databases will be too complex because of the huge data flow
Establish Estimates, Develop a Project Plan			
Weaknesses	Solution Proposals	Expected Benefits	Possible Problems
Selection of right Agile or Plan-Driven software development model in suitable situations	Check list for development model selection (example: (Kettunen and Laanti 2004)) or Boehm and Turner (Boehm and Turner 2003)	Use of right methods in right situations	Lack of knowledge of agile methods success in different product development contexts
Synchronization between agile and plan-driven projects	Identifying of the development rhythm and communication with non-agile projects	Improved communication	Non-agile projects are not willing to change their way of action in any situations
Unclear responsibilities between agile and QE team	quality team should participate in iteration planning meetings and development phase	Improved quality of the system testing	Resistance of the quality team (they are used to traditional working methods)
Balance between information radiators and metrics collection	Tool support for agile project management	Improved metrics collection	Double work that is not agile anymore
Definition of the external documentation needs	Definition of external documentation needs for agile development projects, documentation e.g. using Rapid 7	Documentation is planned not ad hock activity which are taken too much time from project once	External projects define their needs too late from the agile software development projects viewpoint
Definition of the suitable task sizes for information radiators	Definition of the suitable task sizes in information radiators (date = fixed, features = not fixed)	Improved visibility	Tasks are wholeness that is impossible to deal with enough small tasks
Difficulties to sell and market the product which is iteratively changing	Marketing department should have knowledge of agile methods, agile project current state and its affects to marketing	Improvement communication between agile projects and marketing	Marketing has no possibilities to change their way of action in product selling work

Business validation in Figure 10 as well as requirements allocation has been done mainly using the **common processes**. In the business validation phase the improvement suggestion was that the initial requirements elicitation could be strengthened with the workshop techniques (e.g. Rapid7, Essential Interaction design) that would support the communication between business units and project managers as well as inspection activities already during the document creation. Essential Interaction design (Cockburn 2004) is a workshop technique, which can be applied during the initial requirements elicitation when building the incremental release plans, while defining general form of

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

software, delivering user interface from use cases and even acceptance testing. The overall requirements definition could be also supported with the F-Secure level “feature pool”, which could help the right overall level requirements selection in both agile and traditional SW development projects.

Architecture design in all agile projects at F-Secure was implemented using **the architecture baseline developer by research projects**. Based on the agile assessment workshops it seems that, at least in F-Secure, the architecture should be based on the stable architectural baseline, which can be changed in some situations but **not** radically, during the actual development. This is different approach than the incremental rearchitecture suggested by Cockburn where the idea is to start with simple architecture which is modified during the iterations when needed (Cockburn 2004).

Requirements analysis in Figure 9 in agile projects should be done for prioritizing and analysing the requirements for the next iteration together with the marketing, customer and developers (Beck 2000). One way to maintain requirements is to write the features in the product backlog and then make the more detailed level analysis for using some understandable form of the requirements (e.g. user stories, task cards or use cases). The template of the product backlog content and use purposes should, however, be clarified in F-Secure to support the future F-Secure Agile product development model project requirements or feature management.

Tasks planning in Figure 9 could be done like in the Mobile-D and Scrum projects utilizing either the Information Radiator (Cockburn 2004). Its benefits are 1) visibility, 2) understandable, 3) easily to kept up to date. The problem is only how to achieve the balance between the information radiators laborious metrics collection and its benefits in project. Therefore, it could be alternative practice that the F-Secure Agile product development model projects could be used parallel with some electronic tool support (e.g. Xplanner). Bliz Planning is a workshop technique which aim is to support the iterative project planning in agile software development projects (Cockburn 2004). It might be method that could be piloted in the agile projects task and estimations definition.

Most of the recommendations in traditional project agile assessment workshop referred the *feature driven development* in Figure 9 (Karlsson, Andersson et al. 2000) where the idea is to use daily build, automatic testing, design based on the module owners and interface responsible persons. The feature development is useful for both agile software development projects but also in the larger distributed project teams. Therefore, it can be seen as a solution also in the situations where the agile practices, otherwise, may not be suitable.

Increment results should always be evaluated in Figure 9, at least, in agile projects together with the development team and customers at the end of each iteration. This is what Cockburn writes of the frequent delivery:

Fix the end date and have the team deliver whatever they have completed at the time box. In the very changing project there would be need to lock the requirements during the iteration. This gives the team peace to complete the requirements before the new change requests are appeared from the customers(Cockburn 2004).

