Metropolia

Sammy Loitto

Agile in Waterfall

Improving the Flexibility of Product Development

Helsinki Metropolia University of Applied Sciences
Master’s Degree
Industrial Management
Master’s Thesis
25 August 2012
Instructor: Marjatta Huhta, DSc (Tech)

Abstract

Author(s) Title Number of Pages Date	Sammy Loitto Agile in Waterfall: Improving the Flexibility of Product Development 96 pages + 5 appendices 25 August 2012
Degree	Master's degree
Degree Programme	Degree programme in Industrial Management
Instructor	Marjatta Huhta, DSc (Tech) / Principal Lecturer

This Thesis concentrates on improving the existing Waterfall-based product development model used in the case company, to meet the changing needs of product development. Product development is more and more based on standardized and open source components and competition in any industry has increased as the entry barriers has been lowered. Therefore adjustments were needed to enhance the flexibility of product development and shorten the time to market for new products and features.

The research approach applied in this study is action research. The model development is done in iterations, in two action research cycles. The data used for the development of the model are collected in the interviews, discussions, a brainstorming session, and a Kaizen workshop in the case company. The workshop participants were selected to get a broad perspective with people from different departments, including product business management, project office, quality management, system architects and system engineering. Altogether 21 persons give their view on product development and the new model creation.

The model development is done by creating and verifying two prototypes, Prototype 1 and 2, which were developed in several workshops (Workshop 1-4) with subject matter experts and key stakeholders. The prototypes and verification of the prototypes led to the proposal of the final model to the case company. The outcome of the Thesis is a proposal for a new product development model based on Agile development principles, combined with the required tools to meet the targeted levels of quality and management visibility applied in the case company.

Key words	Product development process, Agile, Waterfall, software development, quality, management visibility, Scrum, Scrumban, Kaizen

Contents

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

ASD	Adaptive Software Development
AR	Action Research
BMS	Business Management Suite
CE	Concurrent Engineering
CMMI	Capability Maturity Model Integration
CMMI-DEV	Capability Maturity Model Integration for Development
DoD	Department of Defense
DSDM	Dynamic Systems Development Method
FDD	Feature Driven Development
GMR	Gate Maturity Review
IIDD	Incremental and Iterative Design and Development
IPD	Integrated Product Development
KPI	Key Performance Indicator
NPD	New Product Development
OSSD	Open Source Software Development
PDB	Product Decision Board
PDM	Product Development Model
PFU	Project Follow-up
PRR	Project Redirection Reviews
RAD	Radical Application Development
RFx	Request for Information/Proposal/Quotation
RUP	Rational Unified Process
R&D	Research and Development
SE	Simultaneous Engineering
SFS	System Functional Specification
TDP	Technical Data Package
TPS	Toyota Production System
UI	User's Interface
VSM	Value Stream Map
WiP	Work in Progress
WP	Work Package
XP	Extreme Programming

1 Introduction

This Thesis concentrates on adjusting the existing product development model in the case company to meet the changing needs of the product development process in order to comply with the modern Agile methods to meet the requirement of flexibility.

Product development is one of the basic business processes that can be defined in several ways. Therefore product development has to be defined. Before defining product development, a definition of a product is needed. There are various views on what distinguishes a product and how it differs from a service, and what is the relation of a service and a product. In this study, a broad definition by Ulrich and Eppinger is used which defines a product as something sold by an organization to its customers (Ulrich and Eppinger 1995: 2). As for product development we will use the following definition:

> *Product development is the set of activities beginning with the perception of a market opportunity and ending in the production, sale and delivery of a product.* (Ulrich and Eppinger 1995: 2)

Waterfall is a term used to cover all the development models that are based on a stepwise product development approach. The steps are usually referred to as phases, for example Planning phase, Design phase etc. The different phases are separated by gates. The gates are used to grant access to the next phase, usually the gates include a set of goals or criteria to pass the gate. The main thought behind Waterfall is that a product development project proceeds through the phases and gates in one direction from the beginning to the end. As water flows down a waterfall. The criticism towards waterfall states that there is no room for flexibility and learning during the project.

Agile on the other hand is a term used to cover all the product development models that are based on iterative development. The focus in Agile is on learning during the project and adapting to what is learnt during the project. For example, implementing Feature 1, might expose something that completely changes the need for Feature 2, then Feature 2 can be adapted based on the learning from Feature 1. This is made possible by using iterations. Where Waterfall defines design as something done in the beginning of the project, Agile has continuous design in every iteration. Flexibility and Agile are very often linked together.

In this study flexibility is considered in two dimensions. First of all flexibility in terms of shorter product development projects and therefore shorter time to market for new products. Secondly, flexibility in terms of product development project content, meaning adaptability to changing customer needs during projects. With improved flexibility the competitiveness of the case company can be enhanced, with the possibility to faster meet new customer needs and the possibility to develop products that are more inline with customer needs.

The nature of today's business environment puts pressure on management to extend product and service offerings. Managers are aware of the tough competition and need to do all in their power to differentiate their products from other available offerings on the market. In many industries, however, openness in innovation and possibilities brought by new technologies make it nearly impossible to be more cost efficient than the competitor, or have a superior product. Even if the company can manage to achieve some advantage with better quality or by having a cheaper production, it is very likely that the competitors will soon copy the product, and even improve it in a short while. This puts enormous pressure on innovation and product development.

Product development is, thus, rapidly changing now. Models and processes for product development, such as the Stage Gate model where created decades ago. Today, business practice suggests that companies do not compete with inventions any more; they compete with innovating and developing efficiency, product development included. It means that the focus is now shifting from the questions of "How to create inventions?" and "How to be innovative?" to the challenges of "How to apply the invention?" and "How to implement the innovation?"

The current economic situation has also led to smaller research and development (R&D) budgets for many companies, a smaller R&D budget makes it difficult to justify early investment decisions. A reduced R&D budget puts considerable pressure on the cost of product development. This fact, coupled with the difficulty of influencing decision makers to make an early investment decision, is the reason for this study to focus on improving the product development model. The new economic situation calls for the new model that can provide a shorter time to market interval, avoid delays and better match the product and customer needs. Therefore this Thesis investigates the evolution from traditional Waterfall based product development models to Agile product development models.

1.1 Business Problem

The business challenge of this study is to adapt the currently dominant Waterfall based product development model to become more flexible. This means incorporating elements of the Agile development mode, and by avoiding the negative impacts of the total Agile model.

To address these challenges, this Thesis focuses on the product development model needed for the case company. It starts with the investigation of the standard product development model and also a variation of this model that has evolved from the standard model to meet challenges of adaptability and time to market requirements. It then compares the models used in the industry's best practices to common Agile development models widely known in the software industry. Currently, the case company uses a standard Waterfall-based product development model in all its development projects, with minor adaptation based on the size of the project. The current model brings certain visibility to the project management, and that visibility should not be reduced when adapting the process. In addition, the current model sets a number of distinct criteria for quality, and it has tools to follow up on cost and schedule targets. The Thesis, therefore, aims to suggest a new model that would include the best practices from Agile product development while retaining the visibility and quality needs set by the case company.

1.2 Case Company

The case company is a worldwide leader in global security solutions and systems, providing Lead Systems Integration and value-added products and services to civil and military customers around the globe. In addition the company develops products for the public safety industry, with the company projects ranging from airplanes to small software orders.

Being a large industrial company, the case company has years of experience in product development, with established processes following the main frame of the traditional Waterfall model. In the case company, this model has been used for every development project, with minor adaptation based on size of the project. The adapted Waterfall model has been successful and, so far, satisfied the needs of the case company due to a number of reasons, including the fact that it has provided a required level of visibility for the management. In addition, the Waterfall model has managed to set distinct criteria for quality, and suggested effective tools to follow on the cost and schedule targets.

Presently, the company uses one process model for all the projects, which needs to be adjusted to better suit particular project types. Although some guidelines have been created for how to adapt the model for smaller projects, there are still no guidelines available for how to deal with Agile development models, which is a present need for improving the case company product development processes.

The case company has earlier tried Agile development models, but the results have been unsatisfactory. The main challenge was the lack of management visibility and control of the product development. Therefore, this Thesis develops a new model that would addresses the management visibility and quality requirements from the case company, in addition to meeting Agile requirements.

This Thesis uses an opportunity to apply the newly developed model to an ongoing software development project in which the software is being developed by a subcontractor using Agile methodology. The project follows the existing product development processes, and the in-house parts of the project also follow the standard procedure, including, for example, Documentation, Testing and Verification, Service Creation and other standard process stages.

From the beginning of this project, it has been identified that the Agile method would provide a fast way to create the required product. The challenge in the project is, however, that the Agile method, which addresses very well the way to work within the project, does not bring the same level of visibility for the management to follow the progress of the project. Therefore, the problem of improving the visibility for the management is articulated as one of the foci of this Thesis.

1.3 Research Objective and Research Question

The objective is to bring more flexibility to the current product development model, in terms of shorter time to market and more adaptability to changes during the product development cycle. In addition the objective is to maintain or improve the current level of quality and management visibility.

The research question for the Thesis is formulated as this:

How to meet the targeted levels of quality and management visibility while utilizing Agile development practices in an improved product development model?

This research question results will be applied to develop a model for a pilot project. The pilot project is a software development project. The first two releases of the product have applied standard processes practiced in the case company, and the goal of this Thesis is to propose how to be more flexible and efficient in the next release of the project while meeting the specific company requirements.

The plan to meet the Research objective is: a) to analyze the shortcomings of the currently used product development model (Waterfall-based), especially in small to mid-sized project focused on software development; b) identify areas where improvement is needed; c) analyze the possibility of adapting the current model to meet the Agile development principles; and d) to propose a model that would benefit from the flexibility of Agile development while keeping the Quality metrics of the currently used model, such as quality requirements and management visibility. As the company already has strong processes and guidelines in place for product development projects the current process has to be taken account.

2 Method and Material

This section overviews the research method used in this thesis. This section will also present the research design of the Thesis and the data and material collection process. In addition the criteria for reliability and validity for the thesis are presented in the end of this section.

2.1 Action Research

The research method selected for this Thesis is Action Research (AR). The principles of action research are based on problem solving in cycles where the problem is solved together with the peoplee involved in it. Another definition of action research is research *in* action, compared to research *about* action, aiming for the problem to be solved. (Coghlan and Brannick 2010)

The term Action research was first introduced by Kurt Lewin, an American educator and social psychologist. Kurt Lewin used the term action research for work that did not separate the investigation from the actual action taken to solve the problem. Lewin introduced a cyclical process which involved a non-linear pattern of *planning, acting, observing* and *reflecting* on the changed incurred in the social situations. (Ferrance 2000: 7) Another educator, Stephen Corey at Teachers College at Columbia University was one of the first to use action research in the pedagogical field. He believed that the value of action research lies in the changes that occurs everyday. He argued that by studying the consequences resulting from taken actions would more likely change and improve existing practices than reading about what others have discovered. (Ferrance 2000: 7-8)

The cycles of action research are divided into a pre-set and four main steps. The action research cycle starts with the pre-step, *context and purpose*, investigating the reasons for the research unfolded, at which the context of the research problem and key stakeholders are identified. After the pre-step, the actual action research cycle starts, with the four main stages rotating, namely: *diagnosing, planning action, taking action* and *evaluating the action taken*. (Coghlan and Brannick 2010: 22)

Figure 1 illustrates the stages of the action research cycle.

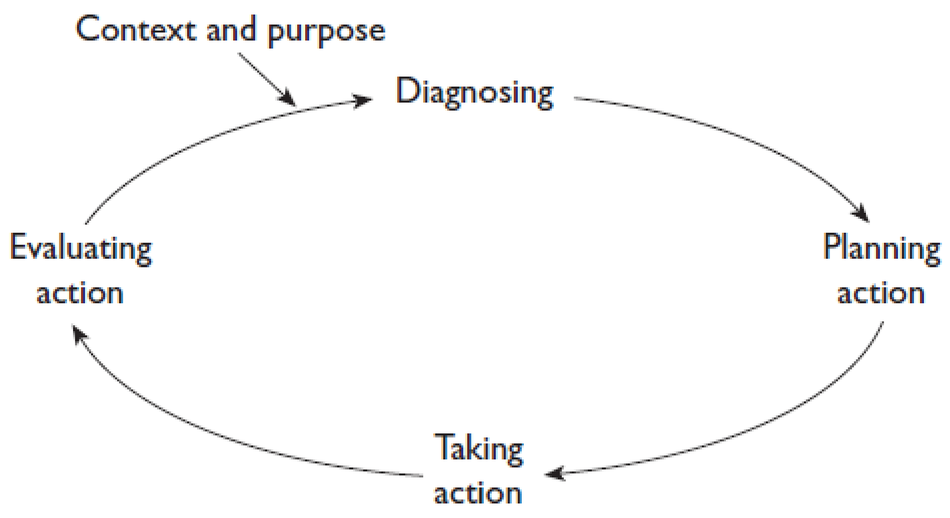


Figure 1. The action research cycle (Coghlan and Brannick 2010: 22).

As seen from Figure 1, *the pre-step* sets the scene, by defining the context and the issues to be solved. When the context is defined and the impacting internal and external forces identified, what is left for the pre-step is to define the desired future state. (Coghlan and Brannick 2010: 21-22)

After the pre-step, the action research process enters the Action research spiral which consists of two or more consecutive research cycles.

The Action research spiral is illustrated in Figure 2.

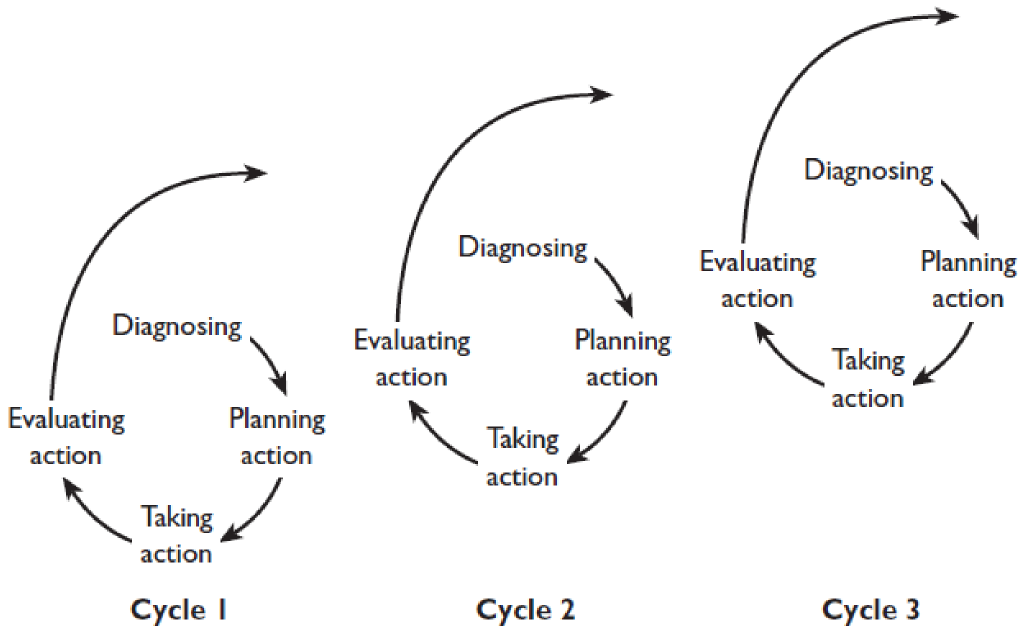


Figure 2. The action research spiral (Coghlan and Brannick 2010: 24).

As seen in Figure 2, the first step of the main steps is *the diagnosing*. The purpose of the diagnosing is to name the issues at hand and to define the actions that need to be done. As the diagnosing will most probably change in later iterations of the research cycle, it is very important to document thoroughly the initial diagnosing as well as the changes that are done in the later iterations. (Coghlan and Brannick 2010: 22-24)

After the diagnosing comes *the planning action* step. The planning of actions is done based on the outcome of the pre-step and the diagnosing, and is consistent with them. After that, the plan is executed and the planned actions are performed in *the taking action* step, which is followed by *the evaluating action* step. The performed action is evaluated to assess whether the original diagnosis and the taken actions were correct, and what should be feed into the next cycle. (Coghlan and Brannick 2010: 22-24)

Although there are many different variants of definitions and implementations of action research, the fundamentals of actions research stay the same and include these main steps and iterations of them. In this Thesis, the three first steps – setting the research question, conducting the current state analysis and analysing literature review – correspond to the pre-step of the action research cycle. The data collection and analysis phase followed by taking and evaluating actions, represent the main steps in the action research cycle. Considering the business problem of this Thesis, action research was selected as it best meets the needs of the Thesis.

2.2 Research Design

The research design implemented in this Thesis is illustrated in Figure 3.

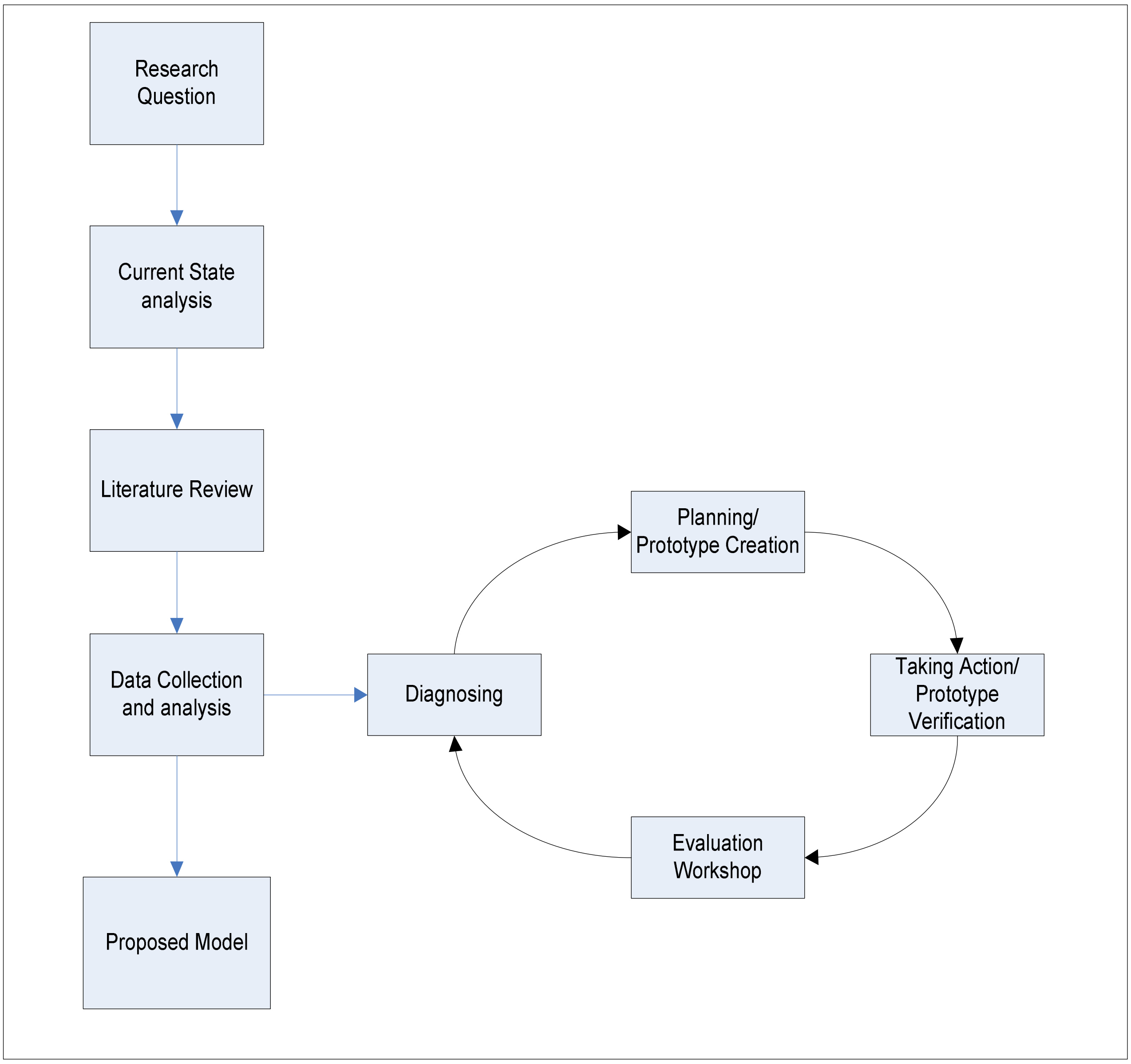


Figure 3. Research design of this study.

As seen in Figure 3, the research starts with setting the research question and conducting the current state analysis, followed by the literature review and data collection and analysis. The purpose of the current state analysis is to establish the reasons for

the research being conducted and to analyse the needs for change. The current state analysis is composed of interviews, discussions and several workshops. The participants to the current state analysis events where selected from different departments and roles, to get as broad perspective as possible. The participants were selected from Product Business Management, Project Management Office, System engineering, System Architects, Quality Management, Customer documentation and Verification. By including all of the important stakeholders the new model is easier to implement and the acceptance of the new model will be easier.
Secondly, the results of the current state analysis are compared to the industry's best practices derived from the literature review. Finally, the results of the current state analysis and the literature review are synthesized to create prototypes for the data collection and analysis part.

As for the first part, the literature review, it investigates various frameworks selected after generic analysis of Agile development principles. To be able to make a comparison between the Agile models and the traditional Waterfall models, theoretical research is made on Waterfall models. One traditional model based on Stage Gates, closest to the model used in the case company, is analysed in detail to make comparison to the Agile development models.

The literature review is illustrated in Figure 4.

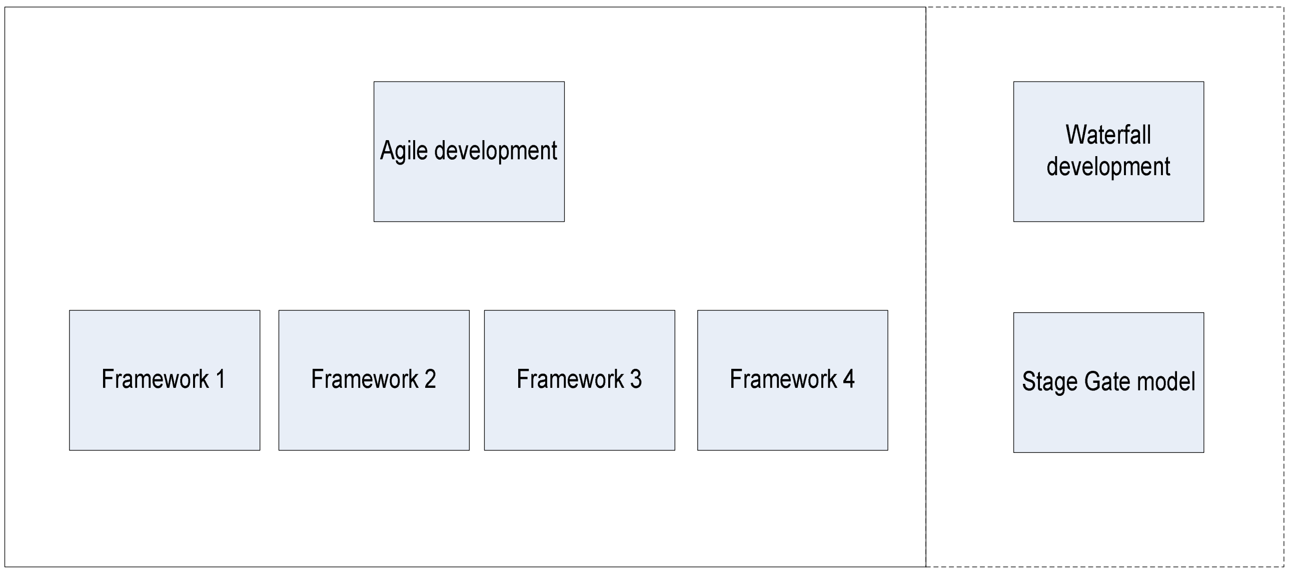


Figure 4. Literature review in this study.

As seen from Figure 4, the literature review selects and analyses the development frameworks using 5-10 theoretical sources per framework to understand, make comparison and create prototypes based on industry best practices.

