ETSF01: Software Engineering Process Economy & Quality

Project in Software Project Management

Elizabeth Bjarnason with input from Eva Jönsson
Lund University, Dept. Computer Science
March 8, 2016

Contents

1 Introduction

This document gives the practical details regarding the project in the course ETSF01 Software Engineering Process – Economy & Quality, hereafter denoted course project. The main objective of the course project is to gain a deeper understanding of how software project management (SPM) is performed by working with two realistic case projects.

The course project is equivalent to 2 ECTS credit points (approx. 50 man-hours per person). Each course project team consists of six people, who are jointly responsible for the course project. All course project members should be involved and the total effort should be evenly distributed among participants.

2 Learning Objectives

The main learning objectives of the course project are to:

- Connect course content to practice (see Course plan. 'Färdighet o Förmåga')
- Gain experience of software project management for own course project and for given case projects
- Gain experience of assessment and evaluation for different types of software development projects using software metrics
- Provide a group-learning setting focused on a realistic project setting
- Practice technical writing by presenting information in a structured way, both written and oral.

3 Course Project Description

The main aim of the course project is to perform an evaluation of SPM tools for different types of software projects and development models, in particular the two software project types (cases) *Application development* and *Software porting* (see Section 6). The work consists of the following parts:

- Design of an evaluation framework for SPM tools
- Assessment of 3 different SPM tools for managing the software development within one large company by
    - simulating the performance of the SPM activities for each of the two project cases
    - applying the evaluation framework and thereby measuring various aspects of the tools
- Analyse the obtained measurements, i.e. the evaluation results, and identify the benefits and weakness for each assessed tool for each of the two project types
- Provide tool recommendations for the two software project types based on an analysis of the evaluation results.
- If any of the SPM activity areas have no or insufficient support in a tool, provide recommendations for suitable tool improvements
- Reporting of the outcome as a written report and through an oral presentation

3.1 Evaluation Method

The tool evaluation shall be performed the following method that consists of four steps, see Figure 1 for an overview.

Step I. Investigations and Design

In this step 2 fictive case projects are to be defined (one per project type), 3 SPM tools selected, and an evaluation framework is designed. ***Fictive but realistic case projects*** are to be defined for each evaluated project type, and then used to explore tools and to identify SPM needs for each project type. An example could be an *application development* project tasked with developing a new media player version for which activities, resources etc are 'invented'. General case information is provided in Section 6 and details for each SPM areas is provided during the lectures.

Explore the tools and the SPM needs (requirements) for each project type by simulating SPM activities for the case projects, e.g. by creating sample plans, assigning resources, creating reports etc. An initial exploration will support selecting suitable tools to evaluate, and a more thorough exploration will support the design of the evaluation framework.

When ***selecting 3 tools***, first perform an initial explorations of available SPM tools. The tools to include in the evaluation are selected by considering the project types, their processes and need for SPM tool support, thereby ensuring that realistic tool candidates are included. You are free to choose any SPM tool but consider the following:

- Access: You need to be able to use the tool including saving project plans etc., so look for free or trial versions.
- Sufficient SPM support for the case projects that you define (see Step I of Section 3.1). Make a rough initial assessment of the tools for SPM activities and the software development projects which you are to evaluate for.
- Available documentation: Good descriptions incl. tutorial will support you in the evaluation. Your exercise tutor will not provide technical support on the tools.