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

This, however, does not give solution for the Scrum 2 project problem to get totally new schedule requests from the customer because of the quick product delivering after the first iteration.

Synchronization with the other, especially, non agile projects in Figure 9 was seen as a very large problem in the agile software development at F-Secure. One way to response to this problem could be just to improve communication between the projects. This could be implemented by increasing the traditional development project responsibility person participation in the agile project requirements analysis in iteration planning meeting or in the daily meeting if necessary. Also the release planning and development rhythm planning together with the agile and traditional projects (e.g. using the essential interaction design) would support the co-operation in that case.

Concerning marketing there might occur some positive and negative implications when agile software development is used in projects in an organization. A few of them were appeared in the F-Secure workshops. One good thing, which was mentioned, is that in case of the agile development marketing department might have more live demos to show to the customers already in the early stages of the project. On the other hand, it might be difficult that the features of a end product are not necessarily well known beforehand in agile software development. That can also make the product more difficult to sell and market. The marketing department might also push some features they think are important or sell products and features which are not yet ready or even intended. Marketing might also cut out some features which were considered important by the development team. Therefore, it was mentioned in the workshops that marketing and development departments should work more as a team (not as isolated units) having a knowledge of the used agile methods and their affects to the marketing department. Thus, the communication and co-operation between different teams and marketing departments can also be seen one of the most important success factor in case of the agile software development.

6.1 Possible agile practices for the traditional FPRP

As suggested earlier in the previous section the business validation could be implemented traditional way, however, piloting some new agile workshop techniques (e.g. Rapid7). Traditional development also in the large, complex and distributed projects could be supported with the feature driven development, which actually includes many activities for design, validation and releasing phases. In the feature driven development the components are build daily and tested automatically (Karlsson, Andersson et al. 2000). In the Feature Driven development, 1) the overall scope of the project is set in terms of end date total resources and direction (only new features and system improvements) and 2) the current list of features, their status and estimated costs are used as a steering mechanism. According to the Karlsson, the feature based development has a clear relation to some XP (Beck 2000) practices. For example, the planning game, small releases, automatic testing, collective code ownership, continuous integration in Table 17 could be used also with the feature driven development in the larger software development projects (Karlsson, Andersson et al. 2000).

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

Table 12. Possible agile practices for FPRP projects

Phase	Possible agile practices
Business validation	Blitz Planning (Cockburn 2004), Essential interaction design (Cockburn 2004) or Rapid7 (Kylmäkoski 2003), Product Backlogs in (Schwaber 1995).
Development	Feature Driven Development/ Design based on module owners (Karlsson, Andersson et al. 2000), Daily meetings, Side by side or pair programming in complex situations
Validation	Automatic unit testing (Karlsson, Andersson et al. 2000)
Releasing	Daily Build (Karlsson, Andersson et al. 2000), Frequent delivery (Cockburn 2004)

Communication between the software development teams is important success factor in both agile and traditional software development. The communication could be made more efficient also in traditional software development projects, for example, piloting some agile practices such as daily meetings, open office space for the feature teams.

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

7 Summary

The purpose of this document was to analyse and compare the agile practice use experiences in the Mobile-D and Scrum projects in F-Secure. The data for the F-Secure agile assessment was collected from four selected case projects (Scrum 1 and 2, Mobile-D and traditional projects) both using qualitative data (e.g. the agile assessment workshops, focused interviews, observation in the further project planning meeting) and quantitative data (e.g. project metrics). In addition, in the Mobile-D case, also, the validated post-iteration workshop and postmortem data (worksheets and action point lists) was used.

The *Scrum 1 project* is focused on developing a security system management tool. The project had been going on for over a year and the Scrum method itself is used in the project about five months until now conducting three Sprints. Another Scrum project, called *Scrum 2*, is a research project, which aim is for developing antivirus software. The idea of the Scrum method piloting was based on Scrum 1 project excellent Scrum method utilizing experiences. The project team has been operated about four months and the Scrum method had been successfully utilized in the project about for three months until now. The goal of the *Mobile-D project* was to develop a mobile security application using the Mobile-D method. The project team conducted five software development iterations. Traditional comparison project included a 55 project developers. It was started about years ago and its goal is to develop a new version of virus protection system for the consumers.

The aim of the agile assessment in F-Secure was 1) to support the F-Secure Agile product development model creation alongside with the traditional F-Secure Product Realization Process (FPRP) model, 2) to establish what agile practices could be utilized in the traditional SW development projects and 3) analyse when the F-Secure Agile product development model should be used in F-Secure projects. The content of the F-Secure Agile product development model as well as the possible agile practices in traditional FPRP were analysed based on the agile practice use experiences in Scrum and Mobile-D projects. The situations when to use agile model, instead, were evaluated via success and failure factors in both literature references and experiences of the different type of F-Secure projects.