2.3 Research Process and Implementation

The research process in this study is based on the actual action research cycles. It takes the current state analysis and literature review as input to the diagnosis step, which is performed with the members of the project team. The data collection and analysis lead to the prototype creation and verification. The project team creates the prototypes based on output from the diagnosis step. The prototypes are then verified in workshops with the key stakeholders from across different functions of the product development process. The key stakeholders consist of persons directly working with the development process and of persons accountable for business results based on new products and project efficiency. After the verification stage, the results are evaluated in evaluation workshops. Three prototypes are created based on theoretical search and workshops with subject matter experts. Each prototype is then separately verified with the experts and refined in further workshops. The final prototype is verified by the product decision board and, based on the decision board feedback, the proposal for pilot project is created.

Figure 5 presents the whole research design process.

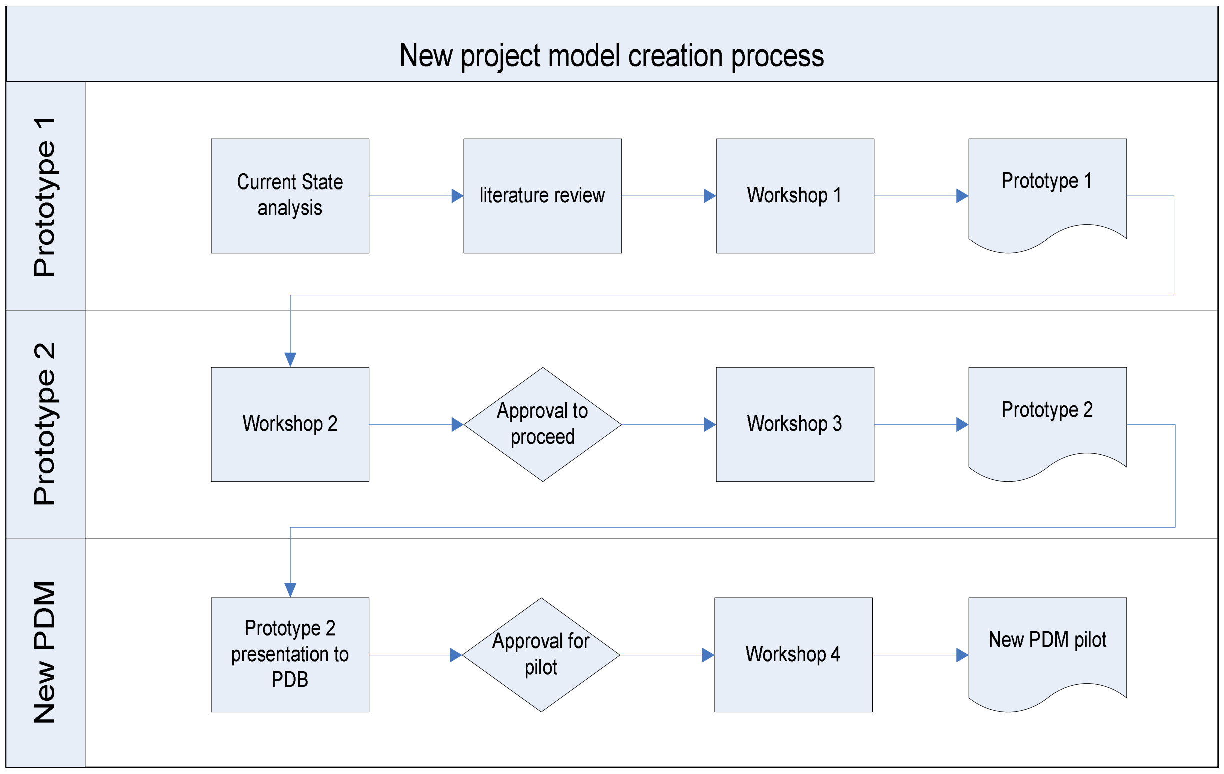


Figure 5. Practical flow of the research process in this study.

As seen from Figure 5, the research process includes the pre-steps (with current state analysis, literature review, decision points) which end up in creating Prototype 1; and the actual steps of the action research spiral, ending with the proposal for the new product development model.

2.4 Data Collection and Analysis

The process of data collection starts with the current state analysis including a series of case company interviews and discussions, as well as a *lessons learnt session* conducted by the Quality Manager of the currently ongoing development project.

The details of the data collection for the current state analysis are specified in Table 1.

Data from (event)	*Participants*	*Data, duration*	*Topics, questions*	*Documents*
Interviews, discussions	The case company experts (4 persons): • System Architect • Project Manager • Quality Manager • Head of Operational Excellence	3 x 1 hour sessions	Appendix 1	Field notes, minutes and PowerPoint presentations
Lessons Learnt Session	The project team members (8 persons): • Quality Manager • Project Manager • Product Business Manager • System Architect • Verification Manager • Documentation Manager • Project Manager, for subcontractor company • Development Manager	1 x 1 hour Data collection session 1 x 1 hour presentation of the results collected	Appendix 2	Field Notes, Minutes and PowerPoint presentations

Table 1. Details of the data collection for the current state analysis.

As seen from Table 1, the participants at the lessons learnt consisted of the project team. The outcome of the current state analysis, from both the interviews and discussions and the lessons learnt material, was used together with findings from the literature review as input to *Workshop 1*.

At Workshop 1, *Prototype 1* was developed. The participants of Workshop 1 consisted of a limited number of the project team members.

The details of Workshop 1 are shown in Table 2.

Data from (event)	*Participants*	*Data, duration*	*Topics, questions*	*Documents*
Workshop 1	Selected project team members (4 persons): • Quality Manager • Project Manager • Product Business Manager • System Architect	1 x 4 hour workshop	Prototype 1 creation. (see Appendix 1)	Field notes, minutes, PowerPoint presentations and illustration figures of the model

Table 2. Details of the data collection in Workshop 1.

Prototype 1, which was created in Workshop 1, was presented at *Workshop 2* for the company's key stakeholders for product development models.

Table 3 shows the details of events in Workshop 2.

Data from (event)	*Participants*	*Data, duration*	*Topics, questions*	*Documents*
Workshop 2 (= Decision Point 1)	Selected members (8 persons): • Selected project team members (from Workshop 1) • Head of Operational Excellence • Head of Project Management office • Head of R&D, software development • Senior Project Manager	1 hours presentation of Prototype 1 1 hour discussion 1 hours data collection for Prototype 2	Prototype 1 verification (see Appendix 1)	Field notes, minutes and PowerPoint presentations

Table 3. Details of the data collection in Workshop 2.

The end of Workshop 2 also became *Decision Point 1.* At Decision Point 1, the Head of Operational Excellence gave his approval for continuing to Workshop 3 and his support for starting the preparation for the new project development model to be used at the pilot project. His approval was needed for securing resources for development of the new project development model. At Workshop 2, the list of invited participants to Workshop 3 was also agreed.

Workshop 3, in addition to the main stakeholders within the organisation that were affected by the project development model, also included external mentors from subcontracting company.

The details of Workshop 3 are presented below in Table 4.

Data from (event)	*Participants*	*Data, duration*	*Topics, questions*	*Documents*
Workshop 3, Kaizen workshop (= Decision Point 2)	Selected members (19 persons): • Selected project team members (from Workshops 1 and 2) • Head of Operational Excellence • Head of Project Management office • Head of R&D, software development • Senior Project Manager • Manager of Verification team • R&D Project Manager • Manager of test laboratory • System Architect 2, not part of project team • PBM representative 2, not part of project team • Development Manager • Senior Quality Manager • Participants from Subcontractor • Lean Mentors from Subcontractor	2 x 8 hours 2 hour of the preparatory event (mentors, subcontractor participant and selected project team members) 1 hour of the retrospective (mentors, subcontractor participant and selected project team members)	Release 1 Project issues, General projects, Values stream map, root cause analysis and Solution brainstorming (see Appendix 3)	Field notes, minutes, PowerPoint presentations, photographs and VSM

Table 4. Details of the data collection in Workshop 3.

Workshop 3 was performed as a Kaizen workshop facilitated by experienced Kaizen mentors. The purpose of Kaizen workshops was to achieve a state of continuous improvement and identify small steps for improvements. This workshop was focused on synchronizing people, finding a common goal, uncovering problems behind problems (root causes), identifying long term solutions as well as the next small steps. The word Kaizen means *"Continuous improvement"*, and the Kaizen philosophy is based on small changes for the better. Instead of looking for a dramatic big innovation the purpose of Kaizen is to identify the small steps that can be taken immediately. The results of following the Kaizen philosophy can lead to significant changes and big improvements, the difference being that the improvements are done in small steps and they never end. (Masaaki 2012: section 1)

As the outcome from Workshop 3, *Prototype 2* was created. The Product Business Manager presented Prototype 2 as part of C-1 milestone presentation to the Product Decision Board. C-1 milestone is the first milestone for any project. At C-1 milestone, the resources are secured until the next milestone. C-1 milestone also serves as *Decision Point 2* for the development process to create a new project development model.

The outcome from the *Decision point 2* was used as input for the *Workshop 4*. At *Workshop 4,* Prototype 2 was finalized with the recommendations given by the PDB. The outcome of *Workshop 4* was the proposed new model to be used in a pilot project.

The details of Workshop 4 are presented below in Table 5.

Data from (event)	*Participants*	*Data, duration*	*Topics, questions*	*Documents*
Workshop 4	Selected project team members (4 persons): • Quality Manager • Project Manager • Product Business Manager • System Architect	4 hours	Prototype 2 finalized. Creation of the proposed model (see Appendix 1)	Minutes, PowerPoint presentations, Visio chart, product backlog Excel and quality criteria Excel

Table 5. Details of the data collection in Workshop 4.

Between Workshops 1-3, qualitative interviews were conducted with System Architect, Project Manager, Quality Manager and Head of Operational Excellence in the case company. All the workshops and interviews were documented; with interviews questions and answers collected, and the workshops minutes and PowerPoint presentations created during sessions kept in the project team and researcher's archive.

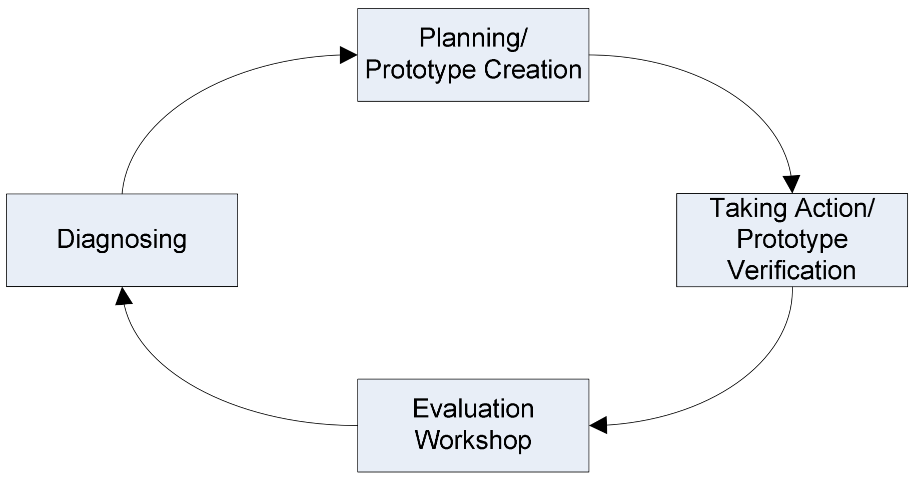

Planning/ Prototype Creation
Taking Action/ Prototype Verification
Evaluation Workshop
Diagnosing

Figure 6. Iterative process of data collection and analysis in this study.

Overall, the process of data collection and analysis for this study can be represented by an Action research cycle and its main stages, as illustrated in Figure 6.

2.5 Validity and Reliability

To be able to correctly measure the results of research, it should be evaluated from the point of view of *validity* and *reliability*. Validity measures the appropriateness, the meaningfulness and the usefulness of the research. Whereas the reliability reflects how free from errors and how consistent and repeatable the research is. (Dooley 1995: 77-78)

One of the measures for improving validity of the research is addressing the question of "Was what was found as a response to the questions originally asked?". This measure of validity is sometimes called *internal validity* or *face validity*. In business research and qualitative research such as this Thesis, the internal validity is usually not a matter of concern, as during the research a lot of data is collected about the subject of the study.

The matter of *internal validity* should reflect if the actual research question(s) was answered. *External validity* as another aspect of validity answers whether the results of the results would applicable in other contexts or situations. In qualitative research with small samples of data, the applicability to other contexts can be difficult. In this Thesis, for example, the appropriate measure of external validity will be the question of "How transferrable the results are to other projects in the case company?", as the research focuses on only one project. (Quinton and Smallbone 2006: 126-129)

Reliability can be explained with addressing the question of "Would we get the same results if we would the research were done again?". This means that reliability deals with consistency of the research. The reliability of the research can be strengthened by applying the following methods: using different data sources, using different data collection tools, applying well-established theory from one area to another, collecting data at different time points, involving different researchers at different points of the research. (Quinton and Smallbone 2006:129-131)

In this study we plan to increase validity by first of all focusing on the fact that the Research Question is answered. In addition the validity is planned to be increased by involving external experts to the model creation phase and by validating if the new model is also applicable to other projects in the case company. The Reliability is planned to be increased by studying a wide range of data from different sources in the Section Best practice for product development. In addition, it is planned to include a broad range of experts from many different fields and from other projects to the model creation phase. The experts will rotate their participation to the different workshops, while some key participants will participate to all workshops and interviews

3 Current State Analysis

This section describes and analyzes the product development model currently used in the case company, stressing the importance of quality, cost and schedule criteria of the existing model. In addition, the visibility of the project progress is described, as well as the main challenges and shortcomings of the currently used model.

This section also shows how CMMI® (Capability Maturity Model Integration) is used to guide the current processes and projects in the case company and how the CMMI level is audited to set the criteria for product development process. CMMI defines the quality requirements used at the case company.

The most important quality metrics of the case company studied in this thesis are quality requirements and management visibility. Where the main part of management visibility is provided by the C1 process, the main part of quality requirements are derived from the CMMI.
Finally, this section also discusses how the standard model is used in the current case project.

3.1 Standard Product Development Model

Presently, the case company uses a waterfall based product development process with clear and distinct guidelines. This process includes several sub-processes; for example, for product development projects in the case company a process model called *C1: Product Management* process is used. The general process policy document (Vancayzeele 2010) defines the scope and objective of the C1 process as follows.

> The objective of Solution and Product Management Process is to provide a clear framework for the solution and product portfolio management including, requirement development, road mapping and business performance, the governance, and all aspects to develop, maintain and remove products or solutions on time, within budget, at the level of quality and with the level of profitability as in defined business case compliant with Line of Business requirements. (Vancayzeele 2010: 8)

The C1 process is a model for following up and managing the whole product lifecycle.

The C1 process is illustrated in Figure 7.

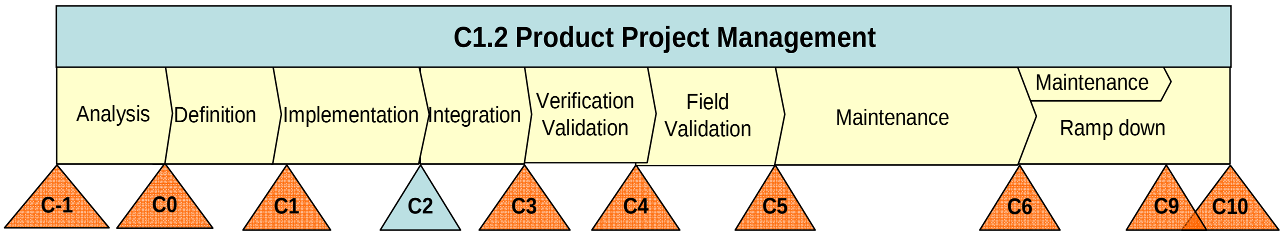


Figure 7. Simplified picture of the C1 process (Case Company BMS 2012).

As seen from Figure 7, the C1 process starts with *Milestone C-1* and ends at *Milestone C-10.* The Gates from C-1 to C5 are used to follow up on the development project, while Gates C6 to C10 are utilized for phasing out products from the portfolio. (Case company Business Process Management Suite 2012)

In the C1 process, for every milestone there are well-defined criteria set for passing the milestone and get the approval to continue to the next phase of the project. At every milestone, the project specific validity for each gate criteria is evaluated. Although this process does allow a certain level of tailoring in the beginning of the project, the case company basically uses the same model for every project. (Case company Business Process Management Suite 2012)

The process policy deployment is ensured by *Product Decision Boards.* The Product Decision Board reviews the gate criteria and makes sure that the C1 process is implemented in a consistent way. (Vancayzeele 2010)

Since this Thesis is limited to the analysis of the product development part in Gates C-1 to C5, only this part of the model will be described.

The C1-process gates and their descriptions are listed in Table 6.

Gate	*Description*
C-1	Content proposal ready
C0	Project commitment
C1	Commitment confirmation and planning ready
C2	Implementation ready and ready for integration
C3	Ready for verification and permission to tender
C4	Ready for customer deliveries and verification ready
C5	Ready for volume deliveries and Field validation ready

Table 6. The C1 process gates used during development projects.

The lag between Gates C-1 to C5 corresponds to the actual phases of a development project. Figure 8 illustrates these gates and the phases between them.

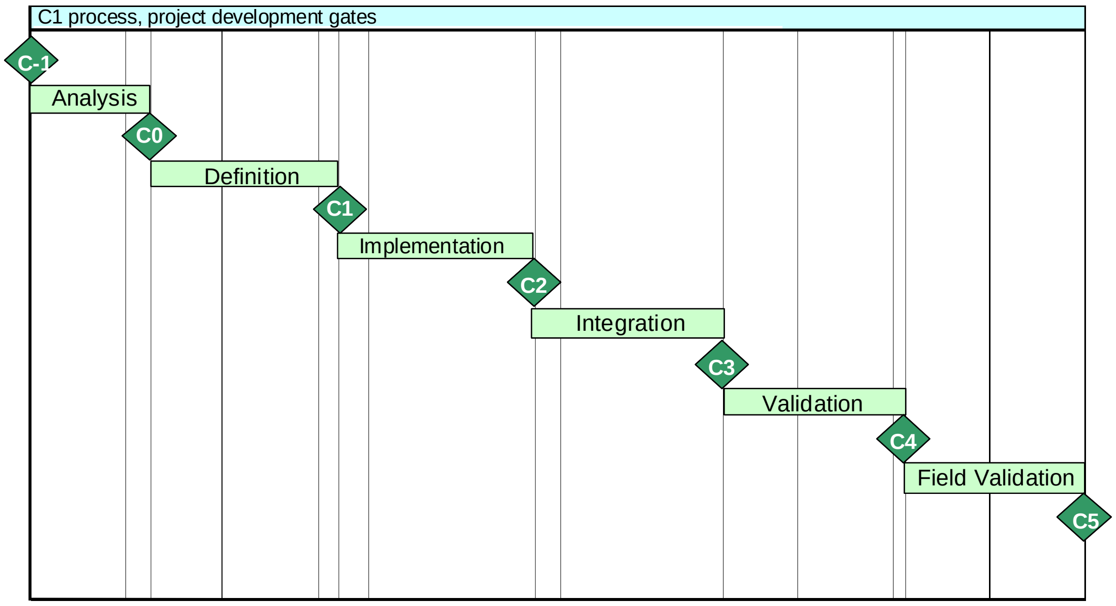


Figure 8. C1 process gates and phases.

As seen from Figure 8, the lag between Gates C-1 to C5 includes the following phases: a) Analysis, b) Definition, c) Implementation, d) Integration, e) Validation, and f) Field Validation. The two phases left out of the scope of this Thesis are Maintenance (after Gate C-5) and Ramp Down (after Gate C-6), which go beyond the product development process. The C1 process is a classic version of waterfall, where product development lifecycle starts with an analysis or discovery phase and continues stepwise through several gates and phases to end up in a product release. Figure illustrates how the waterfall based models works, starting from top left and then sequentially progressing towards the bottom right.

3.2 Capability Maturity Model Integration for Development (CMMI-DEV)

This sub-section gives an overview of CMMI and some understanding of the levels of CMMI. It also shows how CMMI and Agile development fit together. CMMI as such lies outside the scope of this Thesis, but as it sets requirements for the product development process, it has to be considered, and the proposed model has to meet the criteria set by the targeted CMMI level.

Capability Maturity Models (CMM) are used to give a simplified view of the world, the goal of CMMs being to provide the essential tools to describe an evolutionary improvement path from immature ad hoc processes to disciplined mature processes with improved quality and effectiveness (McMahon 2010) To follow up and ensure proper processes, the case company uses CMMI-DEV to manage and measure that the processes are properly implemented and used. CMMI-DEV consists of the best practices and development activities for developing products and services; and it addresses the whole product life-cycle from development to maintenance. The purpose of the CMMI-DEV is to help organisations improve their development and maintenance processes for both products and services. (Chrissis et al. 2011: 3-9)

CMMI for development consists of practices for project management, process management, systems engineering, hardware engineering, software engineering and other supporting processes used in product development and maintenance. The CMMI-DEV consists of exactly 22 process areas, with 16 being core processes, 1 shared and 5 development specific processes. The five development specific processes are addressing the requirements development, technical solution, product integration, verification and validation. Chrissis defines the process area as:

> A cluster of related practices in an area that, when implemented collectively satisfies a set of goals considered important for making improvement in that area (Chrissis et al. 2011: 20).

The process areas are made of different components, these components divided into Required, Expected and Informative. The CMMI-DEV does not suggest using every component of every process area; instead it provides the means to tailor the processes for the project and use the applicable components. It is the task of every organisation to map its processes to the process areas in the model. (Chrissis et al. 2011: 19-31)

The CMMI-DEV uses several levels to describe the evolutionary path recommended for the organisation. Levels are also used for rating and appraisal of an organisation or a smaller group inside an organisation. In CMMI, there are two ways to represent levels, *continuous* and *staged.* The continuous level makes it possible to achieve *capability* levels and follows the achievement of *maturity* levels. (Chrissis et al. 2011: 31-33) Both ways of representing the levels provide means to improve processes, and both provide the same content and use for the same components.

In its practice, the case company uses the representation of maturity levels. In Table 7, the maturity levels of CMMI are illustrated.

Maturity Level		*Description*
5. Optimizing	Continuous Process Improvement	Organizational Innovation & Development and Causal analysis & Resolution
4. Quantitatively Managed	Quantitative Management	Organizational Process Performance and Quantitative Project Management
3. Defined	Process Standardization	Requirement Development, Technical Solution, Product Integration, Verification, Validation, Organizational Process Focus, Organizational Process Definition, Organizational Training, Integrated Product Management, Risk Management, Integrated teaming, Integrate supplier Management, Decision Analysis & Resolution and Organizational environment for integration
2. Managed	Basic Project Management	Requirements management, Project Planning, Project Monitoring & Control, Supplier Agreement Management, Measurement & Analysis, Product & Process Quality Assurance and Configuration Management
1. Initial	Heroic efforts	Design, Develop, Integrate and Test

Table 7. CMMI staged maturity levels. (Koch 2005)

As seen from Table 7, the CMMI maturity levels range from 1 to 5. Maturity Level 1 is called *initial,* and this level is automatically achieved if an organisation can design, develop, integrate and test. Maturity Level 2 is called *managed*, and it contains 7 process areas, all related to management. According to CMMI, disciplined management is needed before technical processes can have any value. Level 3 is called *defined*. This level includes 14 process areas. According to CMMI, the 14 process areas, together with 7 process areas in Level 2, provide a full set of disciplined processes for the organisation. Level 4 is called *quantitatively managed*. With the process areas in place at levels 2 and 3, the organisation is capable of providing statistical data of its performance. The two process areas in Level 4 use the statistical data to give some understanding of the organisation's performance and the quality of the products the organisation produces. Level 5 is called *optimizing*. Level 5 has 2 process areas used to guide the organisation to continuous improvement by finding and correcting the root causes of problems. (Koch 2005)

Presently, the case company is certified to CMMI Level 2 and is striving for CMMI Level 3 certification in the near future. Thus, the current project is followed up according to CMMI Level 3 metrics and any new process model also has to ensure that the CMMI Level 3 criteria are met.