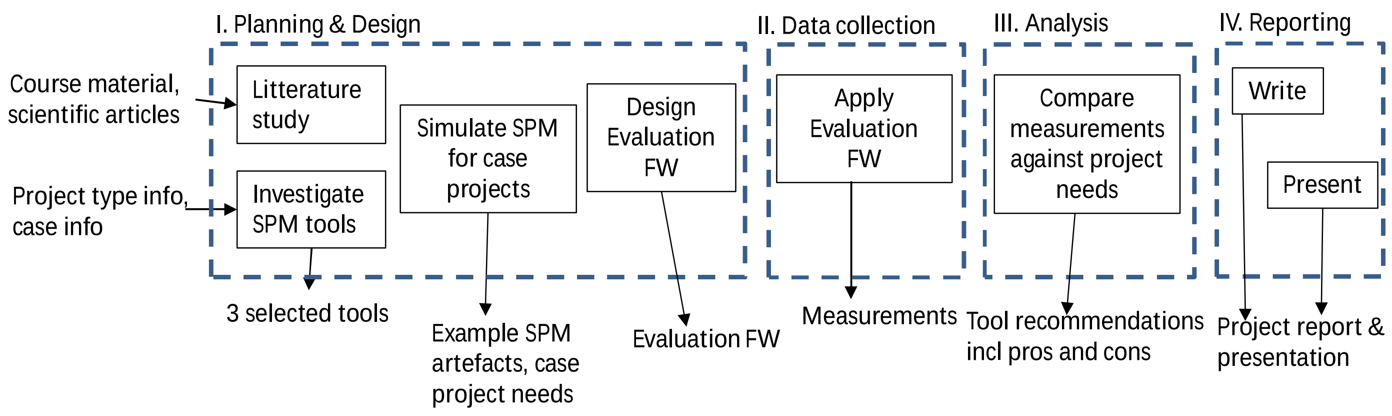
I. Planning & Design
II. Data collection
III. Analysis
IV. Reporting
Course material, scientific articles
Project type info, case info
Litterature study
Investigate SPM tools
Simulate SPM for case projects
Design Evaluation FW
Apply Evaluation FW
Compare measurements against project needs
Write
Present
3 selected tools
Example SPM artefacts, case project needs
Evaluation FW
Measurements
Tool recommendations incl pros and cons
Project report & presentation

Figure 1. The method used to evaluate SPM tools for the case company.

The following are examples of tools previously selected for this assignment:

- 2-plan
- Aceproject
- Apache Bloodhound
- Assembla
- Basecamp
- Bug-Genie
- Clarizen
- Collabtive
- Feng Office
- GanttProject
- Gemini Tracker
- MS Project (covered by LU license)
- LibrePlan
- OpenERP
- Project Open
- TACTIC
- Teamwork
- Trello

The ***Evaluation framework is to be designed*** based on the course theory and on the case needs, and guided by your experience of simulating SPM for the case projects. The framework consists of measurements for relevant factors and is further described in Section 3.2.

Step II. Data collection

The evaluation framework is applied to each of the selected SPM tools resulting in measurements for the aspects included in the framework, i.e. data indicating how well the tool meets different requirements.

Step III. Analysis

Analyse the collected data, i.e. the measurements, and compare against the needs of the case projects. Based on this analysis, identify strengths and weaknesses for each tool and recommend one tool for each project type (the one that performs the best for that project type overall factors). Two types of recommendations shall be given:

a) tool recommendations for the given software development projects (described in Section 6), and

b) tool improvement suggestions for missing or weak SPM support.

These recommendations are given through the written report, see Section 3.3.

Step IV. Reporting

The outcome of the course project shall be reported in a written report and in an oral presentation. The reporting shall cover the evaluation framework and the results from applying this, tool recommendations per software project type based on the evaluation framework, and if relevant, recommendations for tool improvements. Details on the reporting can be found in Section 3.3. The work and the reporting is to be performed iteratively and expected deliveries are outlined in Section 4.

3.2 The Evaluation Framework

Each course project shall design a framework for evaluating software project management tools. This evaluation framework shall cover the following 5 SPM activities:

- activity planning
- effort estimation
- risk management
- resource allocation
- monitor & control of project execution

For each SPM area, suitable factors to evaluate are to be identified based on the course material, e.g. the textbook, and on the two case projects (defined by the project group, see Step I above). The evaluation framework should cover SPM tool requirements relevant for the case projects concerning, e.g. project size & duration, process, dependencies to other projects etc. Case information is provided in this document and during lectures for each SPM area. In addition, relevant quality aspects for the tools should also be included in the framework for each SPM area, e.g. usability, capacity, reliability (see ISO9126, Hughes p. 298).