According to the findings, all of evaluated agile projects included the following mandatory practices: 1) Requirements preparation (e.g. quality criteria) with customer and development team in iteration planning, 2) task definition and effort estimations by the development team, 3) simple design and implementation (e.g. using the mechanisms feature driven development), 4) daily meetings, 5) automated unit testing, 6) QE team participation and 7) continuous system testing. In addition, the increment results should be evaluated together with the customer after each development iteration. In the common FPRP, instead, the possible used agile practices would be 1) the workshop techniques in the overall requirements creation as well as 2) the techniques of the feature driven development and 3) agile communication such as automated unit testing and daily meetings. *Architecture design* in all agile projects at F-Secure was implemented using the architecture baseline developer by research projects. Based on the agile assessment workshops it seems that, at least in F-Secure, the architecture

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

should be based on the stable architectural baseline, which can be changed in some situations but **not** radically, during the actual development.

The empirical data from the F-Secure projects shows that the key success factors evaluated in agile software development in F-Secure were, for example, the feedback from customer, daily communication and documentation just based on external needs. It also seems that the common FPRP model is more useful in the situations where the project team is large, distributed and having much external sources and projects. However, some agile practices such as Feature Driven Development and workshop techniques (e.g. Rapid7, Blitz planning and essential interaction design) could, however, be useful also in this kind of situations. One of the most difficulties task in the agile software development at F-Secure is the agile software development approaches synchronizing with the other non-agile projects and marketing.

In the future, the research activities between the F-Secure and VTT could be focused on 1) the product usability factors in agile SW development and 2) the possibilities to achieve the required CMMI levels also in the agile software development projects. Also, the agile deployment support (Post-iteration workshops) and new agile assessment including the new agile pilot projects experiences (workshops together with the agile product development teams) and the business agility viewpoints (e.g. workshops with the marketing and customers of the agility demands) would be needed.

Authors: Minna Pikkarainen, Kaisa Komulainen, Tapio Matinmikko

Version: 1.0

Date: 19.10.05

References

- Abrahamsson, P. (2000). Measuring the Success of Software Process Improvement: The Dimensions. EuroSPI, Copenhagen, Denmark.
- Abrahamsson, P., A. Hanhineva, et al. (2004). Mobile-D: An Agile Approach for Mobile Application Development. 19th Annual ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'04), Vancouver, British Columbia, Canada.
- Beck, K. (2000). Extreme Programming Explained: Embrace Change, Addison Wesley Longman, Inc.
- Beck, K., M. Beedle, et al. (2001). Manifesto for Agile Software Development. **2002**.
- Boehm, B. and D. Turner (2003). "Using Risk to Balance Agile and Plan-Driven Methods." IEEE Computer: 57-66.
- Boehm, B. and R. Turner (2003). Balancing Agility and Discipline. Balancing Agility and Discipline -A Guide for the Perplexed, Addison Wesley.
- Boehm, B. and R. Turner (2003). Using Risk to Balance Agile and Plan-Driven Methods. Computer, IEEE Computer Society.
- Cockburn, A. (2004). Crystal Clear, A Human-Powered Methodology for Small Teams.
- Karlsson, E.-A., L.-G. Andersson, et al. (2000). Daily Build and Feature Development in Large Distributed Projects. Conference on Software Engineering (ICSE 2000), Limerick, Ireland.
- Kettunen, P. and M. Laanti (2004). "How to steer an embedded software project: tactics for selecting the software process model." Information and Software Technology.
- Kylmäkoski, R. (2003). Rapid7 Method for Document Authoring.
- Pikkarainen, M., O. Salo, et al. (2005). Deploying Agile Practices in Organizations: A Case Study. EuroSPI 2005.
- Reel, J. S. (1999). "Critical Success Factors In Software Projects." IEEE Software: 18-23.
- Salo, O., K. Kolehmainen, et al. (2004). Self-Adaptability of Agile Software Processes: A Case Study on Post-Iteration Workshops. 5th International Conference on Extreme Programming and Agile Processes in Software Engineering (XP 2004), Garmisch-Partenkirchen, Germany, Springer.
- Schwaber, K. (1995). Scrum Development Process. OOPSLA'95 Workshop on Business Object Design and Implementation, Springer-Verlag.