3.3 Release 1 Project

Currently, a development project is going on (we call it *Release 1 Project*) in which the case company is using a standard development model internally and the subcontractor is using Agile methodology for software development. The subcontractor started by using Scrum but after the beginning has changed to Scrumban. At the case company, Release 1 Project is still reviewed using the standard process (C1 process).

Release 1 Project has a challenging set-up which is important to understand before proceeding with the current state analysis. The actual project is led from Helsinki, Finland, and involves two main subcontractors and one smaller subcontractor. The actual product is developed by a subcontractor in Tampere, Finland, with a smaller part of the product being developed in Vienna, Austria. The User's Interface design is done by a design company in Helsinki, Finland; while the final verification of the product is done by the case company's office in Élan court, France. In addition, the lead customer for the product is situated in Germany where the case company has set up a customer project to manage the roll-out and the field validation of the product. The customer project also acts as the interface towards the customer and has facilitated workshops between the case company and the customer (from Gates C-1 to C4), where the customer has had the opportunity to see the product grow and comment on the look and feel of the product. The project also included heavy User's interface (UI) development and the lead customer wanted to participate to the development of the UI. The customer workshops therefore also included agreeing on UI concepts.

When comparing the execution of Release 1 Project to the standard process gates, it was discovered that the gates and their reviews did not fit together. The Agile development methods used by subcontractor lead to a situation where the criteria for the gates were not reached before the end of the project.

Figure 9 illustrates the unfolding of Release 1 Project.

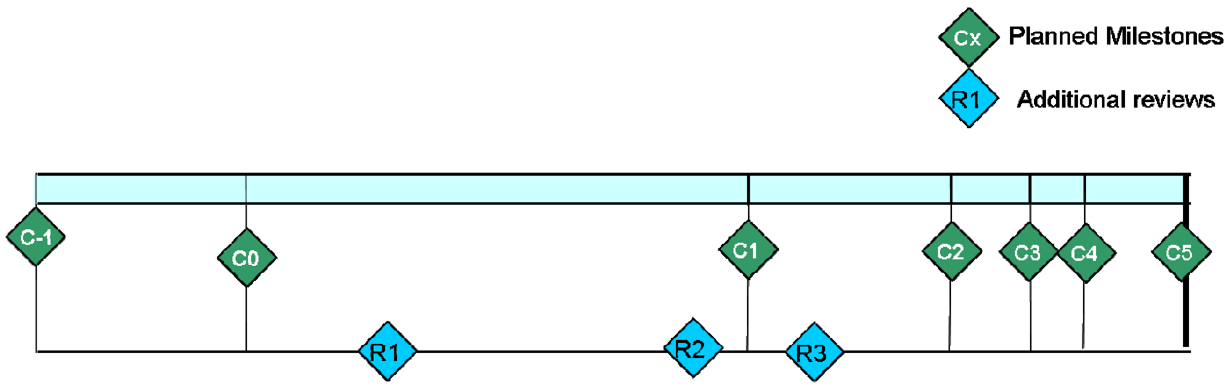


Figure 9. Gates and reviews during Release 1 Project.

As seen in Figure 9, Gate C-1 and C0 were reached in accordance with the case company’s C1 process guidelines, while Gate C1 was reached very late in the project. Gates C2 and C3 were reached just before Gate C4, though the time line between C4 and C5 were in line with the C1 process.

The gates were actually planned as shown in Figure 3, meaning there were no additional delays planned. The challenge in Release 1 Project was that the gate maturity did not match the processes, due to the way the software development was performed. Software development was done using Agile methodology in 3-week iterations, with new features implemented at the very end of the project. The specifications of the last features were done just before the last iteration, just three weeks before reaching Gate C4. In the case company, however, the C1 process expects that the specification should be done already for Gate C1. This mismatch led to a lot of confusion and anxiety among the management. The fact the there was a long gap between Gates C0 and C1, the mismatch of project maturity versus Gate C1 criteria and the late arrival at Gates C2 and C3 lead to a considerable loss of trust from the management side. This loss of trust led to a lot of additional work for the project team to ensure the management of the good progress in the project. Altogether, the management required three additional reviews of the project status (shown in Figure 12). These additional reviews forced the project team to prioritise the reviews and put the project schedule at risk.

Moreover, the close co-operation with the customer, where customer could all the time see the progress of the product and refine their requirements, also led to additional nervousness for the management. The customer's comments led to a sense of new requirements added to the product all the time. Sometimes, the customer's comment actually led to new requirements. Although it was expected to help the final delivery, to get the comments early enough would help much more for the project team to adapt. Even thought the project team was confident that the customer would accept the delivery and be happy with the product, it was difficult to communicate it to the management. The Agile development model used by the subcontractor really helped to involve the customer to the development and led to the customer "speaking for the product" even before it was delivered to them. It also made it possible to meet customer expectations as some requirements were easier to understand while looking at the actual UI together with the customer.

3.3.1 Value Stream Map

To get a picture of how the actual work in Release 1 Project went on, the product development process was analysed using Values Stream Mapping technique (VSM) and root cause analysis.

With the help of VSM, it was possible to identify the top issues and challenges faced during the project. After the VSM for Release 1 Project was created, it was used to show and pinpoint the main issues of the project. The top two issues identified with VSM were investigated using the Root Cause analysis to find the reason for the challenges, which was later applied to Prototype 1 to create Prototype 2, both prototypes presented in Section 5.

In Figure 10, the Values Stream map of the Release 1 Project is illustrated.

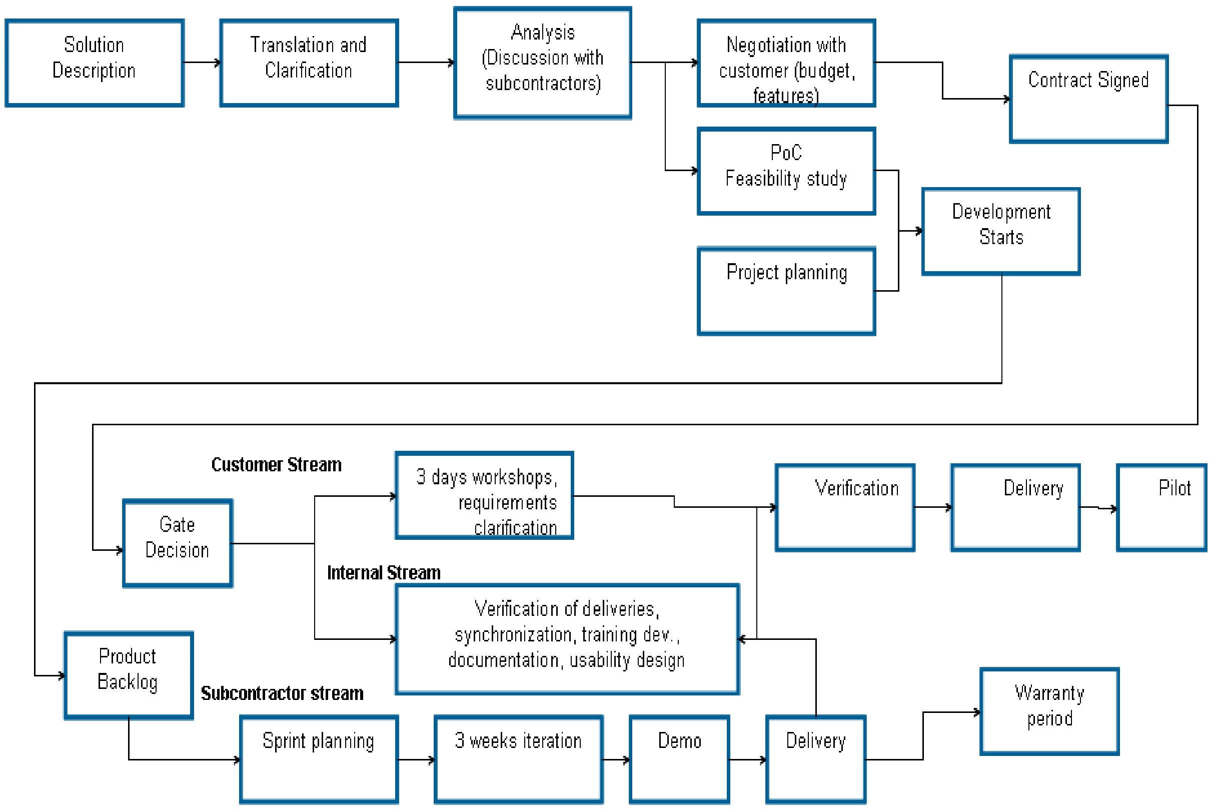


Figure 10. Value stream map for Release 1 Project.

As seen from Figure 10, the project was divided into three different streams, the actual software development stream mainly managed by the project manager from subcontractor. The customer stream was mainly managed by the project manager of the customer project, and the main stream for all internal work managed by the project manager of Release 1 Project. To manage the three different streams, which evolved to have their own schedules, created a lot of additional work especially for the project manager, but also for the rest of the project team.

The Value Stream map was used to pinpoint pain points during the project. The actual process of value stream creation is described in Appendix 3. The main issues found were mapped to the two following issues: 1) *C1-process does not support Agile development model* and 2) *Misunderstood requirements.*

3.3.2 Lessons Learnt

At the end of Release 1 Project, a Lessons Learnt session was held with all the people involved in the project. The lessons learnt session is part of the standard process in the case company, and its results are used to improve the subsequent projects. The discussions conducted on lessons learnt session is given in Appendix 2.

The lessons learn session held after Release 1 Project identified several improvement proposals for Release 2 Project and generally for the company. The most relevant improvement proposals for this Thesis are listed in Table 8 (not in priority order).

Event	*Issues discussed*
Lessons Learn session (after Release 1 Project)	Improvement proposals (towards Release 2 Project) 1. A better tool for live net meetings. 2. A Wiki-page for communication between different sites 3. A new way to report project milestones when Agile software development is used 4. Better communication between different projects about new features in each release. 5. Shorter meetings, teleconferences with customer instead of face to face meetings.
	Other top issues to consider 1. Open requirements in the contract with customer, requirements were discussed for a very long time with the customer 2. Communication with several parties in different locations requires a lot of time and effort 3. C1 milestone process does not suit well to Agile development model, visibility to management about the project's progress and process not good enough

Table 8. Results from Lessons learnt session.

In addition to the improvement proposals listed in Table 8, the lessons learnt session identified other top issues during the project. For example, the C1 process and milestones issues mentioned in both lists indicated that the existing product development process did not support the way of work in Release 1 Project. Also in other parts of the project process (for example, in Customer Documentation and Quality Management) and the mismatch with the C1 process and Release 1 Project was highlighted as an issue.

3.4 Quality Requirements

Quality requirements make up a significant part of the product development process and play an important role in product development projects and the quality follow up of projects for the case company. The management uses the quality requirements to ensure high quality of projects and the outcome of projects. The quality requirements are often used as key performance indicators (KPIs) in projects. In addition the quality requirements together with other targets are often used as basis for personal incentives in the case company.

The guidelines for quality requirements come from the case company process guidelines. The process guidelines are formulated according to CMMI-DEV and define very precisely what deliverables are expected at each gate, the validity of each deliverable decided in the beginning of the project.

The project specific requirements, metrics for them and Key Performance Indicators (KPI) are also defined in the beginning of the project. The usual KPIs utilized by the case company are, for example, the number of test cases performed, the number of test cases passed, reported faults, the number of Critical Faults, Major Faults and Minor Faults, and other indicators. The KPIs are used to set gate criteria for the later gates (C2-C5). An example of a gate criterion for quality requirements is 90% of test cases passed and maximum 0 Critical Faults and 10 Major Faults.

Release 1 Project met the quality requirements according to the initial expectations. One of the main findings was a relatively low number of bugs (bugs versus performed test cases) reported during the verification phase compared to other development projects. The reason for the low number of bugs was partly explained by the Agile development model used by the subcontractor, with a test automation running every night, finding all the most obvious bugs and partly by the high quality of the developed software (presumably also the result of the Agile development model used). In addition, the fact that the specification work was done late, just before implementation, allowed the testing team to take part in the specifications and also the sprint demos to better understand the features to be tested. This fact probably also helped reduce the number of bugs reported earlier due to misunderstanding of the feature.

As a result, all the mandatory quality audits were passed before each of the C1 process gates. But as the audits are tied to the C1 process gates, they did not give the needed information about the project quality. As presented in Figure 9, we see that all the gates after Gate C0 came very late in the project. One of the reasons for the relatively late gates was due to the fact that the project wanted to meet the quality requirements for each gate.

To summarize, the standard quality requirements in Release 1 Project were met, but they did not serve their purpose to provide information about the project quality during the project life cycle, only giving the required information at the end of the project. This is also one of the key reasons for the case company to either not use Agile development in its integrity and adapt the product development model used to serve the case company specific requirements.

3.5 Project Visibility

As the quality requirements is the tool to monitor the project progress and product quality, the project visibility is the tool to follow that the quality requirements are met and that the project progresses according to plans. Therefore, the required visibility is vital to ensure management that projects and products will meet the customer needs with the right quality.

Currently, the project visibility is achieved by passing the gates in the C1 process. In the case company process, the gates are usually reviewed, first, by the Gate Maturity Review (GMR) board and then, if approved, checked by the Product Decision Board (PDB). In addition, the project manager presents a project status report monthly at the Project Follow-Up (PFU) meetings. The gate reviews are evaluated based on their maturity compared to the C1 process gate criteria.

In the Release 1 Project, the project visibility was enhanced by additional management reviews as presented in Figure 9. The conclusion is that the standard tools for project visibility were not enough to provide the management with the needed information. Even with the facts that product development quality was high throughout, the project and quality requirements were met, and software releases were on time, still the project got more management attention than the projects in general. This high attention of the project was partly due to the importance of the lead customer for the product developed, and partly due to the fact that the subcontractor was using Agile development, and not Waterfall-based development which the case company management was used to.

Summing up, the current state analysis described in Section 3 discovered that the Agile development model indeed enhanced the flexibility of the product development project. It helped to meet customer expectations and created a very close relationship between the project team and the customer, which helped communicate and eliminate misunderstandings in requirements and changes needed during the development. Thus, the results of the Agile product development model were undisputedly positive. For example, product quality was high, with a low rate of reported bugs; the development speed was extreme and met the extremely strict deadlines which were set by the customer; finally, customer satisfaction was high and created positive environment between the customer and case company and also between the case company and subcontractor. In addition, the created product was appreciated by other customers, which led to the fact that new releases of the product were planned.

Eventually, the positive impact of Agile development has initiated the case company to start looking for possibilities to adapt its current models to Agile development principles and also to start using Agile development models for projects which utilize in-house resources for the development. In addition, the challenges met with project quality requirements and project visibility have revealed the need to investigate how the C1 process could be adapted to better support Agile development models. The C1 process is proven to provide the required quality and visibility and it is adopted by all the organisations within the case company, therefore, it was decided that the use the C1 process should remain mandatory, and if Agile development is used the only option is to adapt it to the C1 process.

4 Best Practice for Product Development

This section discusses the main features of the product development processes and compares the traditional and Agile models used for product development, based on the review of the research literature and existing best practices.

4.1 Overview of Product Development

Agile product development models have been used for many years and are widely implemented in many software development companies. Lately, the Agile models together with Lean development have gained their footprint also in other industries and product development companies. For several years, software companies have been using Agile development models. This practice has proved that Agile models and iterative project development models provides more flexibility and effectiveness compared to traditional Waterfall-based development models.

Presently, there are many versions of various Waterfall-based product development and Agile development models, with most of them following the same traditional patterns. The main difference between them lies in the formal structure: the Waterfall-based models having phases and checkpoints between different phases, where Agile models have iterations and increments.

Unlike Waterfall methods, Agile development focuses on functionality and flexibility, overcoming the issues that arise in traditional Waterfall-based models, where the project proceeds stepwise from beginning to the end, with little room for changes. For example, most Waterfall-based models start with a design phase, followed by a planning phase, usually added by the specification phase. After the specification phase comes the implementation phase, with the verification phase closing the cycle. This arrangement is a common stepwise approach used in most Waterfall-based models. It is effective in large projects for developing, for examples, airplanes.

However, the driver for the Agile development community is the need for faster and more changeable product development. The stepwise approach from Waterfall-based models is just not suitable. Very early, the software industry learned that *time to market* is very important; therefore, computer industry evolved to develop software products meeting these requirements. Moreover, not only the time to market is important also what *to market* is important. With this, the software industry has to change fast because the customer needs to change fast. Being on a pre-planned track and meeting budget and functionality requirements, business has only two options: either delay the release of the new product, but include the changes needed; or release the product without the needed change. Either way, the company is either late, and does not meet *time to market* requirements; or it releases a wrong product, and thus does not meet *what to market* requirements. Both requirements are addressed by the Agile models, with the traditional model outdated for the current tempo.

In standard waterfall based models, a product development process typically consists of different phases and stage-gate reviews between the different phases. From the point of view of measurements and evaluations, it is also the most common way to control product development in most organizations (Cooper 2002a, Rajesh and Iqbal 2008).

Product development processes in general can be defined as a set of activities that, taken together, produce output to customers where customers can be either internal or external (Benner and Tushman 2003). In other words, product development process is a sum of activities including perception of a market opportunity, production, sale and delivery of an output that can be sold by the organization to the customer, either external or internal. (Benner and Tushman 2003, Ulrich and Eppinger 2004: 2)
Product development is a sum of many factors. Although a central role is played by the product development process or product development model (PDM), not all issues in product development are solved in the product development process. Other issues, for example, the company's general product development strategy, the product development culture of the company, resources available and focus of work (line work vs. product development work) also influence product development.

The most common product development models are based on the Waterfall model, the most common waterfall model is the Stage-Gate model with different phases and stage-gate reviews between the phases, according to the process model used at a particular organisation. (Cooper 2002a, Rajesh and Iqbal 2008). There are many variations of the stage-gate review process, for example, New Product Development (NPD), Simultaneous Engineering (SE), Concurrent Engineering (CE) and Integrated Product Development IPD, to name a few. Usually they consist of 4 to 7 gate reviews. Before, between and after the reviews the process consists of, at least, the following phases: concept development, system-level design, detail design, testing and refinement and production ramp-up (Suomala 2004).

To evaluate product development processes it is necessary first to define what characterizes a successful product development process. According to Ulrich and Eppinger (2004), there are five dimensions that assess the performance of a product development process: a) Product Quality, b) Product Cost, c) Development Time, d) Development Cost, and e) Development Capability. (Ulrich and Eppinger 2004: 2)

The two dimensions - Product Quality and Product Cost - refer to the outcome of the development process. They are important especially for the products expected to have long life-cycle. When reviewing product development processes, these two are easy to forget, with the focus easily going to three dimensions of the development process itself (Development time, Development Cost and Development Capability). But the first two are significant, too. *Product Quality* is ultimately measured by the market share and the price that customers are willing to pay. Product Quality consists of the actual functions that the product has, and should meet the customer needs, and be robust and reliable.

Product Cost consists of the total manufacturing cost of the product developed per unit. Product Quality dictates the price that has to carry the product cost, not to forget the development cost. *Development cost* is a big burden for short lived products. It means the amount of money invested to develop the product. *Development time* is the time from idea to a market launch, also called *Time to market*, which is very important especially when developing completely new products to new markets. Studies show, that the first company on the market has a huge advantage towards competitors (Kulatilaka and Perotti 1997). Finally, *Development capability* is the development team's ability to develop products, based on its experience from other development projects, which should lead to better development skills in the next projects. (Ulrich and Eppinger 2004: 2)

For product development, the importance of one particular factor – the development time (time to market) - is ever growing in the modern markets, and several studies show that first movers have an advantage even in low entry markets (Lieberman 2007). There are two ways to gain time to market advantage towards competitors: earlier investment and faster product development. Early investment generally creates an advantage towards competitors but it also includes a larger amount of uncertainty, since up-front investment is required while customer need and willingness to buy the product is still unknown at the time of investment. (Kulatilaka and Perotti 1997)

The other way to address the time to market challenge is to shorten the development time. Then, even with late investment decision, companies with fast product development can gain an advantage with an early market entry. Moreover, fast development does not just improve competiveness, it also lowers the development cost. Studies have shown that shorter development project equals to lower development cost. (Preston 1999) Therefore, companies try to utilize such product development models which address this issue of a shorter development time. One of such models is the Agile development model.

Even though focus in this work is on Waterfall and Agile models and most of the development models can be classified to either Waterfall or Agile, it is important to get a broad perspective on product development models in general, and not to perform a complete review of Agile development, but instead to find best practices in Agile development, already implemented at many companies, which can help to better combine Agile models with traditional Waterfall models.

To identify the deficiencies of the traditional models against the modern ones, the following sub-section looks into the features of the Agile models in more detail.

4.2 Agile Software Development

Agile development is a broad concept and there are numerous variations of it. There are also many names for Agile development, for example, Iterative development, Incremental development, LEAN development, Concurrent Engineering, Rapid application development, Spiral development etc. They all have slightly different angles but they all address the same challenges of product development. In this study the term "Agile development" is used.

The birth of Agile development can be tracked back to the 1950's. Engineers used a method called iterative and incremental design and development (IIDD). Department of Defense (DoD) in USA was one of the early adopters of IIDD. In the 1950's some of the justifications for IIDD was "avoiding management discouragement" and "increasing customer satisfaction". When looking at the software development which was very often experimental and explorative you can find many of the characteristics that are part of today's Agile software development. IIDD and various variations evolving from IIDD led to the birth of the Agile manifesto in 2001. (Anderson et al. 2008 pages 3-4)

The Agile manifesto was stated by a group of leaders in software development and consultants. It includes the core values of Agile development:

> 1) Individuals and interactions over processes and tools, 2) Working software over comprehensive documentation, 3) Customer collaboration over contract negotiation, 4) Responding to change over following a plan. (Beck et al. 2001)

These four sentences focus on the essentials of Agile development. The first highlight the importance of co-operation and interaction between people. The second brings up that a software development team should continuously produce tested software, with frequent release intervals. The third focus on the interaction between the developers and the customer, where the co-operation between customer and developers is more important than strict requirements. The fourth states that developer and customer should be prepared to make changes to software and also that contracts should be flexible enough to make changes possible. (Abrahamsson et al. 2002: 11-12) These four core values stated in the Agile manifesto can be found in all the different variations of Agile development models.

There are numerous of Agile variations that fulfils the core values of Agile development, some of the most known are Extreme Programming (XP), Scrum, Crystal (family of methodologies), Feature driven development (FDD), the Rational Unified Process (RUP), Dynamic Systems Development Method (DSDM), Adaptive Software development, Open Source Software Development (OSSD) and many more not listed here. They all follow the core values of Agile development. With minor differentiation in the tools and how they are implemented. (Abrahamsson et al. 2002: 18) In addition, Kanban is a widely used model as well as Scrumban, which is a combination of Scrum and Kanban. The Scrum, Scrumban (including Kanban) and XP are reviewed in the sub-sections 4.2.1 to 4.2.3. Crystal, DSDM and RUP are reviewed in the sub-section 4.2.4. The CMMI maturity model and how it can be integrated with Agile development models is reviewed in sub-section 4.2.5.