Measures shall be defined for assessing the identified evaluation factors. That is, for each included factor, detailed evaluation criteria should be defined. The framework should include both objective and quantitative measures (e.g. maximum number of activities) and subjective and qualitative measures (e.g. ease of use, suitability for activity). The Goal-*Question-Metric (GQM) method* [P1] should be used to define the measures.

3.3 Course Project Reporting

The project shall be reported through a written report and an oral presentation.

3.3.1 Written report

The report shall be written in the IEEEtran Proceedings Format[1] using maximum 7 pages for the main report including tables, figures etc. to present the main message of the report, additional pages may be used for appendix. Note that the grading will be performed on the main part of the report and not on the appendix. Thus, the main parts must contain sufficient information to meet the assessment criteria, see Section 5. The appendix may be used for additional information that is not vital for the main report, but that provides additional details, e.g. screen shots of resulting case plans and reports. The report must be original work by each student group, and adhere to the department's guidelines on plagiarism[2]. The final project reports will be checked via Urkund[3].

The main report shall contain:

- Title and authors

[1] http://www.ieee.org/conferences_events/conferences/publishing/templates.html Use US Letter format. For LaTeX: use the LaTeX class file IEEEtran v 1.8 (http://www.ctan.org/tex-archive/macros/latex/contrib/IEEEtran/IEEEtran.cls) and the following configuration (without option 'compsoc' or 'compsocconf'): \documentclass[conference]{IEEEtran}
[2] http://cs.lth.se/english/education/cooperation_or_plagiarism
[3] http://www.urkund.se

- Abstract: A brief summary of all parts of the report; context, addressed topic, what you've done, outcome, potential application of outcome. The main purpose of the abstract is to attract people to read your report.
- Introduction section: An introduction to software project management and to your course project including a description of the evaluated tools. At least 2 scientific publications are to be referenced appropriately in the introduction of SPM. The purpose is to introduce the reader by providing contextual information.
- Method section: A description of the method you use to conduct the evaluation. Describe the activities performed during the evaluation and dependencies between them. Examples of activities are tool selection, analysis of case project types, design of framework etc. Motivate your choices, e.g. why select these tools, why these evaluation factors etc. These are important aspects of your method. The description is ideally complemented by an overview picture that shows your activities, their input and output and relationships between them. In addition, consider the limitations of your evaluation and steps taken to increase the validity of your findings. For example, how reliable are your recommendations, to which extent are the results valid also for other cases and project types?
- Case Projects: A description of the project types for which the tool is evaluated. The description shall include both a high-level description (e.g. project context and main challenges) and a description of the case projects that you have designed, including examples of plans, resource schedules etc. The main purpose is for you to report on the characteristics of the projects which you have considered in your evaluation. In addition, this section is to provide the reader with necessary background information to follow the motivation for the design of the evaluation framework and the tool recommendation.
- Evaluation Framework: A description of the evaluation framework including the assessed factors and measures of these. References to scientific publications can be used to motivate your choice of factors.
- Tool Evaluation and Improvements: The results of the evaluation based on the evaluation framework. Report the results per tool and for each SPM area including improvement recommendations for SPM areas for which there is none or very weak support. The purpose is to describe the assessment results and discuss the strengths and weaknesses of each tool, and to motivate the given tool improvement suggestions.
- Tool Recommendation per Project Type: For each of the two evaluated projects types discuss the strengths and weaknesses for each tool based on the evaluation framework and for each SPM area, and recommend one (overall) tool for each project type. The recommendations shall be clearly based on the outcome of applying the evaluations framework and motivated by the characteristics and needs of the specific case projects.
- Conclusions: main conclusions of the report. The purpose is to summarise the main points in the report.
- References: list of references used in the report

3.3.2 Conference presentation

Each course project group is to present the results for one SPM activity area, e.g. effort estimation. This is done on the last exercise session of the course. Your exercise tutor will select the SPM activity area you are to present.

The presentation shall cover, for the given SPM activity area (if nothing else is stated)

- the evaluation framework and a brief description of your case projects' needs.
- a brief introduction to the evaluated tools including all areas. Mention strong/weak areas and outline tool improvement recommendations.
- for each of the evaluated case projects and tools, what are the strengths and weaknesses relative the needs of the case project.
- Overall (considering all SPM areas), which tool is recommended for each project type? The recommendations shall be motivated by evaluation results and case project characteristics.

4 Deliveries

The course projects will use an iterative approach with 3 incremental deliveries: drafts 1 & 2 and final report. The evaluation framework and the report are gradually extended as each SPM activity area is covered in the lectures. You will receive feedback on drafts 1 and 2 at the scheduled exercise sessions (exercise 2 and 3). Each group is to gather this feedback, analyse and take appropriate action in updating their report. This information (feedback and actions taken) is to be noted in a progress log for draft 2 and for the final report.

The course project has the following three deliveries (see course description for deadlines):

1. **Report draft 1**
   Outline of the report in IEEE format with heading levels 1 and 2, and at least short sentences (in your own words!) of the planned content for each section. The report shall contain opening paragraphs in the Introduction section (as discussed in Exercise 1), an initial description of the select tools (in Introduction section) and of the project cases (in section on Case Projects), a draft of the method planned to be applied (see Section 3.1) and a first version of the evaluation framework covering at least
   a. activity planning
   b. effort estimation
   This version of the report will be peer reviewed at Exercise 2 (SPA I), see instructions for exercises.

2. **Report draft 2 and Progress log**
   The report contains a full draft of all sections in IEEE format. The Evaluation framework is described for all SPM areas and has been used to assess the tools. The report contains initial results, and recommendations for tool selection and tool improvements based on these. The progress log contains the feedback provided at SPA I (Exercise 2) and the group's action log based on the provided feedback (see Section 4.1.1).
   This version of the report will be peer reviewed at Exercise 3 (SPA II), see instructions for exercises.

3. **Final report and Progress log**
   Full report covering all requested points, see Section 3.3. The final report will be graded according to the criteria found in Section 5. The progress log contains the feedback provided at SPA II and the group's action log based on the provided feedback (see Section 4.1.1).

   TIP: Use the check-lists providing for the SPA exercises within your project group to review and improve your own final report. The check-lists are designed to align with the grading criteria.

All hand-ins shall be in pdf format. Name the pdf file <Group ID> + {'v1'| 'v2' | 'final' } and submit as follows:

- **Draft 1 and draft 2:** submit draft of report and, for draft 2 progress log (see Section 4.1.1), via moodle.cs.lth.se. Instructions will be provided on the course web page.
- **Final report:** submit report and progress log to etsf01@cs.lth.se and etsf01.lu@analys.urkund.se . **NOTE**: The subject line **must contain exactly**: 'Report' + <Group ID> + <student IDs of all group members>. For example, 'Report Group A3 ain09aha jur10eib …'

4.1.1 Progress log

In order to show how you progress in your course project work and in particular in your writing, a progress log is to be provided together with draft 2 and your final report. The progress log shall contain a summary of received SPA feedback and a description of how you have improved your report for each of the points in the provided template.

The progress log is mandatory, but does not affect the grading.

The progress log should be submitted as an excel file based on the corresponding progress log template.

5 Course Project Assessment

The report (excluding any appendix) is graded fail–pass (U, G), and up to 10 bonus points can be awarded.