4.2.1 SCRUM

SCRUM is one of the most widely used and known Agile methodologies. SCRUM is a framework consisting of Scrum teams (with associated roles), artefacts, events and rules. The core elements of the Scrum framework are the scrum events, teams and artefacts. (Schwaber and Sutherland 2011) The phases of Scrum include: *The pre-game phase, The Development phase (Game phase)* and *The post-gate phase. The pre-game phase* includes the creation of a product backlog and a high level architecture design. *The Development phase* is the actual development phase with iterations called sprints. *The post game phase* includes the closure of the project and the final release. (Abrahamsson et al. 2002: 30)

Scrum events are used as the tools to provide transparency, inspection and adaptation for the projects. Transparency, inspection and adaptation are according to the Scrum guide the three pillars of empirical process control. The scrum events to support the three pillars are: *sprint planning meeting*, *daily scrum*, *sprint review* and *sprint retrospective*. A *sprint* is a container for the events; it is a reoccurring iteration round that last from 2 to 6 weeks. Each project consists of many sprints depending on the size of the project and the length of the sprints. The lengths of the sprints might change between projects but one of the core principles of scrum is that the sprint length is consistent within one project. Each of the sprints should end with a "*Done*" releasable product, feature or increment. The definition of "*Done*" is an important part of the Scrum framework and should be defined in the beginning of every project. Although the definition of *Done* is specified at the beginning of the project, it can and will evolve as the Scrum team matures and becomes capable of doing a more detailed and stringent definition of *Done*. (Schwaber and Sutherland 2011)

The *Scrum team* consists of a *product owner*, *Scrum master* and the *development team*. According to the Scrum practice, a Scrum team should be self-organised and cross functional, and it should have all the needed competences to accomplish the project without the need for external competencies. (Schwaber and Sutherland 2011)

The artefacts of Scrum are the *product backlog, the sprint backlog* and *the increment*. *Product backlog* contains all the items that might be needed for the product to be developed. The product backlog is the single source for requirements to the product. It is a dynamic list which is maintained throughout the life cycle of the product. The product backlog is maintained by the Product Owner. Any changes to product should go via the product backlog and approved by the product owner. The sprint backlog, in its turn, is the set of features selected from the product backlog to the sprint. In addition to the content from the product backlog, it should contain the plan for delivering the feature, product or increment. It also includes a forecast from the development about effort needed to accomplish the items. Finally, the increment is the sum of all items created during a sprint and all previous sprints. At the end of each sprint, the increment has to be usable and in "Done" state according to definition for "Done" that was defined in the beginning of the project. (Schwaber and Sutherland 2011)

To summarize the Scrum model, it is a framework of tools consisting of roles, artefacts and events. The roles are used to support and manage the work in the project. The artefacts are the tools used by the project team. Finally the events are the means to communicate and share information inside the project team and to external stakeholders.

4.2.2 Scrumban

Scrumban is a variation of Scrum which takes some of the Scrum principles and some practices from Kanban. Kanban is known to come from the Toyota production system (TPS), which introduced "lean manufacturing". Lean manufacturing has been copied and implemented by other companies all around the world.

Lean manufacturing, however, was just a part of the Toyota production system. Another important aspect was the development of products, the so-called Lean Development. According to the studies, Toyota development projects took half the time of US car manufacturers equivalents. As a result, Toyota grew very fast to become one the biggest car manufacturers in the world, which helped the spreading of the Toyota production system making it one of most important and most copied manufacturing and development models in the world. (Ballé and Ballé 2005)

In the Kanban system, utilized by Toyota, the goal is to visualize the workflow, to limit the Work in Progress(WIP) and to measure the lead time. At the heart of Kanban lies the Kanban board which is a tool used to meet these goals. In the Kanban approach, the release is split into items, and all these items are posted on the Kanban board, so that the team and management can see the status of each item, for example, currently located *In development* or *In testing* phase. (Kniberg 2009: 4)

Figure 11 contains an example of a Kanban board.

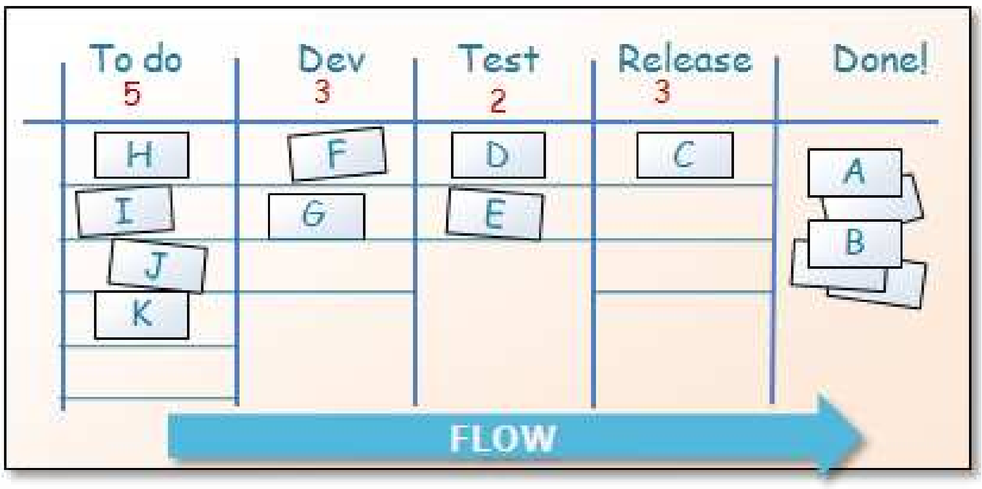


Figure 11. Example of a Kanban board (Kniberg 2009: 5).

As seen in Figure 11, the Kanban board consists of columns representing phases or work packages. Each of the columns has a limit to control the Work in Progress. The work in progress consists of Items/features that move from left to right on the Kanban board. Using a board to visualize the workflow is not something unique for Kanban. In fact it is quite common in the Agile models. The board is usually not the core element of the model. The differentiating factor of Kanban is that it limits the amount of Work In Progress, so the Kanban WiP limit tells the maximum amount of items allowed in that column. This minor difference of using a WiP is a crucial factor comparing for example Scrum and Kanban. Scrum limits the amount of items in an iteration (sprint) whereas Kanban limits based on phase. Kanban is especially useful when there is a risk of items requiring more time than the time box (sprint) allows. In Scrum each sprint ends in a release, where all handled items during that sprint should be in done state. Kanban allows that development of an item runs through the different phases over several iterations as long as the maximum amount of items in a certain column is not exceeded. (Kniberg 2009: 13-14)

Both Scrum and Kanban provide tools and guidelines how to work in the product development project. Depending on the nature of the project one or the other of the models is better. Scrumban is the third option, it combines these two models. It takes parts of both models and the parts are combined, to meet a broader range of project types. As Scrum is one of the most widely used Agile models it is natural that other models evolve from Scrum. In Srumban, the basics of Scrum are followed but as it interferes with one of the fundaments of Scrum it can not be called Scrum. Even though Scrum model is adaptable it has some core elements which should not be removed.

One of the core elements of Scrum is that the length of sprints should not change during a project and each sprint should end with a release with only "Done" items. Some product development projects have faced this element as a limiting factor for an optimal workflow. For example a certain component might be too large to complete within one sprint. It might be impossible to divide it to clear smaller pieces. In addition, the visibility of scrum can be difficult in case of not being able to have small enough items. The workflow in strict Scrum usually just presents the items in the sprint backlog as not started, on-going and done. This leads to items not finished during the sprint, either being not started or on-going. As more items are taken from the product backlog to the sprint backlog, the spring backlog will soon be containing a bunch of items in not started or in on-going state. In long projects, this might look unprofessional and give a poor reputation to the project, with management getting worried about the project schedule, or having other concerns. Taking the freedom of adapting the sprint and by controlling the Work in progress on the Kanban board can help the development team to find a more optimal workflow. In addition, it can provide better visibility of the project for the management. (Ladas 2008) Therefore, these two approaches can successfully work together.

To summarize, Scrumban takes the best practices from Scrum, and uses the same framework of tools as Scrum. In addition, it brings the Kanban board as central artefact to the model. With utilizing the Kanban board, the follow-up on a certain item becomes more detailed. It provides means to see how close to *Done* a certain feature is, whereas Scrum only presents either *Done, On-going* or *Not started* states. Additionally, Scrumban breaks one of the core rules of Scrum; it allows a feature or item to be developed over several sprints. This makes the development of larger entities easier, without the need to artificially break it into smaller items, which is sometimes needed in Scrum. Finally, the Kanban board provides a tool to follow the progress of larger entities, where development time exceeds the time of one sprint. The WiP limit also differentiates Scrumban from Scrum; using WiP limit the work load is controlled by the number of items per phase, instead of the number of items per sprint as in Scrum.

4.2.3 Extreme Programming (XP)

Extreme programming is a collection of best practices from general software development. The idea is not to release something new, but instead collect common sense practices to create a methodology. The name comes from taking these common sense practices to the extreme, thus Extreme programming. (Beck 2000)

The fundamentals of XP can be divided into three parts: the process, the roles and the practices. The process has five stages *Exploration*, *Planning*, *Iterations to Release*, *Productionizing*, *Maintenance* and *Death*. (Abrahamsson et al. 2002)

Figure 12 illustrates the five stages together with the workflows and items in XP.

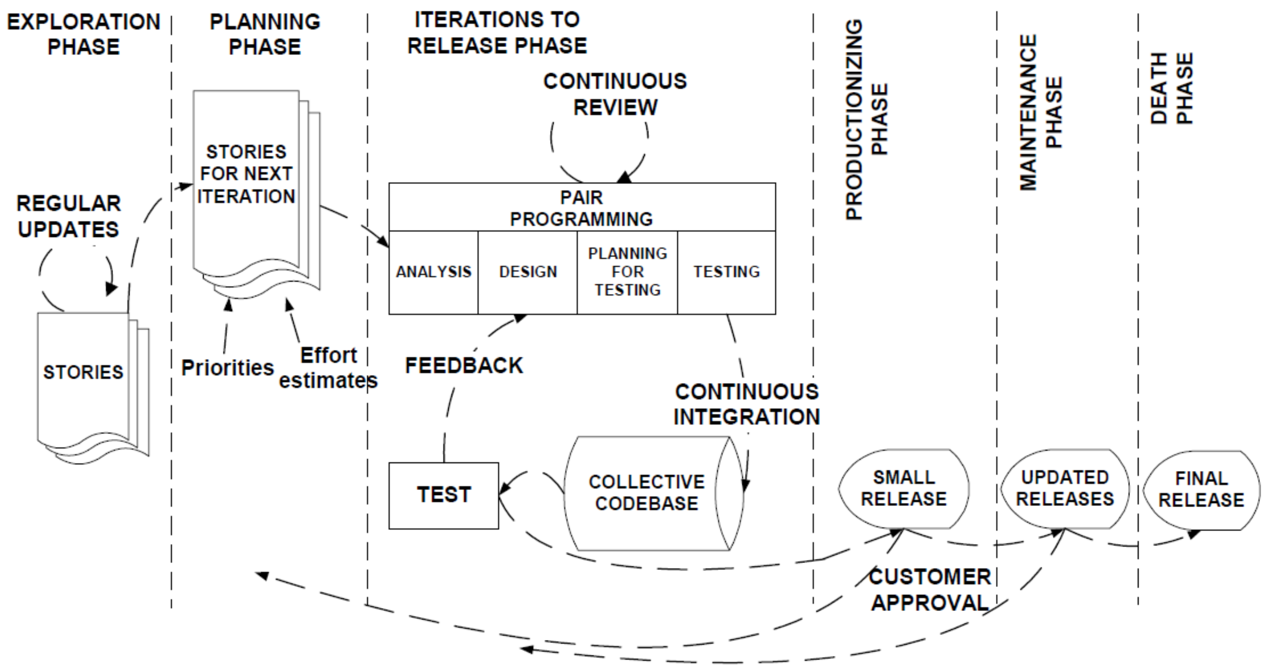


Figure 12. Extreme programming process life cycle (Abrahamsson et al. 2002).

As seen in Figure 12, the Exploration phase contains of Stories (user stories) which are regularly updated. The stories are taken to the planning phase, where Priorities are set and effort estimates are done for the User stories. After the planning phase the User stories are taken to the Iterations to Release phase. Inside the iterations the stories are developed with pair programming. Each story is developed through the cycle Analysis, Design, Planning for Testing and Testing. After the pair programming the code is added to the collective codebase and then the collective code (consist of code from several pairs) goes to the integration Test. The Iterations to Release phase is followed by the Productionizing phase, which consists of a small release and customer approval. After the Productionizing phase the maintenance phase gives an option for an updated release. Finally the project ends with the death phase. The death phase includes the final release and closing of the project. (Abrahamsson et al. 2002)

Similar to the Scrum, the XP approach also defines specific roles in the project. The roles of XP are Programmer, Customer, Tester, Tracker, Coach, Consultant and Manager (Big Boss). To make the XP complete, the process and people are instructed to follow the 14 practices, namely: 1) Planning Game, 2) Small/Short Releases, 3) Metaphor, 4) Simple Design, 5) Testing, 6) Refactoring, 7) Pair programming, 8) Collective software development. The XP was created for Software development and it is not applicable without modification to other industries. The elements of XP are very detailed and easily applied on a practical level. Altogether, the fundaments of XP are comprised of the five stages, the defined roles and the 14 practices.

4.2.4 Crystal, DSDM and RUP

This section look briefly at other commonly known Agile development models, first of all, Crystal, DSDM and RUP, which are well-known and widely implemented Agile development models.

Crystal represents a family of models the Crystal family is differentiated by colour coding, where each colour reflect a different variant of the Crystal model.

In Figure 13, the different colour coding of the crystal family is illustrated.

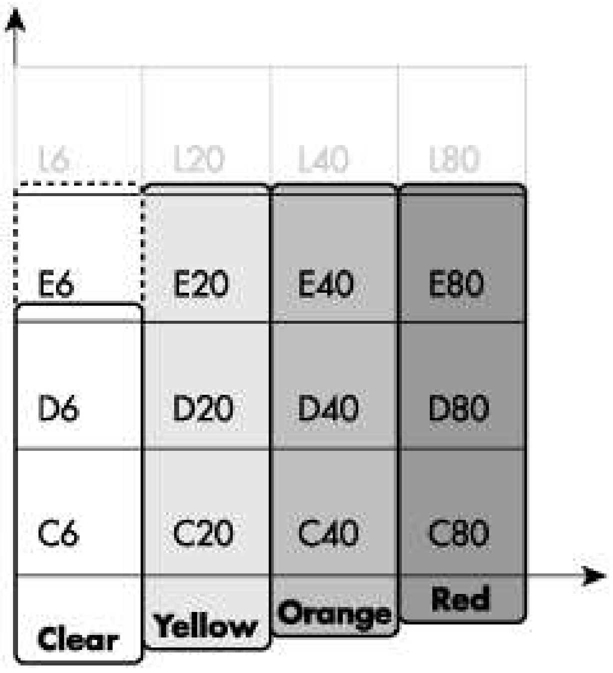


Figure 13. The crystal family names (Cockburn 2006: chapter 6).

As shown in Figure 13, moving right in the matrix means increasing amount of people working for the project (number indicates size of the project, in resources), thus the smaller the project team is, the closer to crystal clear is the model to be used. Moving upward in the matrix reflects increasing the complexity of the project, thus the harder the project is and the higher it is placed on the matrix. (Cockburn 2006: chapter 6) The principle here is that according to the nature of the project, the model to be used is chosen. Different projects should be run differently.

The *Dynamic Systems Development Method* (DSDM) is one of the most known frameworks for Rapid Application Development (RAD). The fundamental idea of DSDM is to lock the resources and schedule, instead of locking the amount of functionality in a product. Then it is possible to adjust functionality of final product, while resources and schedule is not changed. (Abrahamsson et al. 2002)

In Figure 14, the sequential study phases and the iterative phases are illustrated.

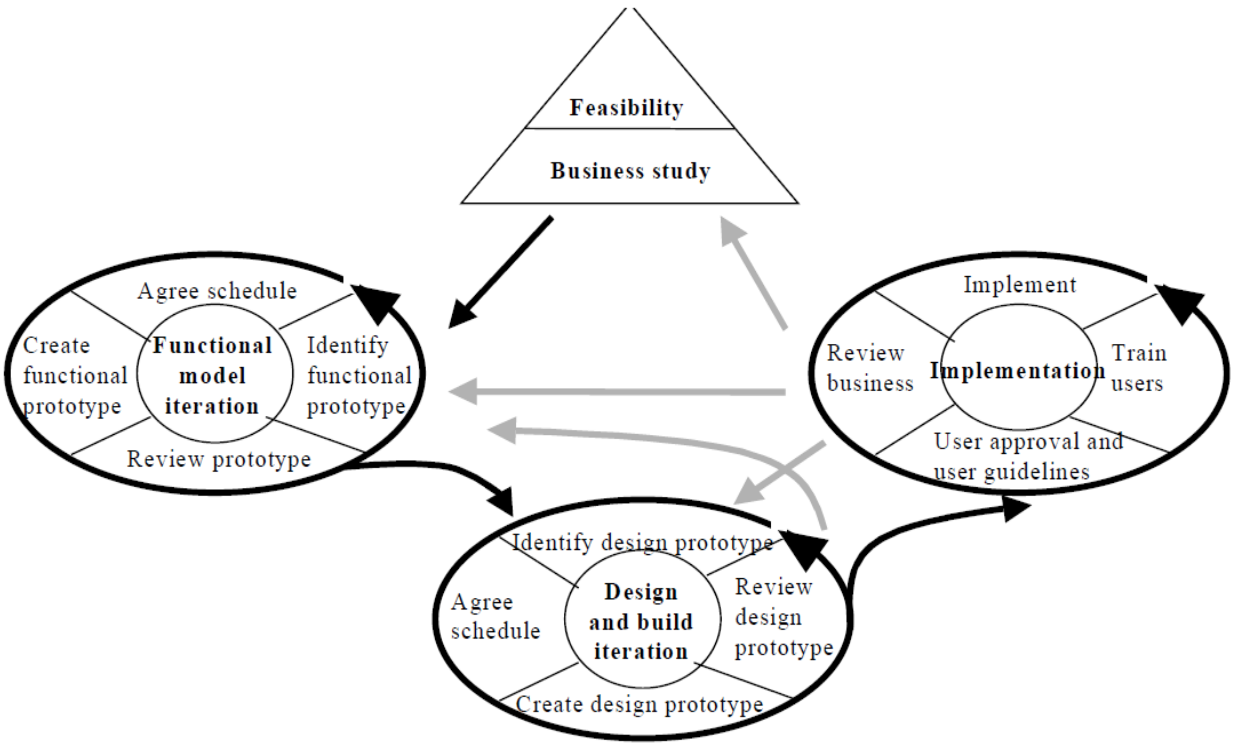

Feasibility
Business study
Agree schedule
Create functional prototype
Functional model iteration
Identify functional prototype
Review prototype
Implement
Review business
Implementation
Train users
User approval and user guidelines
Identify design prototype
Agree schedule
Design and build iteration
Review design prototype
Create design prototype

Figure 14. DSDM process diagram (Abrahamsson et al. 2002).

As seen in Figure 14, the DSDM process contains: Feasibility study, business study, functional model iteration, design and build iteration and implementation. The two study phases are sequential and not iterative inside the phase. The last three phases, which contain the actual development, are iterative and incremental. The iterations are based on time boxes. The time box last for a predefined time and each iteration has to end within the time box. In DSDM a typical time box lasts from a few days to a few weeks. (Abrahamsson et al. 2002)

The *Rational Unified process* (RUP) is a model that is especially suitable to be used in development of object or component oriented systems, an important part of RUP is the use of use cases for modelling requirements. (Ambler 2005)

In Figure 15, the four RUP phases are illustrated.

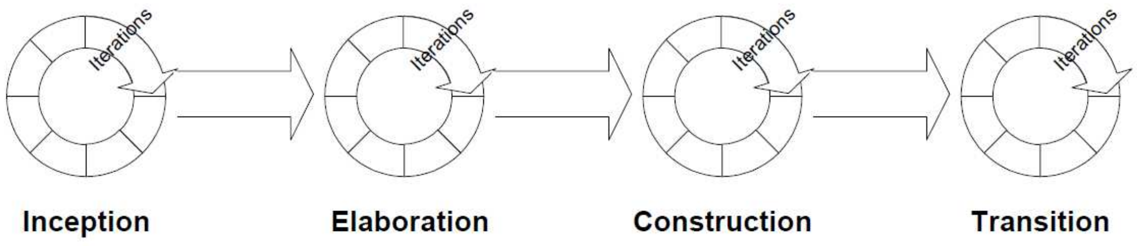


Figure 15. The RUP process (Abrahamsson et. al 2002).

As seen in Figure 15, the RUP process contains four phases: Inception, Elaboration, Construction and Transition. The phases are sequential but the work in the phases is iterative.

The framework of RUP is made of nine disciplines, which are followed inside the iterations. The RUP approach divides all work to the nine disciplines which are Business Modelling, Requirements, Analysis and design, Implementation, Test, Deployment, Configuration and Change management, Project Management and Environment. During each iteration, the team jump back and forth between the disciplines to achieve the goals of the iteration. An iteration always contains a small sub-set of the system to be developed and after each iteration the selected sub set should be ready for a release. (Ambler 2005)

To Summarize, all the Agile development models demonstrate the same basic features, with some minor modifications. In all of the models, the actual development is done in iterations and they all are divided into phases; most of the models also define roles for the people in the project. The models are flexible to the extent that the model is to be followed but, at the same time, open to mixing different features to find the most suitable combination to a particular project.

4.2.5 Agile and CMMI®

Some models suggest the integration of Agile development models and CMMI. Generally, there is a perception that Agile and CMMI contradict each other, which is suggested already in the Agile manifesto. Referring to: *Individuals and interactions over processes and tools, Working software over comprehensive documentation*. Where processes, tools and documentation are presented as less important. On the other hand the manifesto says that it value processes, tools and comprehensive documentation, it simply value Individuals, interactions and Working software more. *(Beck et al. 2001)*

There are three main reasons that position Agile and CMMI against each other. They are *misuse, lack of accurate information and terminology difficulties*. Misuse refers to the fact that CMMI is very often handled as a standard, instead of a model for improving product quality and process performance. The CMMI originated from the context of a certain customer base with unique needs with characteristics of High risk and low trust. The CMMI was introduced to a large industry where a certain attitude was in place already for many years, whereas Agile evolved as a counter measure for ineffectiveness in software development. These facts and the misuse of CMMI as a standard did not support the needs to adapt CMMI to software development. The second reason lack of accurate information has changed lately but it is still one reason for the belief of CMMI and Agile being contradictive to each other. Before the last few years very little research and studies were conducted on Agile and CMMI, the few articles there was about it all came from the Agile community. This lead to the perception of CMMI ignoring Agile, thus enhancing the understanding of contradiction.

The first material covering Agile and CMMI was presented as late as 2005. When CMMI and Agile has co-existed already since 2001. The last reason *terminology difficulties* come from misunderstanding terms from the different communities. Each framework, model, process etc. has its own terminology and the terms are understood under that context. When looking at the same thing from another perspective (for example, framework) the same term might have different meaning, but for the person trying to understand the term it is difficult to forget the previous understanding of that term in an other context. For example in CMMI the abbreviation TDP is used, technical data package, it means a collection of product data from a technical perspective. However in systems acquisition context TDP refers to a specific deliverable of specification documents. A person who is familiar with the systems acquisition context might have difficulties understanding that using CMMI and Agile does not force to deliver the specific deliverables that he associated to TDP, for example MIL-STD-498 version of a TDP. A

Another example is the use of the word *predictable* which might be a red cloth for someone from the Agile community, referring to the difficulty of predicting in software development. The person might think that predictable requires a strict project plan in the beginning of the project that covers the whole project life-cycle. Instead, CMMI is looking for improving the predictability also in Agile development, to reach a higher level of predictability. (Anderson et al. 2008; McMahon 2010; Cockburn 2006)

4.3 Stage Gate Models

Waterfall based models are often used as the baseline when creating company specific models. Product development processes are today strictly steered and controlled in most organizations. The most common product development process consists of phases and checkpoints between the different phases. They are so called Stage-Gate™ or Waterfall models. (Cooper et al. 2002a)

The stage-gate model is a traditional Waterfall based model where each stage ends with a milestone or gate. The gate sets certain criteria for passing and only after criteria are met the next stage can be started. The Stage gate model provides a project progress communication tool and control of quality, as each product has to pass all the gates before being launched and it is easy to communicate the status of the project. For example, if the project has passed Gate 3, everybody in the organisation knows that the project is in the development phase. (Cooper et al. 2002a).

In Figure 16, an example of a stage-gate model is illustrated.