	Minimum requirement for Pass	Criteria for bonus	Bonus points
Scope of work, see Section 3.	All content requested in this project description is included.		
Form			
Use of IEEE template including formatting of pages, title, names, headings, paragraphs, table and figure captions, references, etc.	Correct template used and page limitations observed.		
Structure, see Section 3.3.1.	In-line with project description.		
Language and Writing: spelling/grammar, clarity and appropriateness of expression, correct use of terminology (no slang!) See exercise 1.	Good, clear and correct use of language with some flow including Top-Down structure.	Excellent flow of text and arguments including use of standard moves in Introduction. Excellent use of references.	2
Report Content, see Section 3.3.1			
Introduction, see exercise 1	Appropriate use of (at least) 2 scientific references to describe SPM and the 5 SPM areas.	Excellent description of SPM and 5 areas including main challenges and benefits, and using several (more than 2) scientific references.	1
Evaluation Method, see Section 3.1	Good description of how you have conducted the evaluation.	Excellent description of the applied method and steps taken to mitigate threats to validity.	2
Case Projects, see Section 6.	Good description of the SPM needs for the two included software project types i.e. *Application Development* and *Software Porting*, through one case example per type. Include example schedules and reports etc. in the appendix.	Excellent description of the two case projects, including project context, SPM needs and challenges.	2
Evaluation framework, see Section 3.2.	Appropriate properties defined for each of the 5 SPM areas including quality characteristics. Appropriate measurements including scales for each property; a good combination of subjective and objective assessment.		
Tool evaluation and Improvements, see Section 3.1	Clear reporting of the evaluation results for each tool. For tools without support for certain SPM areas, motivated tool-specific improvement suggestions are provided.		
Tool recommendation per Project type, see Section 3.1**Error! Reference source not found.**	For the two project types, a tool recommendation is given for each project clearly motivated by evaluation results, and based on a discussion on pros and cons.	The tool recommendations per project type are excellently motivated, and discussed based on the evaluation results and on the case project characteristics and needs.	2
Oral presentation, see Section 3.3.2			
Clarity of message & adapted to audience, see Exercise 1.	Clear and understandable.	Excellent clarity, audience contact and interest grabbing. Clear evidence of use of the classical rhetorical model.	1
Timeliness	An overview of the work performed and the main conclusions are presented within the given time frame.		

6 Company Description: DauMob Ltd

DauMob Ltd is a whole-owned subsidiary of a company producing consumer devices. DauMob has around 4,000 employees and develops mobile devices such as mobile phones and tablets for a highly competitive and fast moving consumer market. The customers consist of the end users (anonymous consumers) and of key customers. The key customers are network operators that purchase customised product versions to sell to their own customers together with subscriptions to their mobile networks. The products are developed on the Android OSS platform and the hardware components are provided by external vendors. The majority of the company's development effort is spent on software development including internal software development and using software components from third party vendors. The company is experiencing challenges in managing their software projects and is looking for new and improved tools that can support them in managing the complex reality in which they develop software.

There are several types of software projects that span the life time of a product from the conception of a product idea to the launch and eventual death of a product. For example, a new idea can be explored in a *proof-of concept project* or prepared for development in a *pre-study project*. There may be internal *hardware projects* that develop and adapt hardware to the company's product lines. When the hardware and middleware layers are updated a *software porting project* is responsible for adapting the existing software layers to run on the new underlying layers. *Application development projects* are used to develop new software functionality both as applications embedded in the device and stand-alone (downloadable). The *SW platform projects* are very large projects that tie together HW and MW versions and bundle it with the set of available applications (developed by App Dev projects). A SW platform project is responsible for developing a new major release of the SW platform (see Figure 2), and includes one SW porting project and a number of application development projects. The software produced by the software platform projects is combined with product-specific hardware in *product projects* that are responsible for the individual consumer products. Once a product is released to the market the responsibility for upgrades and bug fixes is handled by a *maintenance project* until the decision is taken to no longer support the product, i.e. product death.