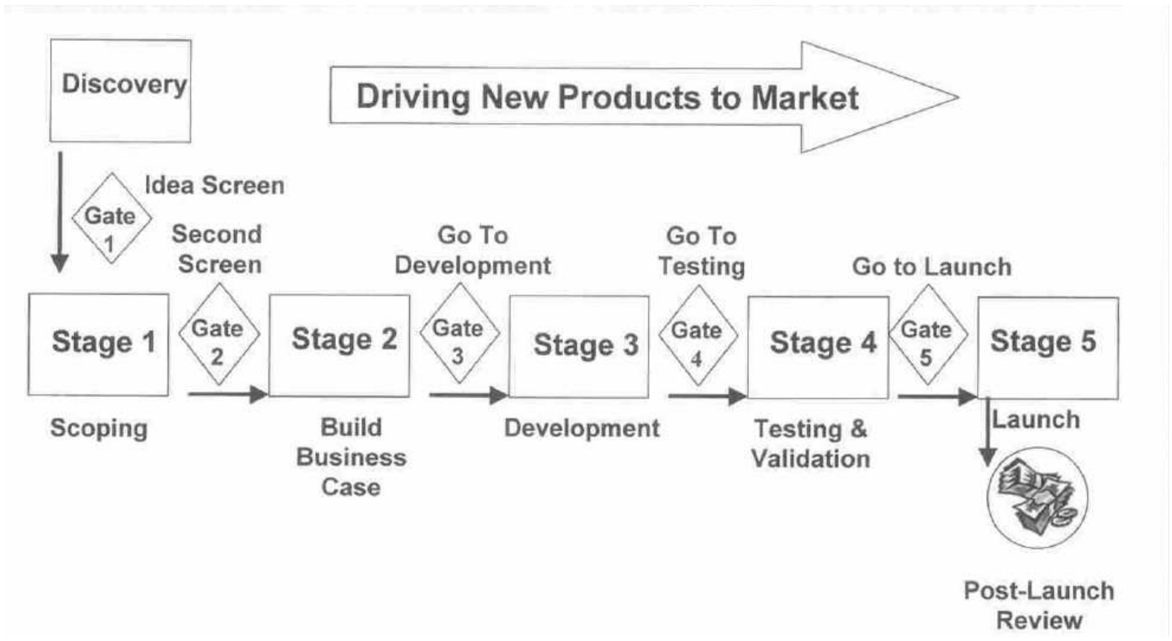


Figure 16. Example of a five staged Stage-Gate model (Cooper et al. 2002a).

As seen from Figure 16, the Stage-Gate model consists of five gates and six stages (including the Discovery stage). Before, between and after the gates the process consists of the following stages: Discovery(Idea screen), Scoping, Build Business Case, Development, Testing & Validation and Design. In addition, it includes a post-launch review. (Cooper et al. 2002a).

Over time several different variants of traditional Stage-Gate methods have evolved to overcome the challenges that a strict Waterfall based method brings. Also in the Stage-Gate model there is a mechanism to allow proceeding to the next stage, without meeting all the criteria for the gate. This allows the project to proceed even when a certain task is late. The unfinished pars are left as concessions for the Gate. The model set targets for when to close the criteria that were not met at the gate review. These concession are followed up and controlled and should be at latest closed at the following gate. In some case a very strict gate control can lead to unnecessary breaks in the project if just one area is late, all the other project members have to wait. (Cooper et al. 2002b).

To summarize, the stage gate model or any variation of it is widely implemented and well known in product development companies. It provides excellent visibility and control for the management as progress is strictly controlled by gates. It also makes it easy to follow up quality criteria as criteria are easy to link to gates. The drawback of the Stage-Gate model is that it is not flexible. It doesn't cope well with delays or changes. Therefore organisations using Stage-gate model put a lot of effort on change management. With different practices to perform changes to project schedule, budget and content.

4.4 Agile to Waterfall Best Practices

Software development companies and development organisations have been studying and considering ways to go from strict stage gate driven product development models (Waterfall models) to Agile software development models. The challenge that most organisations face immediately when trying to gain the benefits from Agile methods is, how to merge the Agile processes with the standard industry processes, without completely killing the "agility" of the new lightweight Agile process and, simultaneously, by meeting the quality criteria that have been defined and fine-tuned for years with the standard process. (Boehm and Turner 2005)

Since the Waterfall model is based on linearity, the output of each phase is the input to the next phase. This practice gives control and makes it easy to follow the progress. The challenge is that when the market changes it will lead to customer needs changing and then the requirements will change. A strict Waterfall model cannot adapt to changes. As each phase ends with a "verification" milestone, changing a requirement would force the project to start from the beginning, or as usually is done, finalise the project with initial requirements and develop the new requirements in the next release. Therefore, risking to "miss" the market opportunity. Most software companies realise the challenges of Waterfall models and have implemented modifications and work-arounds to the Waterfall model, making it a bit more flexible. Even with the modifications companies encounter challenges. (Cusumano and Smith 1995)

Cusumano and Smith listed the top ten challenges for software companies in 1995 to be 1) Inadequate requirements statements, 2) Lack of specific and measurable goals, 3) Architecture design flaws and changes, 4) Inadequate change control systems, 5) Inadequate project status reviews and reporting, 6) Inadequate project metrics, 7)

Lack of open project communications, 8) Lack of clear project milestones, 9) Overly optimistic estimations of project feasibility, 10) Various management difficulties.

Similar lists of challenges in development has also been created in 1960's, 1970's and 1980's, it shows that overtime the top challenges have not changed significantly. (Cusumano and Smith 1995)

For example, Microsoft Corporation is one of the world's largest software companies, which develops successful products since the 1970's. They were one of the companies that very early started to find practices that combined traditional Waterfall models to more Agile like models. Microsoft started to use a variation of Agile development by gradually adapting their software development model from 1988 and forward. As Microsoft grew, the requirements for structure and better quality of software got more important. In the 1980's Microsoft development teams grew bigger and the unstructured way of working could not be sustained. Microsoft products were used by large companies and governmental organisations in mission critical operations, these customers started to require more structure and predictable processes. Microsoft started of with looking at Waterfall based models but noticed quickly that it was not suitable for how they wanted to work. It had too much structure, Microsoft did not like the fact that in Waterfall based models the product requirements are frozen at the beginning of the project and that components are built exactly after specifications. Instead Microsoft wanted the development team to evolve the product designs incrementally, to be able to adapt to changes and to respond faster to customer needs. Microsoft implemented a model that we refer to as the "synch and stabilize process". (Cusumano and Smith 1995)

One of the major changes that Microsoft implemented was the move from functional driven releases to date driven releases. This applied for all products, except Operating Systems. Before the change a release was driven by feature availability. After the change the release was driven by release date and trade-offs to meet the release date were made on the features and functionality. The reason for change was the history of missed dates, which Microsoft and their customer could not accept anymore. Microsoft actual development process was also different from traditional Waterfall models. Microsoft development process was constructed to have 3 or 4 development cycles, each of the cycles consisted of coding, testing and stabilisation phases. Features were grouped to the cycles and the most difficult and important features were developed in the first cycles. The use of cycles is also one of the core principles of Agile development. The Microsoft development process also uses something they call "Vision Statement" and a functional specification from users perspective instead of using complete specification written at the beginning of the development project. This is also known from Agile Development, where we are familiar with the term user stories. In addition the development process requires a new "build" everyday which is verified and tested. The build is used to verify that all new code is functioning and does not break anything. The daily build process helps to evolve the product incrementally and catches interferences between code developed by different developers. Daily builds are also part of many Agile development models. (Cusumano and Smith 1995)

To summarize, the "sync and stabilize" model developed by Microsoft mixes practices from Waterfall based models with practices that later became cornerstones of Agile development. With this mixed approach, balance was found between an unstructured creative environment and a structured uncreative environment.

4.5 Analysis of Vital Elements from Best Practices

The vital elements visible in any of the analysed product development models consist of a process, roles, rules and deliverables. The process illustrates the life cycle of the product development and is usually divided to different phases. The roles reflect the people working for the project, in the Agile models the roles are very often defined and in a central position; whereas in traditional Waterfall model the roles are not that important from model perspective. Instead the companies using the Waterfall model very often define the roles based on resources and type of organisation. The models very often define rules and deliverables that tie the process and people together.

The major difference between the different Agile models is mostly concentrated in how prescriptive they are. Some of the models are extremely prescriptive telling exactly what should be done, when it should be done and by whom. Whereas other are less prescriptive just providing tools, means and guidelines but the implementation and how they are applied are up to the project.

In Figure 17, different models are aligned on an axis ranging from prescriptive to adaptive.

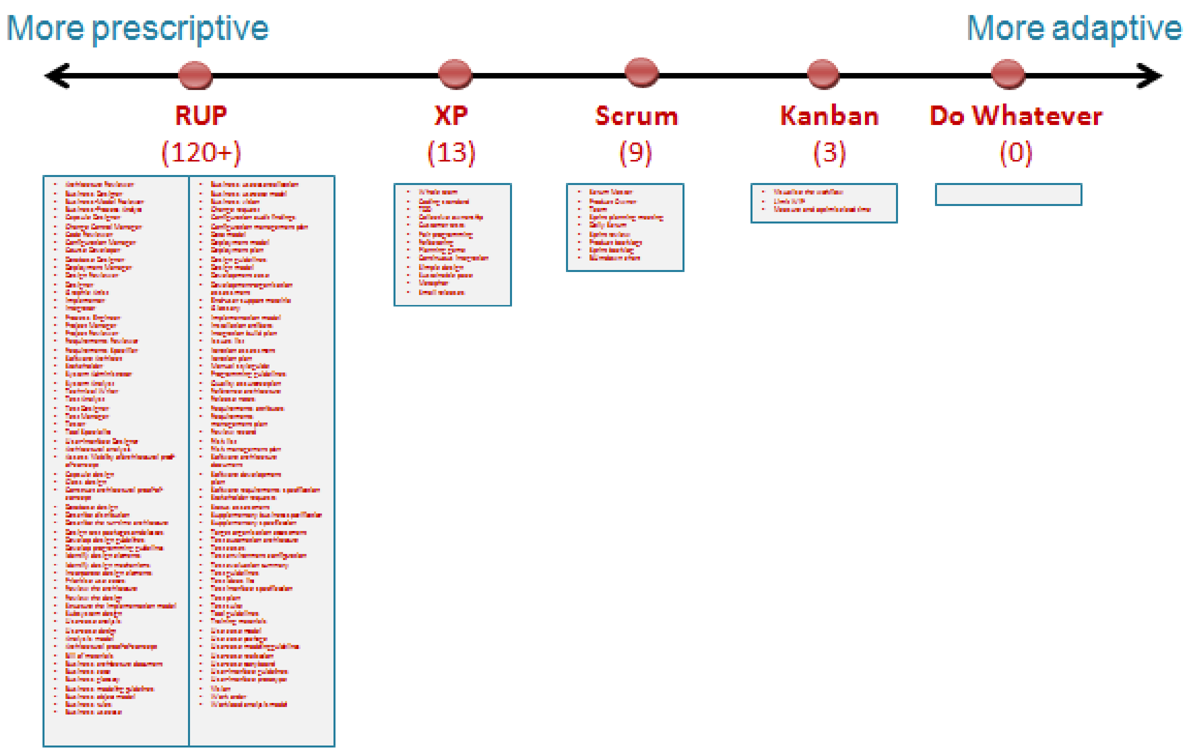


Figure 17. Prescriptive versus adaptive, in Agile models (Kniberg 2009: 8).

As seen in Figure 17, different models are more or less prescriptive. Comparing for example RUP and Scrum, it is seen that RUP is very prescriptive with over 120 different components (roles, artefacts and activities), whereas standard scrum only has 9 components. The fewer components a model has, the more freedom for adaptation this model can demonstrate. A model which has fewer components is easier to adapt to the organisation specific needs, but on the other hand it demands more effort to define the actual model and practices. (Kniberg 2009: 8)

Each of the Agile development models consists of a set of components. They all have some vital elements which differentiates the models from each other.

In Table 9, the 7 discussed product development models and their vital elements are listed.

Product development model	*Vital elements*
Scrum	Length of sprints do not changes, all sprints end with a "done" release. Events, roles and artefacts
Scrumban	Scrum combined with Kanban. Kanban board added to Scrum. WiP limit to control workload.
Extreme programming	Collection of best practices from software development taken to the extreme. Process, roles and practices
Crystal	Right model for right project, different colour coding based on project nature
DSDM	5 phased process, fixed schedule and resources, functionality is adaptive
RUP	9 disciplines inside iterative sequential phases.
Stage-Gate	Project goes through stages which are separated by gates with passing criteria.

Table 9. Vital elements of the different models.

As seen in Table 9, each of the models has a theme, i.e. something new that it brings to the scene. In addition to the vital elements listed in Table 9, all the Agile models fulfil the core values of Agile product development and all are based on iterations and increments. On the contrary, the Stage-Gate model is a traditional Waterfall based model, with some tools to adapt to changes. Based on these conclusions, the Agile models can be combined with CMMI, since CMMI does not impact the selection of Agile model. In this study, we have not found any reason for one Agile model integrating better with CMMI then another.

To get an understanding of the workflow of the different Agile models compared to standard Waterfall-based development, these models are summarized in the following comparison matrix. The comparison matrix visualizes the models in sequential phase format, using the Stage-Gate model as reference.

In Figure 18, the different models and their processes are illustrated in a linear comparison matrix.

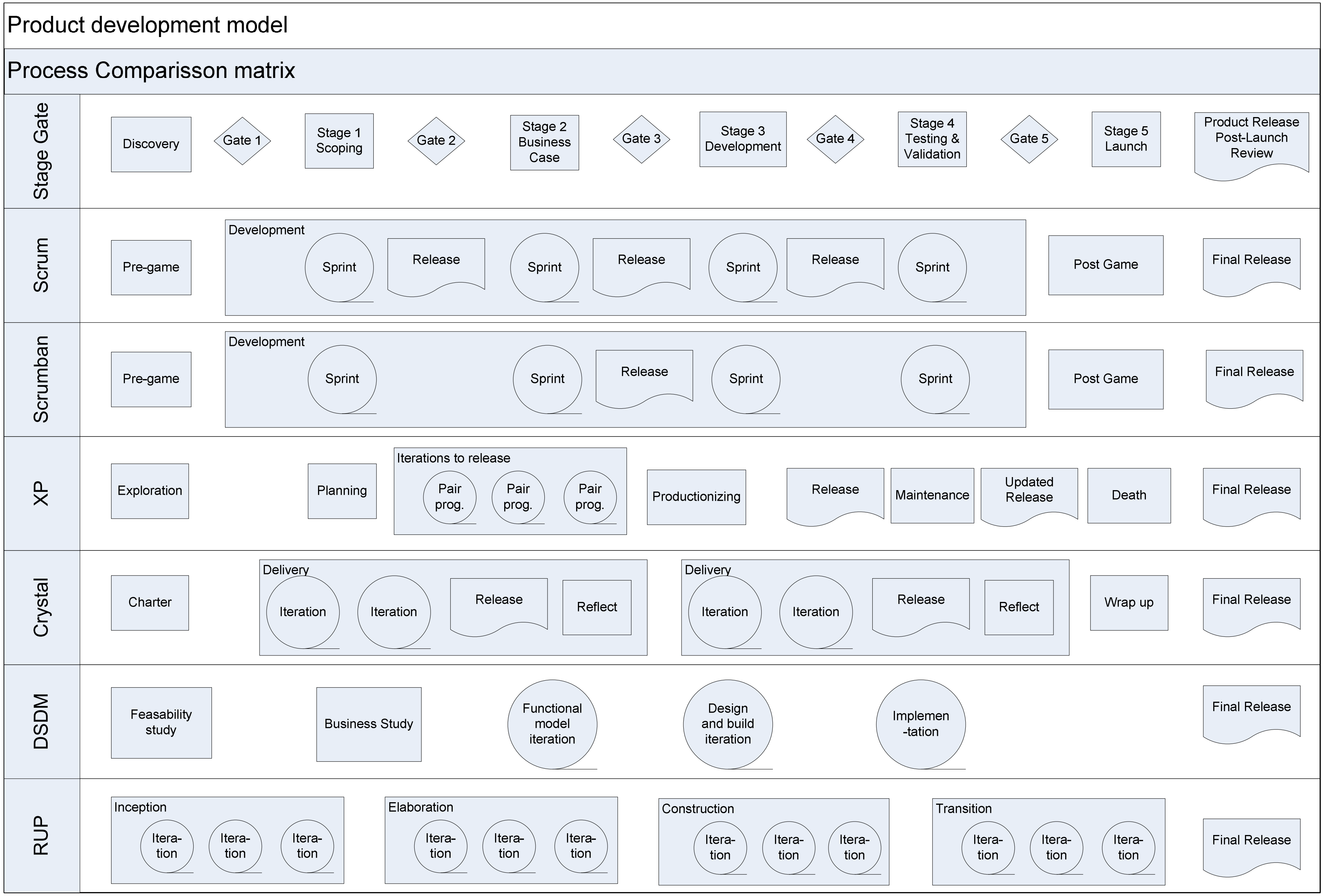


Figure 18. Product development models in a comparison matrix.

As seen in Figure 18, the Stage-Gate model is sequential and easy to present in a linear model. Although some of the Agile development models do not suggest a process map, they still can all be presented in a linear way, as they are all product development models for projects with different phases. Projects always have a starting phase the ramp up phase and the target is always to produce a product release. The major difference between the stage gate model and the Agile models is that all of the Agile models are based on some level of iterations.

In addition, one major difference is that the Stage-Gate model is the only model that suggests formal reviews or gates between the different phases. None of the Agile models has as a fundamental element addressed the control of moving from one phase to the next one. Finally, all of the Agile models, excluding RUP, include a starting phase before entering the iterations. RUP is the only model where even the work of the first phase is done in iterations.

In Figure 19, the comparison matrix in Figure 18 is mapped against different phases common in most product development models: concept development, system-level design, detail design, testing and refinement and production ramp-up (Suomala 2004).

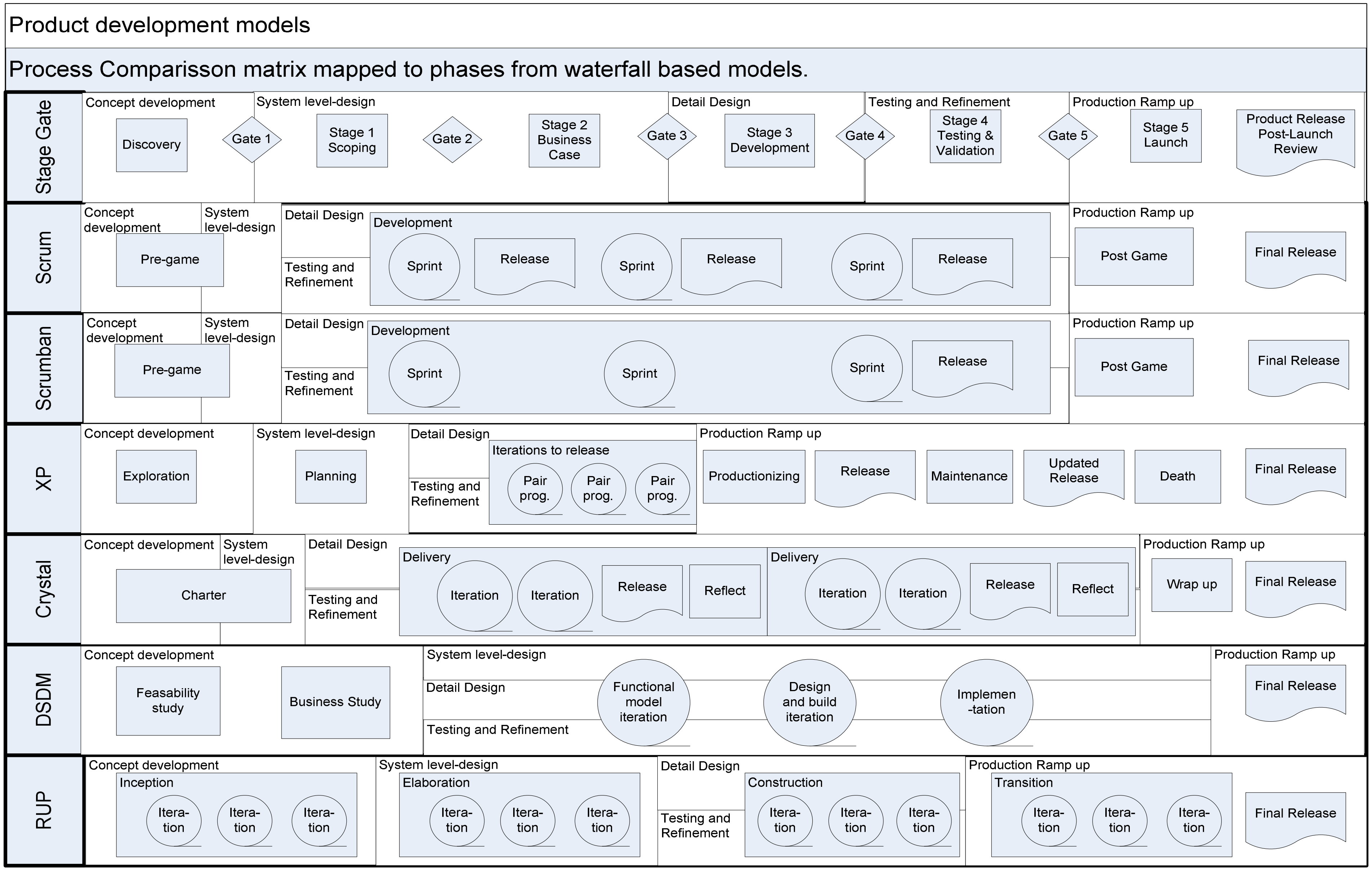


Figure 19. Model comparison matrix mapped project phases.

As seen from Figure 19, if compared to a traditional Stage-Gate model, even the Agile models can be divided to the normal project phases. We can also see that in all of the Agile models the Detail Design and the Testing and Refinement are merged in to the development phase and in Agile models these two phases are always inside iterations. Of all of the Agile models, RUP is the only one that is best mapped to the phase division in Waterfall-based models. Even though RUP is based on iterative development, it has four phases that are similar to Waterfall models, as noticed earlier the Detail Design and Testing and refinement are part of the Construction phase.

Overall, all the analysed Agile models focus on practices for product development. They all include practical components for how to perform the product development. Most of the Agile models divide the components to three parts *events, roles* and *artefacts.* In addition they all include some level of project breakdown to phases. The lack of gates between phases is discovered to be common to all of these Agile models. The fact that the Agile models do not provide gates makes their mapping against the Waterfall-based models more difficult. The lack of gates and strict gate criteria is also a major difference between the Waterfall and Agile models. This is the reason why large companies, used to utilizing the Waterfall based models, usually resist the use of Agile models. Because Agile models as such do not address the need for management visibility and provide no tools for quality requirement follow-up. Therefore, it is evident that the Agile models need to be adjusted to the practices outside these models and improved in terms of management visibility if this is required by the company.

To summarize the analysis, waterfall based models are well accepted and provide tools and guidelines for how to ensure quality and proper follow up of product development projects. It focus on ensuring that projects reach the end, as defined in the beginning of the project and makes it possible for the management to keep control of on-going projects. The downside is that a lot of effort is wasted on gate preparation and specification and planning of issues which might be unknown in the beginning. Therefore there is a risk that projects are delayed, as there is little room to adapt during the project when unexpected issues and challenges are met. The customer need might also change during long projects and there is a risk that the final product does not meet the customer need. Agile models provide tools to learn and adapt during the project. The project content can be adapted during the project and the delivery date is always kept. Agile models provide means to prepare and adapt for unexpected issues and challenges. It is more likely to meet the customer need as it shortens the delivery time and provide tools to adapt during the project. Overall Agile models provide more flexibility to the product development projects, meaning more options to adapt project content during projects and possibility for faster product development projects. The downside of Agile models is that they provide little or no tools for management to follow up on project progress. Too much uncontrolled flexibility can also be seen as a risk and it can scare the management team as it makes it more difficult to estimate and plan the future.

5 Model Development

This section presents an overview of the model development process based on the workshops and discussions in the case company as for Prototype 1, 2 and 3 of the new product development model.