There are dependencies between all these software projects. In addition, there are also dependencies to an internal HW project responsible for the hardware platform, and to external software suppliers, which supply component to both the software platform and to applications. Furthermore, the software projects deliver to product projects and to the companies (on-line) SW update centre. See figure 1 for an overview of the software project types and dependencies.

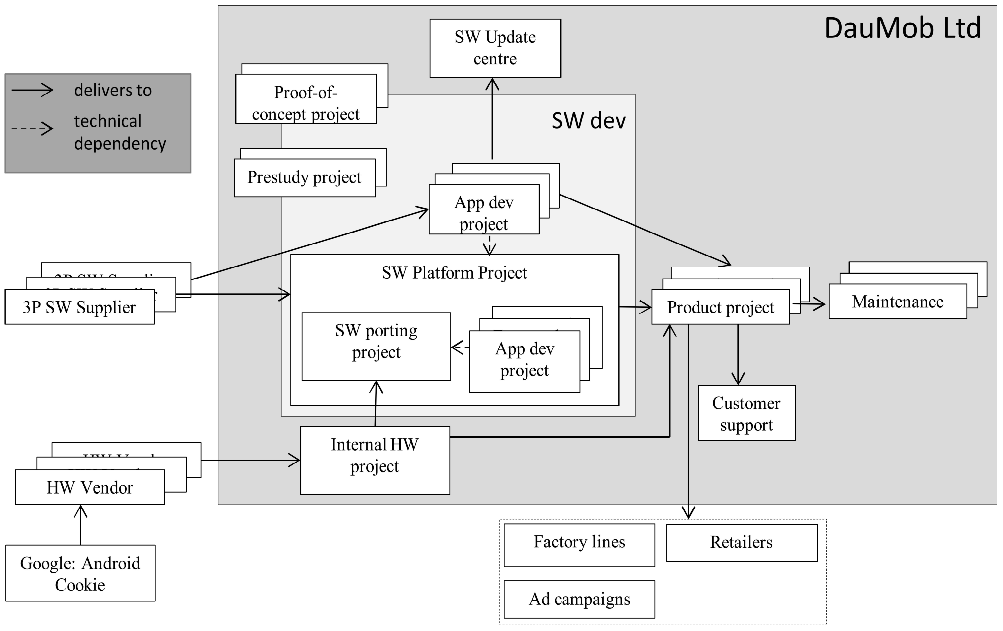

Figure 1. An overview of DauMob's main software development projects including dependencies.

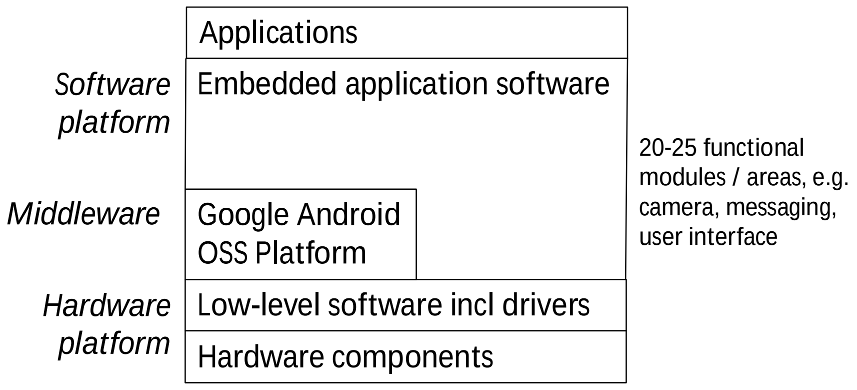
Applications
Software platform
Embedded application software
Middleware
Google Android OSS Platform
Hardware platform
Low-level software incl drivers
Hardware components
20-25 functional modules / areas, e.g. camera, messaging, user interface

Figure 2. An overview of the architectural layers.