5.1 Overview of Model Development Process

Section 5 presents the development of the new model based on the workshops (Workshops 1-3, Product Decision Board meeting and Workshop 4.), interviews, discussions and the Lessons Learnt session conducted in the case company. Backed up with the findings from the literature reviews and best practices search, they led to the creation of the proposed model (in Section 6).

The results of the development process are collected and presented according to the prototypes discussed/created in these events:

Structure of Section 5	*Event*
Section 5.1 Overview	---
Section 5.2 **Prototype 1**	*Workshop 1.* The project core team held the workshop to create Prototype 1 for the new model.
	Workshop 2. Presentation of Prototype 1 to the stakeholders and owners of the project process.
Section 5.3 **Verification of Prototype 1**	*Workshop 2.* Verification of Prototype 1 with the stakeholders and owners of the project process; collecting questions to be addressed in Workshop 3.
Section 5.4 **Prototype 2**	*Workshop 3.* This workshop was held with members from the project core team, process owners and external experts from the subcontractor. The outcome is the input needed for creation of Prototype 2
	Product Decision Board Meeting: Prototype 2 is presented to the Product Decision Board. The Product Decision Board approves the use of a non-standard development model and gives recommendations for further activities.
Section 5.5 **Verification of Prototype 2**	*Workshop 4.* The project core team analyses Prototype 2 based on the Product Decision Board recommendations. The creation of the new model for use in the pilot project.

Table 10. Structure of Section 5.

As seen from Table 10, there was a series of events held in the case company to create the proposed new product development model presented in Section 5. Each of the events was documented in the case company meeting minutes. The events which included the prototype creation were also documented by taking notes on a whiteboard. The whiteboard drawings were documented with photographs and later transferred into PowerPoint presentations.

5.2 Prototype 1

Prototype 1 was created together with the project core team at Workshop 1. It was based on the lessons learnt from the previous project and on the analysis of theoretical models presented in Section 4.

At Workshop 1, a shared understanding was reached that the actual product development model chosen was not the most vital issue for the new model development. The key to success was how to adapt it to the existing case company processes and provide the required visibility on project status for the management team. If the new model were transparent enough, it would get the support from the management side, which is essential for the project success. It was decided that Scrumban would be the reference model to be used. The reason behind this decision was that Scrum is a well-known and widely adopted product development model, already familiar to the subcontractor and the case company. It was agreed that, with pure Scrum, there could be a risk that the necessary visibility to the project might not be reached. Therefore, some additional elements were needed. Since Scrumban (Scrum and Kanban) already included some useful adaptations to pure Scrum, the first step at Workshop 1 was to match the Scrumban model to the C1 process in the case company.

Figure 20 shows the key milestones of Prototype 1, mapped against the C1-process gates.

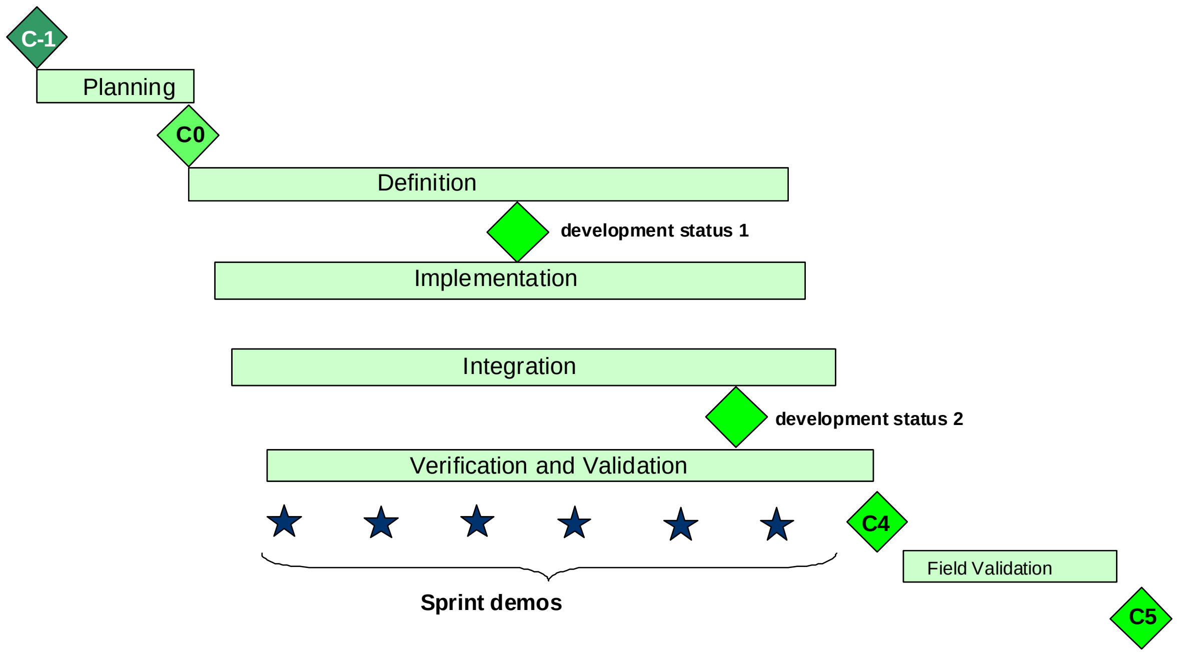


Figure 20. Prototype 1 of the development model against the C1 process.

As shown in Figure 20, the different phases from the C-1 process are still visualised. To be able to make a comparison of the standard model and the prototype 1. We can see that *Planning* phase is still between C-1 and C0 gates. Main difference is that Definition, Implementation, Integration, Verification and Validation phases are continuous from C0 gate to C4 gate, compared to C1-process where C1 gate ends the definition phase, C2 gate ends the Implementation phase, C3 gate ends the integration phase and C4 gate ends Verification and Validation phase. The Field validation phase is between C4 gate and C5 gate as in the C1-process.

Table 11 compares the standard C1 process gates to the Prototype 1 proposed gates.

Gate	***Prototype 1 Gates***	***Description***
C-1	C-1	Content proposal ready
C0	C0	Project commitment
C1	Development phase	Commitment confirmation and planning ready
C2	Development phase	Implementation ready and ready for integration
C3	Development phase	Ready for verification and permission to tender
C4	C4	Ready for customer deliveries and verification ready
C5	C5	Ready for volume deliveries and Field validation ready

Table 11. C1 process gates mapped against the proposed gates in Prototype 1.

Table 11 matches the C-gates (C-1 to C-5) and their descriptions against the Prototype 1 proposed gates. As seen from Table 11, Gate C-1 is needed for a project to start and to get the needed resources for the project planning. In addition, it was stressed that the project planning phase is needed to define the scope and targets for the project. At Gate C-1, it would be too early to present the product backlog, therefore it was seen that also Gate C0 was necessary to introduce.

To get the project commitment and secure budget from the management team, it was proposed that Gate C0 in Prototype 1 C0 would also include the commitment confirmation and the planning ready; whereas in the C1-process the commitment confirmation and planning ready are presented separately in Gate C1. The Planning ready stage in the prototype means that the Product backlog is made, with the work estimates collected and all quotations from subcontractors received, which means that the project budget is known.

In the C1-process, one of the main deliverables in C1 gate is that all the project specification work is done (i.e. the SFS - System Functional Specification - for the project is completed or updated). On the contrary, in Prototype 1 only the Product backlog and user stories are completed at C0 gate; therefore, the actual specification work becomes part of the 3-week iterations of the development phase.

In Prototype 1, the actual development phase starts after gate C0. In the C1-process the Gates C1-C3 are used for ending a phase and entering the next phase, and they also help maintain visibility for the management and following up the project quality and progress. As the prototype one has continuous development in iterations from C0 gate to C4 gate it was proposed that Prototype 1 has to have two Development Status Reviews, between Gates C0 and C4. The purpose of the Development Status Reviews is to make up for the removal of gates C1 to C3 and to create transparency of the project and maintain management trust, by avoiding uncertainty as for the project being on time and progressing as planned. Development Status Reviews are also important to have the possibility for the project team to flag any possible challenges or delays in the project. This is a minor benefit as the process already allows for any project to have additional PRR (Project Redirection Reviews) at any time for the purpose of informing timely about possible project delays or budget challenges.

Gates C4 and C5 also served their purpose in Prototype 1 as they did in the C1-process. Gate C4 ends the development phase and serves as the gate where the product is ready for the first customer delivery. It is also the transition point from the development phase to the field validation phase. The field validation phase is the same in Prototype 1 as it is in the C1-process, although it was expected that the field validation phase could be shorter in Prototype 1 than in the projects using the standard C1-process. The reason for an opportunity to have a shorter field validation phase is due to the fact that Agile development provides a possibility to perform early customer demonstrations and get feedback and comments about the product before the actual field validation phase.

The field validation phase ends at Gate C5, as is normal in the C1-process. Gate C5 indicates the readiness for volume deliveries of the product to the customers, and it is also the transition from the product development project to the product maintenance cycle. At Gate C5, the project is closed and the resources are released to other projects.

For the new model, it was also proposed that, at Gate C0, a progress plan should be made to show which items would be done at each of the Development Status Reviews. This could be used in Prototype 1 as a baseline and, compared to the actual Feature Board at the time of the Development Status Review, to show if the project is progressing according to the plan at Gate C0.

As agreed, in Prototype 1 the development phase is following the Agile product development models, with Scrumban chosen as the reference model. The development phase consists of 3-week iterations typical of Agile models. Each of the iterations in prototype 1 should end up with a sprint demo at the end of each iteration. It was proposed that, to better follow the project progress, the project team should use an adapted Kanban board called *Feature Board,* as the focus would be shifted to present how the features are progressing towards *the Done status* in the localization phase.

Figure 21 shows an example of a feature board, proposed to be used for project progress follow up in Prototype 1.

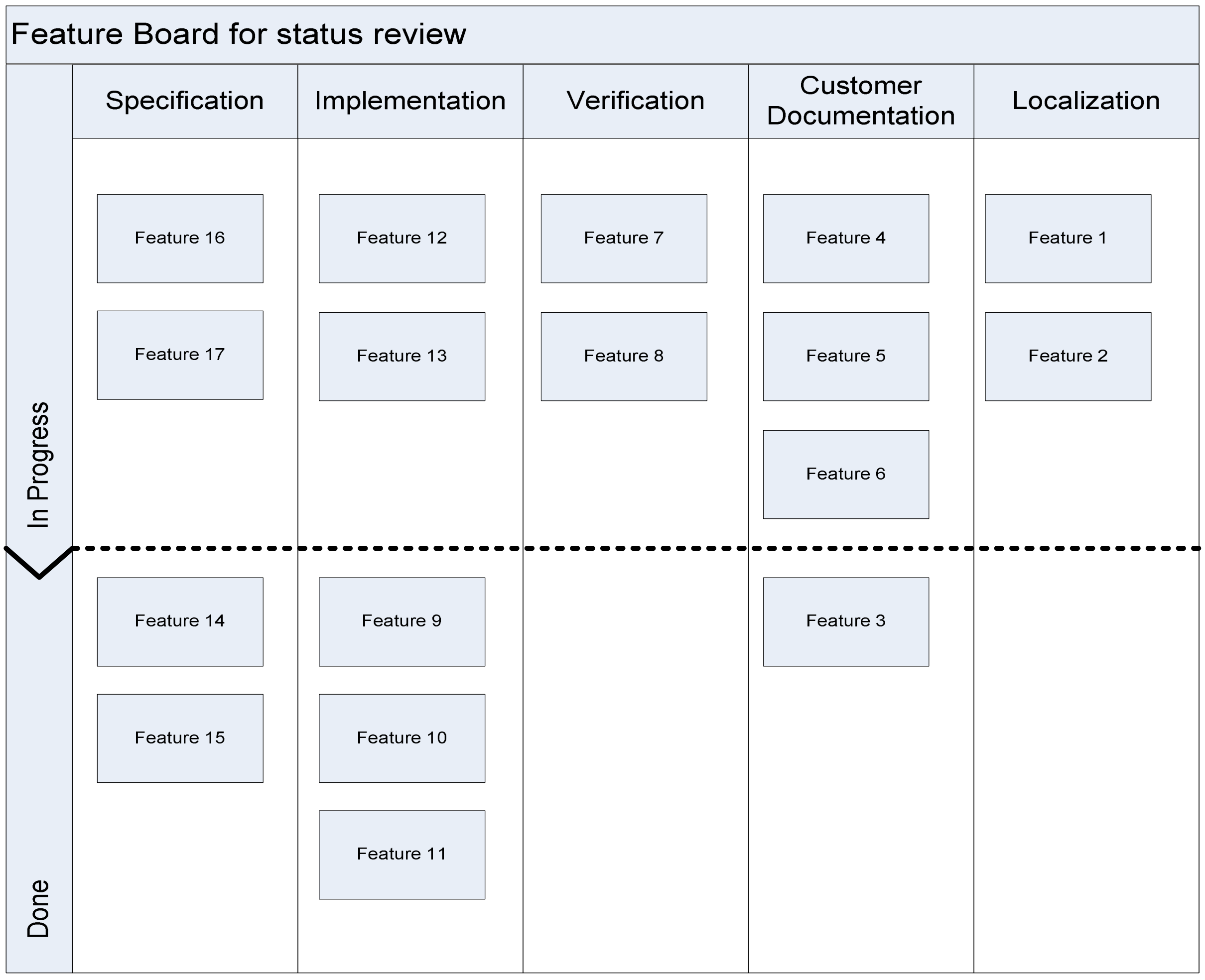


Figure 21.A feature board example from Prototype 1.

As seen from Figure 21, the progress of the features can be followed with the Feature Board, which serves mainly as a tool for the project manager. In addition, the Feature Board can also be made available for the management team and be presented at the development status reviews.

Other important element discussed at Workshop 1 was a Product Backlog, which can contain all the features proposed to be implemented in the project. The Product Backlog can also include priorities for all the items and initial work estimates. The proposal was that the work estimates should be based on *story points*. The story points can be calculated as roughly as possible, with the purpose to get some initial understanding of the project size. The terms *small, medium* and *large* can be used for work estimates, each term reflecting a certain amount of story points, for example: Small=3 points, Medium=5 points and Large=8 points. The Product Backlog, borrowed from Scrum, is owned by the product owner. At the case company, though, the title *product owner* is not used. Therefore, the proposal was that, in Prototype 1, the product business manager should own the product backlog and any changes to it have to be approved by the product business manager.

To summarize, Prototype 1 was based on Scrumban and the practices from Scrumban which provide tools for the project management and product development. To meet the quality requirements and visibility requirements by the case company, some elements from the C1-process were retained. Using the standard C1-process gates wherever possible, it could help to communicate the product development project progress to the management in a "language" which they are familiar with. As practices are already in place for auditing projects at the gates, using the standard gates as references could help to provide visibility to the project. In addition, if using the same gates as reference, the quality of the project and the project outcome is comparable with other projects. On the other hand, it was important to differentiate from the standard C1-process to be able to benefit from the Agile development model. Therefore, Gates C1 to C3 were removed and replaced by the development status reviews. During Prototype 1 creation, it was also proposed to use Gates C1 to C3 instead of the development status reviews, with a joint Gates C1 and C2 instead of Development status review 1 and Gate C3 instead of Development status review 2. But that idea was discarded as it would lead to the same outcome and challenges that were met during the Release 1 Project. Instead, the C1 to C3 gate criteria were reviewed and mapped to the Prototype 1 gates, to provide means to perform the necessary project audits.

5.3 Verification of Prototype 1

The aim of the Verification of Prototype 1 was to present it to the key stakeholders within the case company with the purpose to get feedback and find possible week points in Prototype 1. These week points were used as input to the creation of Prototype 2. The verification of Prototype 1 took place in Workshop 2, and its results are collected and documented. Workshop 2 was also the point where approval for proceeding to Workshop 3 with the external company was gained.

In Workshop 2, several questions, findings, actions and recommendations were listed that should be taken into account in Prototype 2. Table 12, presents the most important of the questions, findings and recommendations discussed at Workshop 2.

Id.	*Question, Finding, Recommendation or Action*
1	Question: When is the ready to sell achieved?
2	Question: What is the process to reprioritize if needed?
3	Question: How to catch up with delays during the project?
4	Question: Is a regression testing needed, when should it take place?
5	Question: Should the proposed Gate C0 actually be called C1? Ref. contracts and money tied.
6	Question: Does the proposed model fit for hardware development projects?
7	Recommendation: Divide the content into breakable and unbreakable (60/40)
8	Finding: It shall be made clear that all features in the product backlog are not implemented during the project.
9	Recommendation: All items/features should be divided to sprints before Gate C0/C1.
10	Recommendation: All work estimates should be split to small packages.
11	Finding: Model should not try to solve all project issues. Focus on the project to get started.
12	Recommendation: Major changes to the product backlog should be approved by the Product Decision Board.
13	Recommendation: Create a list of main differences between prototype and standard model.
14	Action: The C1-process deliverables from C1 to C4 should be mapped to the proposed model.
15	Action: Go ahead with Prototype 2 creation and plan Workshop 3
16	Action: Start creation on the new model and prepare C-1 presentation for the pilot project.

Table 12. Issues discussed in workshop 2.

Overall, at Workshop 2 it was concluded that Prototype 1 was a good basis to continue a new model development. It was also agreed that the work should continue, and the next step should be to go for Workshop 3 and create Prototype 2, and start preparation of the C-1 presentation of the pilot project.

5.4 Prototype 2

Prototype 2 was created based on the results from Workshop 3 held together with the project core team and additional key stakeholders. It was based on Prototype 1 and the results of the Verification of Prototype 1.

As the Prototype 1 frame was approved at the Verification of Prototype 1, no major changes were made to the main parts of Prototype 2. One semantic change was made to the naming of the Development status review; in Prototype 2 they were called Checkpoints 1 and 2. These gates in Prototype 2 are presented in the Table 13.

Gate	*Description*
C-1	Content proposal ready
C0	Project commitment
CP1	Checkpoint 1
CP2	Checkpoint 2
C4	Ready for first customer deliveries
C5	Ready for volume deliveries and Field validation ready

Table 13. Gate naming and their descriptions in Prototype 2.

As seen in Table 13, Gates C-1, C0, C4 and C5 were made similar to the gates in Prototype 1, but the development status review gates were replaced by two checkpoints. In the creation of Prototype 2, the actual content of the gates and the phases in between was developed further to a more detailed level.

According to the more detailed structure, Gate C-1 contains the Initial content, Schedule, Budget, Business Case and the plan to reach Gate C0. For the proposed pilot project, Gate C-1 also contains the presentation of the new model, and Gate C0 contains the product backlog. The product backlog includes the division of the project content to breakable/unbreakable content. In addition, Gate C0 includes an updated budget and schedule, and Gate C0 also includes the named resources and the project plan to get from Gate C0 to Gate C4.

The Verification of Prototype 1 questioned the opposition of C0 versus C1 gates. In Prototype 2, this recommendation was considered. The outcome was to keep Gate C0 and project commitment in Prototype 2, as it was in Prototype 1, as this was considered a matter of terminology and the project team wanted to highlight early the difference between the proposed new model and the standard model. Using the standard project model, Gates C0 and C1 are sometimes combined. Calling Gate C0 as Gate C0/C1 would not differentiate the model from the standard C1 process. Therefore, the project team decided to differentiate from the standard process as the gates and gate criteria become different in the new model.

An important addition that Prototype 2 suggested was the creation of a product backlog in the planning phase, presented and approved at Gate C0. The product backlog purpose is to manage everything that is done during the project. The product backlog is more than just a tool to manage the software development. Instead, the Prototype 2 suggests that all work could be managed by the product backlog. This includes, for example, customer documentation, service creation and sales capability creation. An example of the product backlog is given in Appendix 5. It was included in Prototype 2 because, by putting everything in the product backlog, the project manager's work is not overloaded. In the standard model, the product manager has to trace down many tasks in project management in addition to the actual Agile software development. For the new model, putting everything on the backlog would allow for the project manager to work effectively with both models, managing the supporting tasks as is normally done and software development tasks, as in Agile models.

In Prototype 2, as it was in Prototype 1, the mapping of the C1-process gate criteria list is required to meet the CMMI criteria used in the case company. In case of CMMI audit this document is needed to show that the required CMMI maturity is managed. The C1-process gate criteria are used as a project follow up tool by the project manager in standard projects, mainly to check before each gate review that all deliverables needed to pass the gate are done. However, due to the introduction of the product backlog, in Prototype 2 this document is not used, instead the project manager will follow up on progress using the Product Backlog. At Gate C0, the mapping of C1-process gate criteria deliverables is presented to the PDB and GMR. The deliverables expected from the C1-process are reviewed and all deliverables are mapped to the new project model. For each deliverable, a justification is done for the necessity of the deliverable. In prototype 2, deliverables that cannot be justified by the core team are dropped out from the list of deliverables. This work was documented in the development of the gate criteria list of the C1 process (see Appendix 4).

Another addition in Prototype 2 was the checkpoints. The function of the checkpoints is to present the current progress status with the Feature board and follow up on the budget and schedule. In addition, the second checkpoint contains the permission to sell the project product release. The main purpose of the checkpoints is to close the gap between Gates C0 and C4, and to provide visibility for the management of the progress.

In Prototype 2, C4 gate was kept as it was. With gate C4 being the point where test results are reviewed and criteria for going to customer delivery are checked. It is an extremely important gate since after it the product is delivered to the first customers and the quality of product has to be good enough to meet the customers' expectations.

The field validation process and following C5 gate were also kept unchanged. It was proposed, though, that the field validation and C5 should be further investigated. Once again, it was discussed that the new model might make it possible to shorten the field validation time, as identified already during Prototype 1 creation.

Prototype 2 also suggested that the detailed specification work should be done at the beginning of each sprint, instead of fixing all the specifications early in the project. This would remove the need for heavy change management processes, as issues learned early in the project can be used in the specification work later in the product development project.

Figure 22 illustrates the gates and phases in Prototype 2.

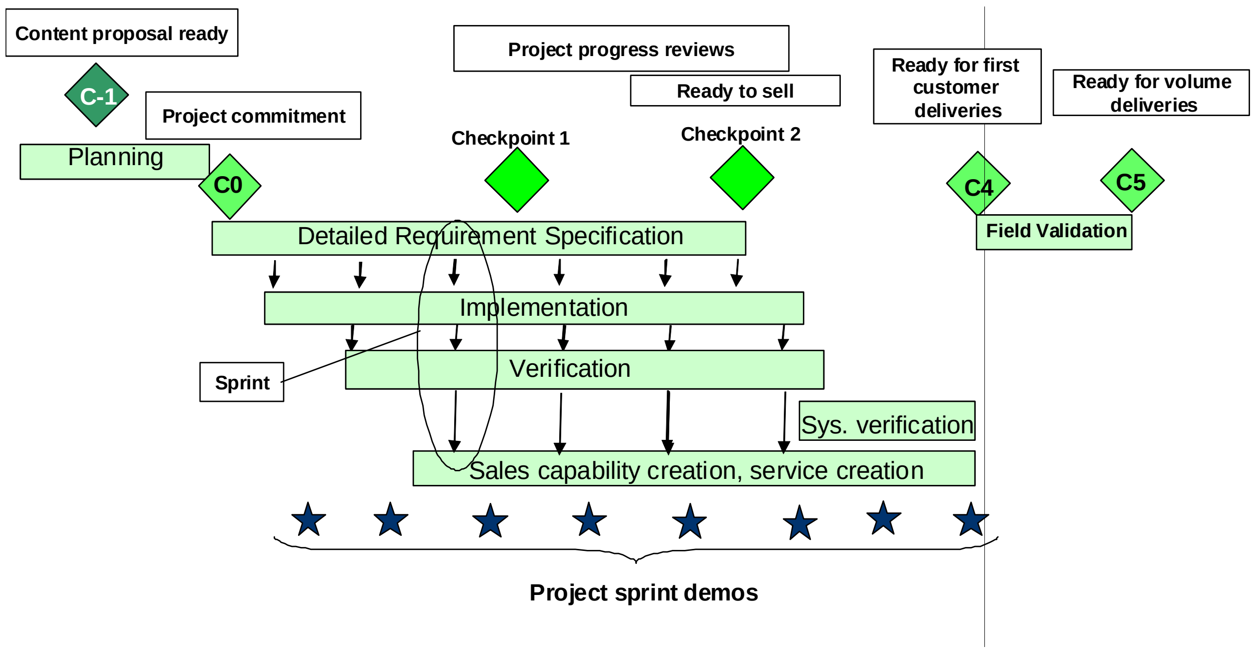


Figure 22. Prototype 2 of the development model in the C1 process.

As seen in Figure 22, the milestones and phases are developed in more details in Prototype 2 as compared to Prototype 1. In addition, Prototype 2 includes the ready-to-sell milestone and a system verification phase which includes the complete regression testing of the software after the implementation phase has ended. Moreover, Prototype 2 presents the workflows inside the sprints/iterations, where an item or feature goes to specification in one sprint, implementation and integration testing in the next sprint, verification testing after that, and finally proceeds to the service creation, including sales capability and documentation creation.

As in Prototype 1, Prototype 2 suggests that the product backlog is created in the planning phase and presented and approved at Gate C0. In the product backlog, the item/features on the product backlog are divided to breakable and unbreakable content. With breakable content we refer to the features that can be cut away without impacting the customer commitment. Unbreakable content means the features that must be present in the final release of the project. By distinguishing these features we achieve the possibility to work in a schedule and resource-driven way, instead of a functional-driven manner. It is considered an advantage for the new model as the company usually fixes the budget and release dates; therefore, there need to be some flexibility in the project to be able to apply Agile development principles to the new model in which initial planning does not include the final specifications. By implementing the concept of breakable and unbreakable content, a possibility is gained to drop those features for which the implementation is more challenging then expected.

Another important aspect provided the product backlog for planning is an opportunity to sort the features by importance and complexity. The target here is to implement the most important and challenging features first. By implementing the most complex features first, the project team can get the information about delays as early in the project as possible. The implementation of the most important features first also guarantees that some less important and breakable content is left to cut out at the end of the project, if it turns out to be necessary. The concept of breakable/unbreakable content and priority/complexity order in introduced in Prototype 2 are borrowed from the standard procedures of Scrum and other Agile models.

Figure 23 presents the sprint/iteration with the different elements and events inside each sprint.

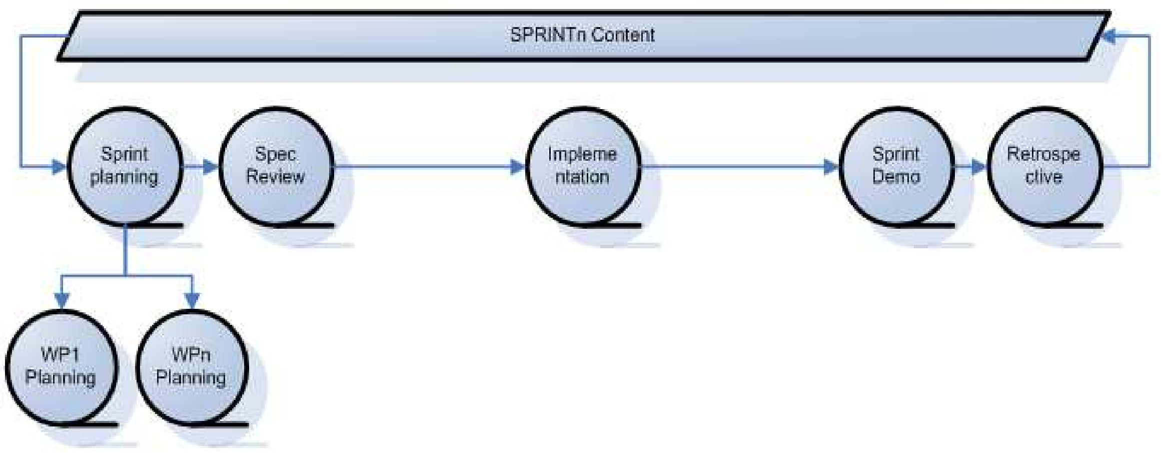


Figure 23. The sprint cycle of Prototype 2.

As seen in Figure 23, Prototype 2 also defined the actual iteration or sprint cycle. Sprint cycles start at Gate C0 and they continue in 3-week cycles until Gate C4. Each Sprint contains four main objects: *Sprint planning, Specification review, Sprint demo* and *Retrospective*. Between the main objects lies the actual implementation work, specification work etc.

Each sprint starts with *the sprint planning*. In the sprint planning meeting, the sprint backlog is created from the product backlog. The Feature board is updated and new items are taken from the product backlog, if there is room for new items on the feature board. The Sprint planning meeting can also contain smaller work package (WP) planning meetings. The WP planning meetings are used if there are smaller teams inside the project, for example, customer documentation or System verification. If the WP planning meetings are used, then a WP manager is also needed who maintains the WP specific backlog items. Even though the WP specific backlogs are used, the product backlog is the master document and all items are stored in the product backlog.

The specification review is the second main object. In the specification review, key project members review the specifications done by the system architects and approve of them before they go to implementation. The specification reviews plays an important role, primarily for getting the actual feasibility of specification from the actual developers, and secondarily as an information sharing tool among the project team.

At the end of each sprint there is *a sprint demo*. During the sprint demo, everything that has been implemented during the sprint is demonstrated to product manager and other project members. Dedicated persons from the management team are also invited to the sprint demos, to enhance the visibility of the project. The proposal is that the sprint demo includes demonstration of everything done during the sprint, not just new software, but also, for example, new documents created etc.

The last object of the sprint cycle is *the retrospective*. The purpose of the retrospective is to evaluate how the sprint worked out, and if something should be changed or improved for the next sprint. The retrospective is a key element in Scrum, and it should lead to continuous improvement of the process, to immediately learn from the last sprint and adapt the next sprint. It is different from the standard C1-process projects, where a lessons learnt session is usually held at the end of the project and the findings are taken to the next project.

5.5 Verification of Prototype 2

Verification of Prototype 2 was done at the actual project C-1 presentation to the Product Decision Board. The Product Decision Board (PDB) is the forum which decides and allocates budget to all projects. It is also the forum which approves the milestones to proceed in a project. Therefore, it also has the mandate to approve the milestones that will be followed in the project. The verification of Prototype 2 was the point where decision was made for Prototype 2 to be allowed to be used in the pilot project, and what type of the standard process should be used. Based on the recommendations and decision given by the PDB at the Verification of Prototype 2, the final project plan for the new model was prepared for the pilot project.

Prototype 2 was approved by the PDB and the finalisation of the new product development model started. The PDB questioned the need for checkpoints explaining that the PDB was a forum for decisions, less interested in project follow-ups. Instead, they recommended that the standard project follow-up process should be used for this purpose. The project follow-up meeting are held ones a month and serve as a point where all the project managers present the progress of their projects. It was suggested that the Feature Board would be the tool for the project follow-ups.

According to the PDB recommendation, the first checkpoint was removed from the final proposed model. The second checkpoint was kept as it includes also the permission to sell criteria, which needs the PDB approval. Apart from that, Prototype 2 received the approval in the Verification session and no other major points were voiced in this round.

Overall, after the creation of prototypes 1 and 2 and two round of verification, the main principal differences of the standard C1 process compared to the new proposed model are summarized in Table 14.

C1 process	*Prototype 2*
Waterfall model: Specifications finished before implementation. All implementation done before verification is started	Agile model: Specification, implementation and verification done in 3 week iterations.
Basic principle: Functional driven, content remains the same, schedule and cost changes	Basic principle: Schedule and resource driven, schedule and cost remains the same, content changes
Plans and specifications ready at C1	Detailed planning and specification ready for each feature before a sprint
Requirements frozen at C1. Changes managed by Change requests	Requirements frozen at the beginning of each sprint
Software implemented at C2, ready for integration	Working integrated software package available after each sprint for demos
Formal project management	Close team co-operation
C2 and C3 milestone approvals	Two project progress checkpoints
Demo capability at C4	Early customer demo/involvement capability

Table 14. Principal differences between C1-process and the Prototype 2.

As seen in Table 14, the proposed model fulfils the fundamental principles of Agile product development. It is based on iterations and close co-operation among the project team members. The new model also creates functioning releases early in the process and makes it possible to perform customer demonstrations very early in the project.

Summing up, as a result of the development process described in Section 5, it was identified that Agile development model provides the needed flexibility and enhances the time to market of products. In addition, it was shown that customer satisfaction is enhanced with Agile development models and early customer involvement. Scrumban was selected as a base for the new model, and based on that, two prototypes (Prototype 1 and 2) were developed and verified with the case company experts. The major challenge was to meet the quality requirements and visibility requirements for the case company when using standard Agile models. Prototype 2 also provided suggestions for the tools to follow up on the quality requirements and gained a general approval from the final verification by the Product Development Board.

Table 15 compares Prototype 2 solution to the findings in Section 3 and Section 4.

Section 4, (Cusumano and Smith 1995)	Section 3	Section 5, Prototype 2 solution
1. Inadequate requirements statements.	Open requirements in the contract with customer, requirements were discussed for a very long time with the customer	-
2. Lack of specific and measurable goals.	C1-process does not support Agile development model	C1 process Gate criteria list mapped as items to the product backlog.
3. Architecture design flaws and changes.	Misunderstood requirements	Agile development, with late specification to learn and adapt
4. Inadequate change control systems.	C1-process does not support Agile development model	Breakable and Unbreakable content in the Product backlog
5. Inadequate project status reviews and reporting.	C1-process does not support Agile development model	C1 process gates reused (C-1, C0, C4 and C5) together with additional checkpoints.
6. Inadequate project metrics.	C1-process does not support Agile development model	C1 process Gate criteria list mapped as items to the product backlog.
7. Lack of open project communications.	Communication with several parties in different locations requires a lot of time and effort	Scrum and Scrumban events used for project communication
	Better communication between different projects about new features in each release.	Standard Project follow-up process used
	Shorter meetings, teleconferences with customer instead of face to face meetings.	-

Table 15. Product development challenges versus prototype 2.

As seen in Table 15, the findings in Section 3 can all be mapped to the list of project challenges listed by Cusumano and Smith (1995) and discussed in Section 4. The findings from Section 3 are collected from the *Lessons Learnt session* and *The Value Stream map* analysis. Some of the Lessons Learnt findings overlap with the issues found in the Value Stream map analysis. In these situations, the lessons learnt findings are used as the primary issues.

As demonstrated in Section 5, Prototype 2 addresses most of the challenges identified. Those challenges which are not managed by the Prototype 2 will be collected and proposed for further investigation in Section 7.

6 PROPOSED MODEL

This section presents the approved new model developed for the case company. The proposed model is based on Prototype 2 and the final adjustments after Prototype 2 verification.

The basis of the model is one of the agile development models, Scrumban, and the actual development process follows the Scrumban model very closely. The other parts of the product development model come from adapting the current C1 process used in the case company to the existing best practices found in Agile development, selected quite freely from different Agile product development models. In addition, the proposed model also follows the CMMI guidelines utilized in the case company, as the case company already has achieved certain level of maturity in CMMI levels, and regular CMMI audits are performed for all product development projects.

Figure 24 illustrates the proposed new model for product development.

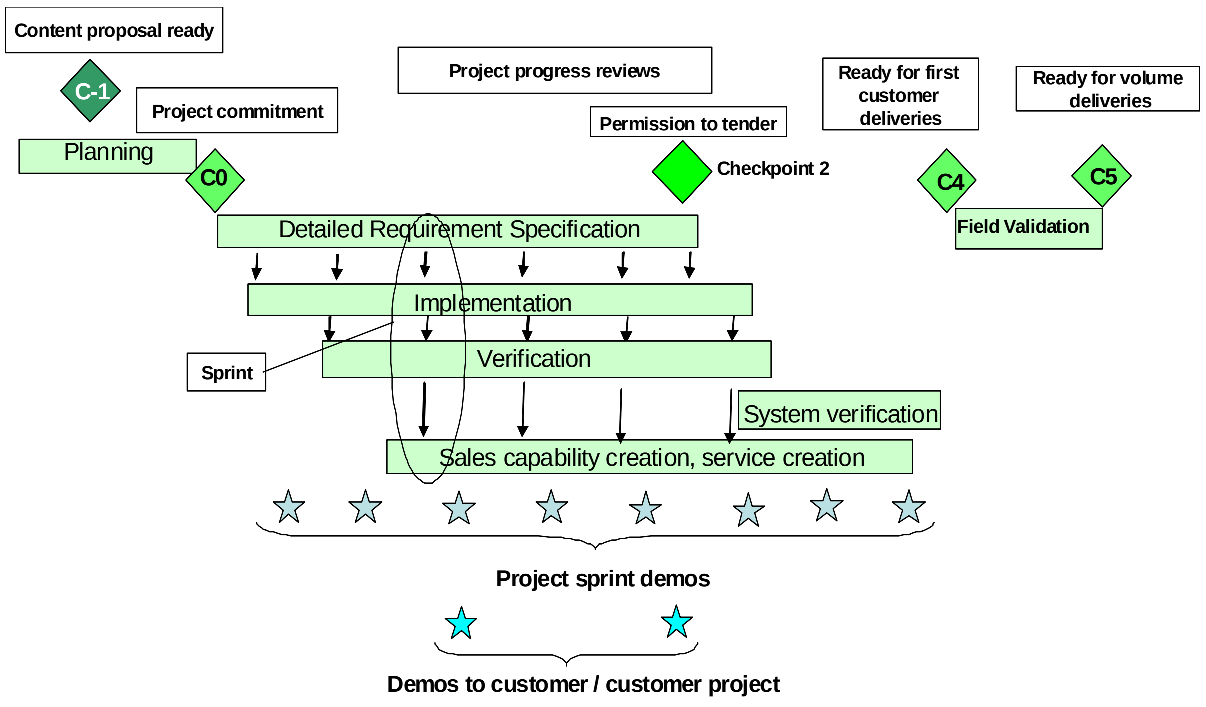


Figure 24. The proposed new model, presented in gates and phases.

As seen in Figure 24, the new model is build using iterations, known from Agile development, combined with the gates, known from the C1-process at the case company and from Waterfall-based models.

Table 16 presents the seven new features of the new product development model.

New feature	*Description*
Development in Sprints and incremental development	The new model is based on *Scrumban*, and all work items are done following the Scrumban framework in 3-week iterations called sprints. The sprints include the events *Sprint planning, Specification Review, Sprint Demo* and *Retrospective.*
Re-use of existing gates	*C1-process gates* are used as baseline for following up on project progress and quality. The gates applied are C-1, C0, C4 and C5.
Additional checkpoint	*Checkpoint* is used to fill up management visibility needs between Gates C0 and C4.
Product Backlog	*The Product Backlog* is used to follow up on all work items in the product development project. Product backlog includes necessary information for budget and schedule follow up. The product backlog is owned by the Product Owner (Product Business Manager in the case company)
Feature Board	*Feature Board* is used to follow up on the project progress, as a tool for the project manager. In addition, it is a tool to enhance management visibility to the project.
Customer Demonstrations	*Customer demonstrations* are added to the project schedule to be able to perform early customer demonstrations and to better meet customer expectations. Customer demonstrations also serve as an opportunity to impact the customer expectations.
C1-process gate criteria list mapping to the new model	*The Gate criteria list* from the C1-process is used to map all the deliverables expected from the project to the new model. The mapping of deliverables support the quality requirement follow up and comparability to other projects.

Table 16. The 7 features of the new product development model.

As seen in Table 16, the seven features of the proposed new product development model include *Development in sprints and incremental development, Re-use of existing gates, Additional Checkpoint, Product Backlog, Feature Board, Customer demonstrations,* and *C1-process Gate Criteria list mapping to the new model.*

One of the vital elements of the new model is the *Development in sprints and incremental development* which are utilized not just for software development but also for all the other tasks performed during the product development project. For example, the service creation including training courses, customer documentation, roll-out services, and engineering rules. In addition, the proposed model includes the creation of sales capability of the project outcome, i.e. the released product. The main benefit of iterative development is the possibility to adapt to changing needs and learn during the project. With the new model, the detailed specification work is done just before the implementation which makes it possible to take into account the learning from the development done so far, as well as the changes required by customer or market needs. Including all the work done during the project, the sprints help the project manager to keep track of progress. In addition, the project manager can better plan the different work packages when they also follow the sprint model. For example, if the implementation of a certain feature is postponed, it is easy to notice that also the testing of that feature has to be postponed. In addition, the project manager has, at the same time, to notice that the *Customer Documentation* work for that feature has to be postponed. Including the Customer documentation work, the sprints also make it possible to manage the changes, through the whole chain, from specification to service creation.

Another constituent of the *Development in sprints and incremental development* is the Retrospective event. It is a tool for continuous improvement, as at the end of each sprint the sprint is reviewed and the project team has the possibility to communicate *What went well?, What went bad?* and *What needs to be changed in the next sprint.* The retrospective forces the project team to focus on what can be improved instead of simply complaining, and the Retrospective therefore should improve the project quality. The results of each sprint are demonstrated in the Sprint demos, which ensure that all project members have the opportunity to present what they did during the sprint. It works as a knowledge sharing session inside the project, so that the testing team can see what was implemented etc. In addition, the sprint demo is an opportunity to present the progress to project team external and certain members from the management team are also invited to the sprint demos, which enhances the management visibility of the project.

Another improvement suggested in the proposal is *the Re-use of existing gates* in the new model, the formal gates to be used being Gates C-1, C0, C4 and C5. The quality metrics to be followed are suggested to be set for Gates C0, C4 and C5. The creation of quality metrics is mandatory to be able to follow up on project quality in product development and to be able to pass the audits required by the case company. The audits are mandatory to meet CMMI criteria. In the case company C1-process, as well as in the new model, there is a dedicated Quality Manager who is in charge of performing the audits and reporting the results of the audits. At each gate, the audit results are reviewed and, if quality targets are not met, the project is not allowed to pass the gate.

In the new model, *Additional Checkpoint* is used to enhance the management visibility of the project. This additional checkpoint was added as the gap between Gates C0 and C4 is long and, without the checkpoint, there would not be any formal management reviews from the actual development start gate to the ready-to-deliver gate

Another new feature, *Product Backlog,* collects all the work to be done during the project. It is used as a product management tool which also includes the work estimates and schedule for implementation. The product backlog content is divided to breakable and unbreakable content. The unbreakable content is mandatory content which can not be dropped from the new product release, with breakable content being the content that can be dropped, if needed. With the breakable content it is possible to fix the schedule and resources. If some feature implementation causes delays, breakable content is dropped and the schedule is not changed. Compared to the standard model where all content is fixed and the only option is to delay the schedule (or add resources to the project, with any of these options adding to the cost of the project), the product backlog allows easy recovering from a delay. The product backlog also puts the features to priority and complexity order, where the most complex and highest priority features are implemented first. In case of delays or a sudden cancellation of the project, the most important features gets implemented first and any delays caused by complex features can be identified early in the project.

The Feature Board is used to follow up on project progress, as a tool for the project manager and, additionally, as a tool to enhance management visibility to the project. The feature board is divided to columns of phases (for example, Specifications), the items from the product backlog being taken to the feature board. For each column, a limit is decided for how many items are allowed to be in that column; for example, 3 items in a specification column. This means that there are never more than 3 items in progress in the specification phase. In the sprint planning session, the specification team project manager decides which features to take to the specification column.

Naturally the features from the top of the product backlog should be selected. After the specification has put the feature in *the Done state*, the next phase, for example, implementation, can take the feature to its *In progress* column, if the limit for the maximum number of *In progress* items in that column is not exceeded. At the end of the sprint, in the retrospective, the whole team reflects if the limit for their column is right, i.e. whether the team capable of having more in progress items, or if is it already too much and the limit should be lower. The feature board provides an excellent communication tool inside the project, as well as a tool for the project manager to manage the project. The feature board also helps to enhance the quality of the project and supports the quality manager in the quality follow-up. In the new model, the feature board is also used as a tool to enhance management visibility. For this purpose, it is made available for the management team, and it is also formally presented at the checkpoint and Gate C4.

Another feature of the new model, *Customer demonstrations,* are added to the project schedule. As incremental development provides demonstration capability after each sprint, it is an excellent opportunity to create early product demonstrations to the customer. There are three big benefits of early customer demonstrations: 1) Development capability is proven to the customer, 2) It gives an opportunity for immediate customer feedback, and 3) Synchronise with the customer expectations. The first benefit, to prove development capability, is extremely useful when discussing new markets and with new customers. In big tenders, references are very often requested in the Request For Information/Proposal/Quotation (RFx) process. With the early demonstration capability, it is possible to make up for the lack of proper references. In addition, in ongoing projects, even after the contract is signed, early demonstration capability can help to prove that the supplier is capable of providing the tool and thus enhance the relationship between the customer and the supplier.

The second benefit of early demonstration capability, immediate customer feedback, helps to meet the customer requirements. As customer can early say if they are not happy with certain features or functions, a minor change early in the project can make the product much more suitable for the customer. A change that could be impossible when the product is finalized and released can be easily implemented in the earlier stages. The third benefit, Synchronise with the customer expectations, is a two-way opportunity. The developers get information about the customer expectations early; in addition, as the customer can see what they will get, the customer can easier provide the acceptance for the delivery. The customer already knows what they will get and is not expecting anything more than the functionality that they have seen in the demonstrations. In addition, it is easier for the customer to plan the acceptance tests and acceptance test procedures. A bonus benefit from early customer demonstrations is that the project team can use the feedback that the customer provides as evidence of good progress and customer satisfaction. This can be communicated to the management, thus enhancing the management visibility to the project.

Finally, *C1-process Gate Criteria list mapping to the new model*, is the last of the seven new features in the new model. Although the Gate criteria list from the C1-process is used, it is used not as in the normal C1-process. In the normal C1-process, the gate criteria are applied before each gate. Moreover, before Gates C-1 and C0 the baseline is created, i.e. which criteria are applicable for the project and what are the related deliverables. After Gate C0, the gate criteria list is reviewed by the project manager and the project team and all deliverables are checked. At each gate review, the results of the deliverables are reported. In addition, the gate criteria list is used in CMMI audits performed on projects.

To meet the CMMI quality criteria the Gate criteria list is also used in the new model. But instead of using it as a continuous tool during the whole project, it is mainly used before Gate C0. Before Gate C0, all project deliverables expected from the gate criteria list are reviewed, each criteria is marked as applicable or not applicable. The ones that are applicable are given a description of the actual deliverable related to the criteria, and description of when it is expected to be done. If the criteria deliverable is related to the actual work and includes a tangible deliverable, it is added to the product backlog. An example of a tangible deliverable is the data sheet of the product, which should be part of the marketing material gate criteria. When the deliverable is added to the product backlog, it is easy to manage by the project manager, and the project manager can make sure that the deliverable is delivered according to the plan. The gate criteria which are related to deliverables that are not tangible are not taken to the product backlog. Instead, a description of the deliverable with target date is added to the gate criteria list. An example of a non-tangible deliverable is the project tailoring decision. Such a decision is purely related to the process and is not worth to follow up on the product backlog. On the other hand, it is highly relevant especially for the pilot project as the use of the new model is the tailoring decision. The process related deliverables are only followed in the gate criteria list, and instead of the project manager, the quality manager is updating it. Moreover, it is only updated before the project audits, and as tangible deliverables are followed with the product backlog, there is a reference to the product backlog for those criteria, and for the process related deliverables the deliverable is explained directly in the Gate criteria list. With the Gate criteria list, the project quality can be assured to be at the same or higher level than for other projects.

Overall, the new model is improved to include 7 new features, *Development in sprints and incremental development, Re-use of existing gates, Additional Checkpoint, Product Backlog, Feature Board, Customer demonstrations* and *C1-process Gate Criteria list mapping to the new model.* With the seven new features, the new model utilises the flexibility of Agile development models and fulfils the fundaments of Agile development. It meets the targets set for the new product development model by the case company. First, it addresses the quality requirements of the case company including the CMMI requirements. Second, it enhances the management visibility of the project, provided with a set of new tools to follow the project progress and with transparency of the project progress from the open feature board and invitation to sprint demos to the management.