The product architecture consists of a hardware layer, a middleware layer, an embedded application layer and applications, see Figure 2. The hardware layer, a.k.a. hardware platform, is made up of hardware components, drivers and low-level software. The middleware layer consists of the Google Android OSS Platform. The embedded application layer contains applications that in addition to the Android functionality also access the hardware platform, e.g. native camera or display functionality, and can thereby provide more integration functionality than pure Android applications. This layer contains software for around 20-25 different functional areas, e.g. user interface framework, media player, messaging, communication etc. Finally, there are applications that run on the Android operating system.

For this course project, we will focus on two projects namely on a software porting project and an application development project.

6.1 SW Porting Project

The *SW porting project* ports the previous version of the software platform to the new HW platform version delivered by the internal HW project. The internal HW project is responsible for selecting and composing hardware components purchased from external HW vendors into a company HW platform including low-level software for drivers etc. for the product line and for each product. This HW platform contains a basic (google vanilla) version of the Android OSS platform as provided by the HW vendors. The SW platform bring-up project applies the relevant patches to this Android OSS implementation. These patches are company-specific and either fix bugs in the Android implementation or tweaks the behaviour to match the company's specific requirements, e.g. for camera behaviour.

This software sub-project mainly applies a phase-based development model with the following phases: architectural design and planning, implementation (porting), system testing including software certification, and maintenance. No explicit requirements phase is performed rather requirements are managed and discussed during the design phase. In addition, the requirements that this project aims to deliver are the ones supported by the previous version of the application software platform, but on a new version of the Android OSS platform.

This project type aims to minimize the lead time and thereby the time-to-market of the company's products. An incremental delivery model is applied with 2 pre-releases, before the main release of the HW platform. This is followed by bi-weekly maintenance releases during the maintenance phase. The design and planning phase has a lead time of 2-3 months, while the implementation phase is approx. 3-5 months. The maintenance phase starts after the first full release. This phase lasts until all the projects to which the SW porting project delivers are completed in order to support these projects with HW issues including bug fixes.

The project is staffed by 2 project managers, 1 requirements coordinator, 1 senior architect, 1 integration and configuration manager, 1 system test leader and 1 quality coordinator. These roles coordinate and lead the work in the project. There are people working on porting each of the 20-25 software areas of the software platform. For each area the following roles are involved in this porting project: 1 team leader, 1 product owner (responsible for any requirements impact that might surface) and 1 architect that all work part-time, and 1-3 developers. The developers are responsible for porting the application software for their area and for function testing it.

The design and planning phase involves analysing the impact of the new Android version on the previous software platform version. For example, functionality previously supported in proprietary SW platform code may now be available in the Android OSS, partly or fully, thus requiring changes to the SW platform code. When previously supported functionality is affected in a non-trivial way the product owner of the affected technical area is consulted to also include business / requirements aspects on the final decision.

The porting effort per software area is estimated and planned during the design and planning phase. This includes identifying the dependencies between the areas, as well as, to the intermittent releases available from the Internal HW project before the final version of the HW platform is delivered. The identified dependencies then affect the planning with the aim of minimizing the risk of delays and the risk of chains of delays.

The execution plan for the SW porting project including the pre-releases is an important input to the planning for the SW platform and feature development projects. As implementation phases progresses the progress is tracked and reported on a weekly basis to the projects to which the SW porting project delivers. The project managers continuously deal with occurring delays and risks, and proactively try to avoid them and to minimize the impact of these.

6.2 Application Development Projects

The *application development projects* develop, test and deliver new or extended software functionality, e.g. a new media player, or a pre-loaded game. These project can either deliver pre-installed applications to be bundled with the consumer product or as downloadable applications. This makes it possible to meet late customer requirements and market needs by quickly providing the requested functionality, sometimes even after the products have been released through applications downloadable through the company's SW update centre and Android market. In this way application development can provide an important competitive advantage.

These projects apply an agile development process inspired by the Scrum method. Each application development project team has the full responsibility for detailing the requirements, designing, implementing and testing a feature. The Scrum iterations (sprints) in which the implementation takes place are preceded by a design and planning phase. In this phase, the main user stories of the feature are agreed and prioritized, and technical dependencies are identified. The user stories are placed in the product backlog in the order agreed for implementation and a high-level plan is made indicating when the application will be completed. This plan is provided to the project managers of the SW platform project to which the application development project delivers.

The implementation is performed in sprints (iterations) of 3-5 weeks. Before a sprint starts user stories at the top of backlog are selected to fill up the work load of the sprint. The user stories are then detailed within the feature project, and implemented and tested.

An application development project varies widely in both size and lead time. The entire project can have a lead time of 9 weeks to 2 years. The following roles are involved in an application development project: 1 product owner, 1 project sponsor, 1 scrum master (project manager), 1-20 developers who are also testers, and possibly one dedicated tester. The product owner represents the customer and is responsible for requirements and scope decisions including clarifying what the customer wants. The project sponsor is responsible for ensuring sufficient resources to execute the project. The developers and the tester are responsible for iteratively detailing the requirements for each sprint in collaboration with the product manager, and develop and verify software that meets those requirements. The project manager is responsible for planning, risk management and monitoring the project, and to communicate the status to the SW platform project. The reporting to the SW platform project is done on a weekly basis and includes progress, remaining effort and risks of delays or quality issues.

When an application project is initiated an initial investigation (1-2 weeks) is performed to identify high-level requirements and rough effort and time estimates. In some cases the business requirements are very high-level, e.g. a tetris-like game, and the development team is free to implemented such a game without detailed discussion with the product owner. In other cases the product owner has specific input to the detailed requirements. The product owner thus act as the customer in requirements discussions and decisions, so called proxy customer. The requirements are validated through demos with the product owner.

The plan mainly consists of number of resources and delivery dates for main and possible pre-releases for validation purposes. Since time-to-market is a critical aspect of these projects, requirements scope is reduced whenever the delivery dates are at risk. The plan is communicated to the SW platform projects and to the product project for which the application is developed. Progress is tracked on a daily basis within the application development project, and reported to the receiving projects on a weekly basis.

6.3 Overview of Case Project Types

	Scope	*Development method*	*Plan*	*Organization*
Proof of Concept	Make a prototype and pitch a new idea towards the business unit.	Optional, though if pure software then agile is preferred. If very hardware dependent then a phase-based plan is the choice.	Informal planning dependent on the selected method. At least an initial project specification and end of project report.	Very small team often with the development engineer as project manager. Team members are selected dependent on the system impact of the innovation.
Pre-Study	Invent a new feature proposal. Focus on an idea and not necessarily to implement it. The step after this could be proof of concept or a feature development project.	Brainstorming, workshop and analysing trends.	Part of an Innovation Sprints or extraction of Roadmap plans. Initial Project specification and End of project report.	Very small team just a few engineers.
SW Porting	Upgrade existing product with new HW and a new OS release. The requirements are defined by the existing product.	Waterfall, phase-based model	Incremental release time plan with focus on time to market.	Medium size with a project team, which has defined roles and responsibilities
Application Development	Develop, test and deliver a specific feature for example a Media Player Application.	Agile, Scrum method	Scrum iterations (sprints) and Product Backlog plan with user stories in priority order. High level time plan.	Small size with dedicated product owner and developers focusing only on this feature.
SW Platform	The scope is to combine a porting project with many feature development projects. The business unit provides the project with prioritized requirements as input to the different feature backlogs.	The method is both phase-based with embedded Scrum based feature development teams.	Traditional time plan with synchronization gates to align with the outcome of all the embedded feature projects.	Large project consisting of sub projects such as a SW Porting Project and many Application development projects. A team of sub-project managers.
Maintenance	Responsible for maintenance releases and bug fixes of a release product.			