Finally, the new model meets the ideology of continuous improvement, as it provides the concrete first steps, for adapting the Waterfall based model used, towards an Agile model. It provides also a tool to improve the process during the project. The Product Decision Board made a decision to apply the mew model in a pilot case company project.

7 CONCLUSIONS AND RECOMMENDATIONS

This section summarizes the results presented in Sections 2-6 and suggests a number of recommendations for the management of the case company. Finally, it evaluates the results of the Thesis and considers reliability and validity of the research done. It also identifies the next steps and suggests possible directions for the future development.

7.1 Summary

This Thesis concentrates on improving the existing Waterfall-based product development model used in the case company to meet the demands of flexibility. The improvements are needed to provide more flexibility to product development while maintaining the quality targets and keeping the management visibility that are already in place in the case company.

The case company of this study is a product development company in global security solutions and systems integration which has utilized the product development processes within the established Waterfall model. The current model has satisfied the case company, especially in its needs for the required level of quality and management visibility. Presently, the case company has decided to switch to a more Agile product development model, and this study should address, in particular, needs for the management visibility and quality requirements, in addition to meeting Agile requirements, in the new proposed model.

The research approach applied in this study is action research. The model development is done in iterations, which are used for the development and verification of the new product development model in two action research cycles (for the creation of Prototypes 1 and 2). The data used for the development of the prototypes are collected in the interviews, discussions, a brainstorming session, and a Kaizen workshop (altogether, Workshops 1-4) in the case company.

The study starts with a current state analysis which involves: a) the description and analysis of the existing case company model (Standard product development model), b) the overview of the Capability Maturity Model Integration for Development and its levels applied in product development by the case company, c) a sample analysis of a development project currently on-going in the case company (Release 1 Project) in which the case company is using its standard development model, d) the analysis of the current product development process using the Values Stream Mapping technique, e) a description of the current quality requirements and project visibility requirements currently practiced in the case company, as well as the needs for quality and visibility requirements for the new model.

The results of the current state analysis are used for the focused search for best practices in literature and best practices review. This review includes the investigation of the product development models, especially concentrating on the Agile development and Waterfall-based models. Based on the findings from the current state analysis and theoretical search, model development is started. The model development became a process carried out in several workshops, as a team effort, with subject matter experts and key stakeholders closely involved in the model development and its validation. The model development includes the creation of two prototypes, Prototype 1 and 2 with their subsequent verification. The prototypes and verification of the prototypes led to the proposal of the final model to the case company.

Thus, the outcome of the Thesis is a proposal for a new product development model created for the case company and decided to be applied in a pilot project that started during the Thesis work. The proposed new model is based on Agile development principles and provides the tools to meet the targeted levels of quality and management visibility in the case company. In addition, this study suggests a set of managerial implications that can help to successfully implement the suggested product development model in practice.

7.2 Managerial Implications

The new product development model calls for taking a number of steps on the management sides to help put the proposed model into practice, so that the case company could benefit from this Thesis. The proposed steps can also address the issues that were not solved by the proposed model and further enhance some issues which were improved by the new proposed model. These steps include the following measures:

MI-1: To adopt the new model and use it in the pilot project.

MI-2: Broaden scope to all projects, perform the same analysis on the whole project portfolio

MI-3: Investigate the whole release cycle, starting from the release planning, the start of projects and other beginning stages.

MI-4: Investigate a possibility to move from the project-oriented mode to the release-oriented mode.

MI-5: Investigate the field validation process. It can be shortened when using Agile development.

MI-6: Investigate a possibility to co-locate projects as much as possible.

MI-7: Collect and study the results from the pilot project in which the new model will be applied, for example, the time to market interval, product quality, customer satisfaction, project team satisfaction.

MI-8: Share information about the new product development model, to avoid resistance at later gates of the pilot project.

The managerial implications are intended to serve three purposes: 1)*Further improve the new model, 2) Take the new model in to use and expand to other organisations,* and *3)Improve the quality and visibility requirements not visible in the listed project challenges.*

To stress this visibility, in Table 14, the managerial implications and the new features are mapped against the project challenges identified in Section 3 and Section 4.

Section 4, (Cusumano and Smith	Section 3	Section 6, Features of new model	Managerial Imlpication
1. Inadequate requirements statements.	Open requirements in the contract with customer, requirements were discussed for a very long time with the customer	-	MI-3
2. Lack of specific and measurable goals.	C1-process does not support Agile development model	C1-process Gate Criteria list mapping to the new model.	MI-2
3. Architecture design flaws and changes.	Misunderstood requirements	Development in sprints and incremental development	Mi-2
4. Inadequate change control systems.	C1-process does not support Agile development model	Development in sprints and incremental development, Proudct Backlog	MI-2
5. Inadequate project status reviews and reporting.	C1-process does not support Agile development model	Re-use of existing gates	MI-2
6. Inadequate project metrics.	C1-process does not support Agile development model	C1-process Gate Criteria list mapping to the new model.	MI-2
7. Lack of open project communications.	Communication with several parties in different locations requires a lot of time and effort	Development in sprints and incremental development, Proudct Backlog and Feature board	MI-6
	Better communication between different projects about new features in each release.		
	Shorter meetings, teleconferences with customer instead of face to face meetings.		
8. Lack of clear project milestones.	A new way to report project milestones when Agile software development is used	Re-use of existing gates	MI-2
9. Overly optimistic estimations of project feasibility.		Proudct Backlog and Feature board	MI-3 and MI-4
10. Various management difficulties.	C1 milestone process does not suit well to Agile development model, visibility to management about the project's progress and process not good enough	Additional checkpoint	MI-6 and MI-8

Table 17. Issues solved by the new model and enhanced with managerial implications.

As seen in Table 14, the new model with its features includes improvement proposals to most of the project challenges. The Managerial implications, which are connected to the improvements on the identified challenges, give further possibility to enhance the effects of the proposed model.

To summarize, the recommended seven steps on the management side could help put the proposed model into practice, so that the case company could benefit from this study. In addition, these steps provide opportunities for further improvement of the model and serve as a basis for *Continuous improvement.*

7.3 Evaluation of the Thesis

This Thesis proposed a new model using Agile software development methods as its basis combined with the vital elements of the processes already in place in the case company. One of the successes of the Thesis is that the new model was approved to be used in the pilot project, even though there has earlier been resistance against Agile development. A change of mindset towards Agile development in the case company could be seen as a particular achievement of this study.

The first milestone of this Thesis was the analysis and comparison of different Agile models to create a broad picture of product development, especially for software development projects. Comparison of the Agile models showed that the core fundamentals of them are not very different in different models, since most of them evolved from the same principles. An interesting result of the literature review was that most of the models have a long list of recommendations on how to work during the actual implementation and what kind of roles are needed in the projects. In this study, less focus was put on the parts and phases outside the implementation part, due to the fact that the Thesis was concentrated on the actual development stages.

An interesting result of the best practices search was that the managerial visibility important to this study was not highlighted in any of the models. This was also noticed in the material that considered the co-existence of CMMI and Agile, as the study particularly looked for the contradictions between CMMI and Agile, with the actual result being that they actually fulfil each other if used properly. The lack of focus on the managerial visibility in the studied models supported the reuse of practices from the C1 process.

Another unexpected discovery was that the challenge set at the beginning of the research of trying to use Agile in Waterfall was not actually a challenge, but rather an opportunity. With this we mean that an organisation with well-measured and managed processes already in place should re-use its existing practices and thus achieve higher quality in Agile product development.

On the other hand, it became clear from the results of the workshops that the use of Agile development models will bring more flexibility and adaptability to the product development projects. While, for the small to mid-sized software development projects, the standard Waterfall model created a lot of excessive work and poor adaptability, leading to a too longer time to market interval. The traditional Waterfall model was also spotted as creating uncertainty among the development team. These deficiencies in the current model used in the case company were identified from the comparison with Agile models.

The proposed new model was well received by the management and also the project team members. The management team approved the use of it in a pilot project and gave its mandate for the project team to adjust the process that is currently used in the case company. Although customer satisfaction and shorter time to market were not investigated in this study, lying outside the scope of this Thesis, the new model can be further studied for these matters when put to use in the pilot project.

To measure the results of the Thesis, it is also evaluated by two dimensions, validity and reliability of the conducted research.

7.4 Reliability and Validity in this Study

The internal validity of the Thesis is measured by checking whether the research question was answered. The research question for the Thesis was: *"How to meet the tar-*

geted levels of quality and management visibility while utilizing new development practices in an improved product development model?" Looking at the proposed model and summary of the Thesis, we can conclude that the research question was answered. The new model proposes to use an Agile model as a basic framework and it provides tools to ensure visibility of the project for the management. It also takes into account the quality assurance metrics to guarantee that quality targets are met.

The external validity is measured by addressing the question of "How transferrable the results are to all projects in the case company?" Since this study was taking other projects into account and the workshops always had a broader scope then just the targeted pilot project, this question is also considered to be answered positively. In addition, people from other projects participated to the development workshops. Therefore, it can be concluded that, to some extent, external validity was achieved. Additionally, one of the managerial implications listed in Section 7.2 was to perform a similar analysis on the other case company projects which would further enhance the external validity of this study. It was especially recommended that project of other nature, for example, hardware projects should be evaluated against the proposed model.

In this Thesis, the reliability concerns were managed by using a broad range of data sources based on the literature review and by evaluating several similar Agile product development models to get a broader perspective on Agile product development. For the literature review, a selection of reliable article from leading academic journals was made at two different time points. To complement the scientific studies, the data for the actual Agile software development models were mainly collected from articles and books on the application of these models.

In the data collection and analysis phase, the participants of the workshops were changed, so that different perspectives and viewpoints were gathered from a wide range of project team members. The model creation was also done in iterations in order to create the new model gradually, at different time points, and strive for consistent improvements. Even though the model was iterated several times, almost the same result was achieved each time. To improve the reliability further, it could be recommended to perform a similar model creation with people less familiar with the current C1-process in the case company. It might provide new knowledge on the current model limitations and add to the creativity and the innovativeness to develop the new model further.

On a more general level, this Thesis has shown that Waterfall based product development models have strengths which are not covered by the Agile models. On the other hand, Agile models have clear advantages compared to the traditional Waterfall models in terms of flexibility and ability to adapt to changes in the environment. This Thesis has shown that it is possible to integrate the strengths of both types of models to create a product development model which would include features from both.

REFERENCES

Abrahamsson, P.; Salo, O.; Ronkainen, J; Warsta, J (2002). Agile Software Development Methods: Review and Analysis. *VTT Electronics*, Espoo: VTT Publications 478.

Ambler, Scott .W (2005). A Manager's Introduction to the Rational Unified Process (RUP). © Ambysoft 2005 http://www.ambysoft.com (Accessed 3 July 2012).

Anderson, D.; Dalton, J.; Glazer, H.; Konrad, M., and Shrum, S. (2008). CMMI® or Agile: Why Not Embrace Both! *Software Engineering Process Management*. Technical Note CMU/SEI-2008-TN-003

Ballé, Freddy; Ballé, Michael (2005). Lean Development. *Business Strategy Review.* Autumn 2005, 17-22.

Beck, Kent (2000). *Extreme Programming Explained: Embrace Change.* 1st ed. Addison-Wesley: Pearson Education.

Beck, K.; Beedle, M.; van Bennekum, A.; Cockburn, A.; Cunningham, W.; Fowler, M.; Grenning, J; Highsmith, J; Hunt, A; Jeffries, R; Kern, J; Marick, B; Martin, R.C.; Mellor, S; Schwaber, K; Sutherland, J; Thomas, D (2001). *Agile Manifesto* www.Agilemanifesto.org (Accessed 7 July 2012).

Benner, M. J. and Tushman, M. L. (2003). Exploitation, Exploration, and Process Management: The Productivity Dilemma Revisited. *Academy of Management Review.* Vol.28 (2), 238-256.

Boehm, B. and Turner, R. (2005). Management Challenges to Implementing Agile Processes in Traditional Development organizations. *IEEE Software.* Vol.23 (5), 30-39.

Case Company Business Process Management Suite (2012). C1 – Product and Solution Management. *Case Company-BMS-0372 Process Handbook,* Version 4.1. [Internal document Q1 2012].

Cockburn, A. (2006). *Agile Software Development: The Cooperative Game.* 2nd ed. Addison-Wesley: Pearson Education.

Coghlan, D. and Brannick, T. (2010). *Doing Action Research in Your Own Organization.* 3rd ed. Sage Publications.

Cooper, R.. G.; Edgett, S. J. and Kleinshmidt, E. J. (2002a). Optimizing the Stage-Gate Process: What Best-Practice Companies Do – I. *Research-Technology Management.* Vol.45 (5), 21-27.

Cooper, R.. G.; Edgett, S. J. and Kleinshmidt, E. J. (2002b.) Optimizing the Stage-Gate Process: What Best-Practice Companies Do – II. *Research-Technology Management.* Vol.45 (5), 43-49.

Chrissis, M. B.; Konrad, M., and Shrum, S. (2011). *CMMI® for Development: Guidelines for Process Integration and Product Improvement* .3rd ed. Addison-Wesley: Pearson Education.

Cusumano, M. and Smith, S. (1995). *Beyond the Waterfall: Software Development at Microsoft.* MIT Sloan School of Management. Working paper (3844-95). August 1995

Dooley, D. (1995). *Social Research Methods*. Englewood Cliffs, NJ: Prentice-Hall, Inc.

Ferrance, E. (2000). *Action Research*. Northeast and Islands Regional Educational Laboratory at Brown University http://www.alliance.brown.edu/db/ea_catalog.php (Accessed 11 August 2012

Imai, Masaaki (2012) *Gemba Kaizen: A Commonsense Approach to a Continuous Improvement Strategy.* 2nd ed. The Kaizen Institute ltd. McGraw-Hill Companies Inc.

Kniberg, Henrik (2009) *Kanban vs Scrum: How to Make the Most Out of Both.* Version 1.1 June 2009, Crisp AB

Koch, A. S. (2005). *CMM® and CMMI®: Show Me the Value.* ASK Process Inc. www.askprocess.com (Accessed 25 July, 2012)

Kulatilaka, Nalin; Perotti, Enrico C. (1997) *Time-To-Market Capability as a Stackelberg Growth Option.* School of Management, Boston University and University of Amsterdam and CEPR. May 1997

Ladas, C. (2008). Scrum-ban. *Lean Software Engineering: Essays on the Continuous Delivery of High Quality Information Systems.* July 2008 www.leansoftwareengineering.com (Accessed 24 July 2012)

Liebermann, M. B. (2007) *Did First-Mover Advantage Survive the Dot-Com Crash?* Anderson Graduate School of Management. http://www.anderson.ucla.edu/faculty/marvin.lieberman/papers.html (Accessed 18 August 2012)

McMahon, P. E. (2010) *Integrating CMMI® and Agile Development: Case Studies and Proven Techniques for Faster Performance Improvement.* 1st ed. Addison-Wesley: Pearson Education Inc.

Quinton, S. and Smallbone, T. (2006). *Postgraduate Research in Business: A Critical Guide* Sage Publication ltd. https://www.dawsonera.com/guard/protected/dawson.jsp?name=https://idp.metropolia.fi/idp&dest=http://www.dawsonera.com/depp/reader/protected/external/AbstractView/S9781847878038 (Accessed 1-24 July 2012)

Schwaber, K. and Sutherland, J. (2011). *The Scrum Guide, The definitive Guide to Scrum: The rules of the Game.* www.scrum.org (Accessed 12 July 2012)

Sethi, R. and Iqbal, Z. (2008). Stage-Gate Controls, Learning Failure and Adverse Effect on Novel New Products. *Journal of Marketing* Vol.72 (1), 118-134

Smith, P. G. (1999). From Experience: Reaping Benefit from Speed to Market. *Journal of Product Innovation Management* Vol.16 (3), 222-230

Suomala, P. (2004*). Measurement of New Product Development Performance – Life Cycle perspective*. Tampere University of Technology. Tampere TTY-Paino

Ulrich, K. T. and Eppinger, S. D. (2004). *Product Design and Development.* 3rd ed*.* McGraw-Hill/Irwin

Vancayzeele, D. (2010). Case Company-BMS-0004-PO 1 7 M5 BMS General Process Policy. *Case Company BMS General Policy* [Internal document]

Appendix 1.

Interviews and Discussion

Interviews were conducted personally, in face-to-face meetings, in open discussion sessions, in the different workshops, and separately appointed with the experts. The following questionnaire was used as a catalyst for discussion both in the interviews and workshops. Discussions were documented in the case company minutes and used for the prototype model creation.

Q1: Why is a change needed?

Q2: What expectations do you have for the new model?

Q3: How do we evaluate the success of the new model?

Q4: What are the most important visibility requirements for the new model?

Q5: What are the most important quality requirements for the new model?

Q6: What are the biggest challenges that we meet in projects?

Q7: What are the biggest challenges for you in projects?

Q8: What are the characteristics of a successful project?

Appendix 2.

Lessons Learnt session

In the lessons learnt session all the work package managers of the Release 1 Project were present. The participants had the opportunity to comment on a wide range of topics including, for example, the following points.

Name the main good practices during the project:

Name the main general challenges during the project:

Name the main good practices in working with the customer:

Name the main challenges in working with the customer:

Name the main good practices in working with the subcontractor:

Name the main challenges in working with the subcontractor:

Name the main good practices for internal project work:

Name the main challenges for internal project work:

Name the main issues related to Quality Requirements:

Name the main issues related to Configuration Management:

Name the main issues related to Risk Management:

Appendix 3.

Workshop 3 (Kaizen Workshop)

Workshop 3 was conducted as a Kaizen workshop, with external Kaizen mentors invited to facilitate the workshop.

Kaizen Workshop

Workshop (not an assessment) focused on synchronizing people, finding a common goal, uncovering problems behind problems, identifying long term solutions (ideal state) as well as very next small (Kaizen) steps

Key points

Different roles involved (Management, S&M, Development, Testing)
Common vision and goal identified and agreed
Value Stream map created and issues and their root causes visualized
Short and long term solutions identified

Duration

Workshop 2 days
Start-up and preparatory meeting 2 hours
Wrap-up and retrospective 1 hour

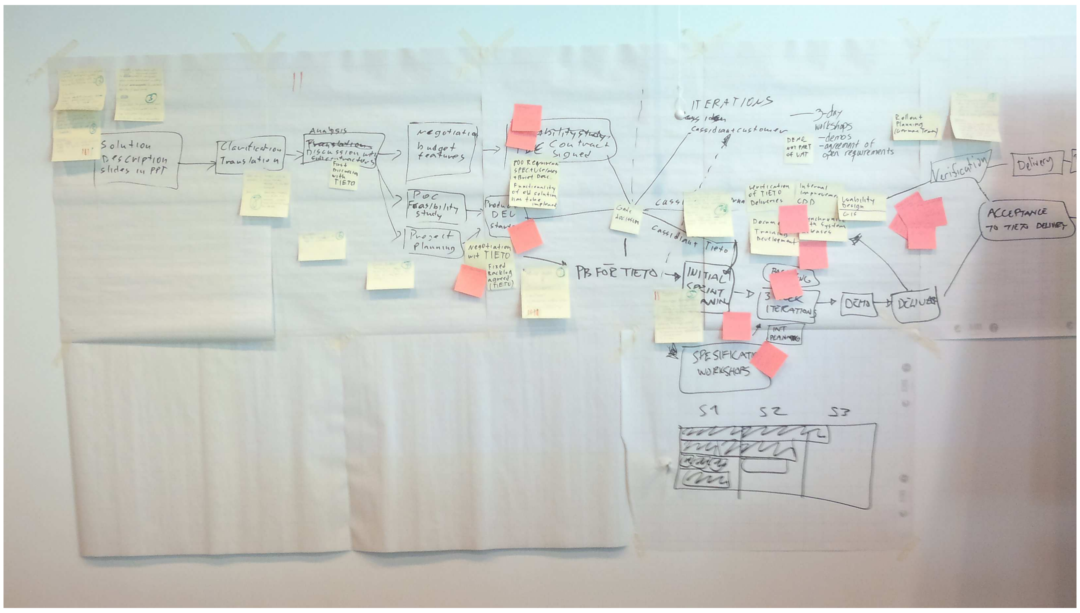
Fundamental principles behind

Team is better than one expert
Don’t just plan… act!
It’s better to get a 50% improvement now than to wait months hoping for perfection

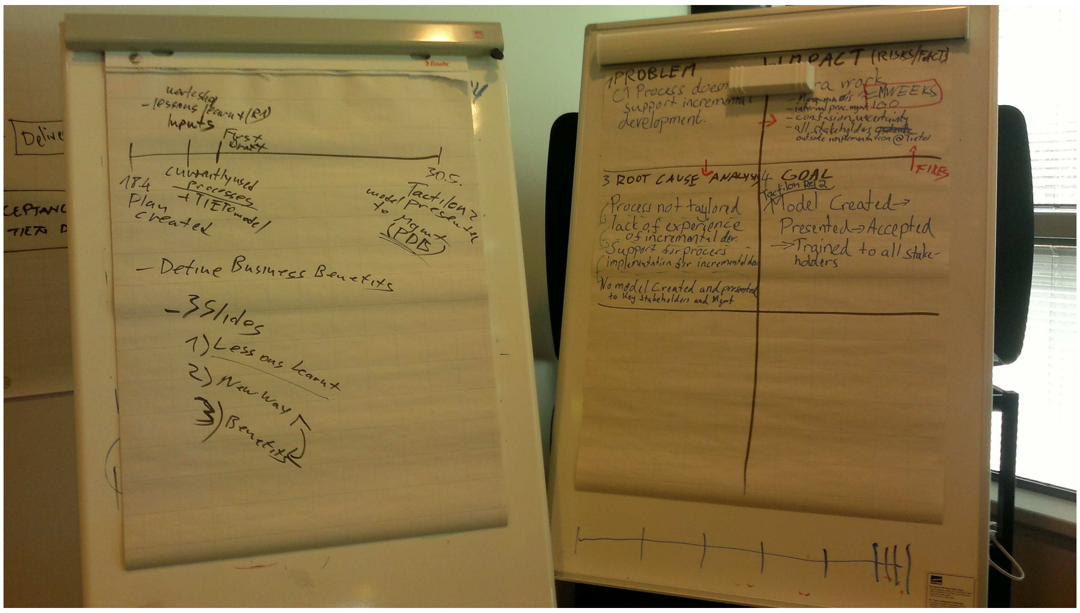
Outcome from workshop

Value stream map of Release 1 project – Issues and pain points highlighted
Root cause analysis – of main issues and pain points
Solution Brainstorming – short and long term solution proposals for improvement

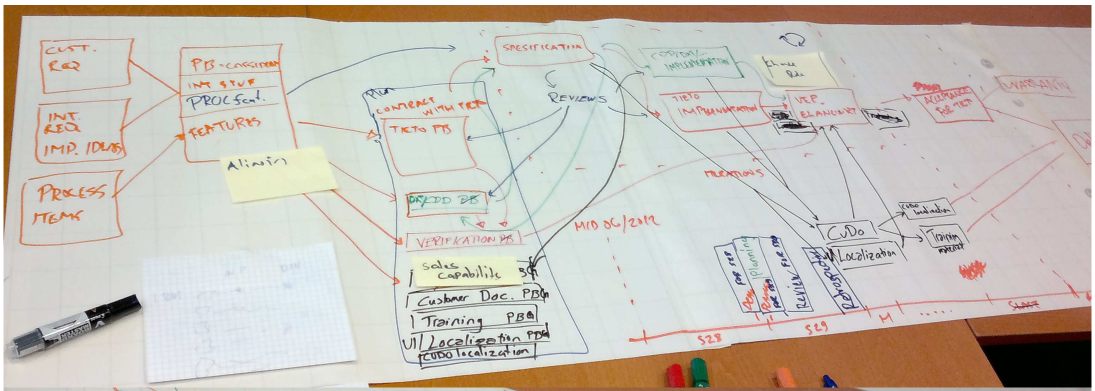
Value Stream map with pain points pinpointed:

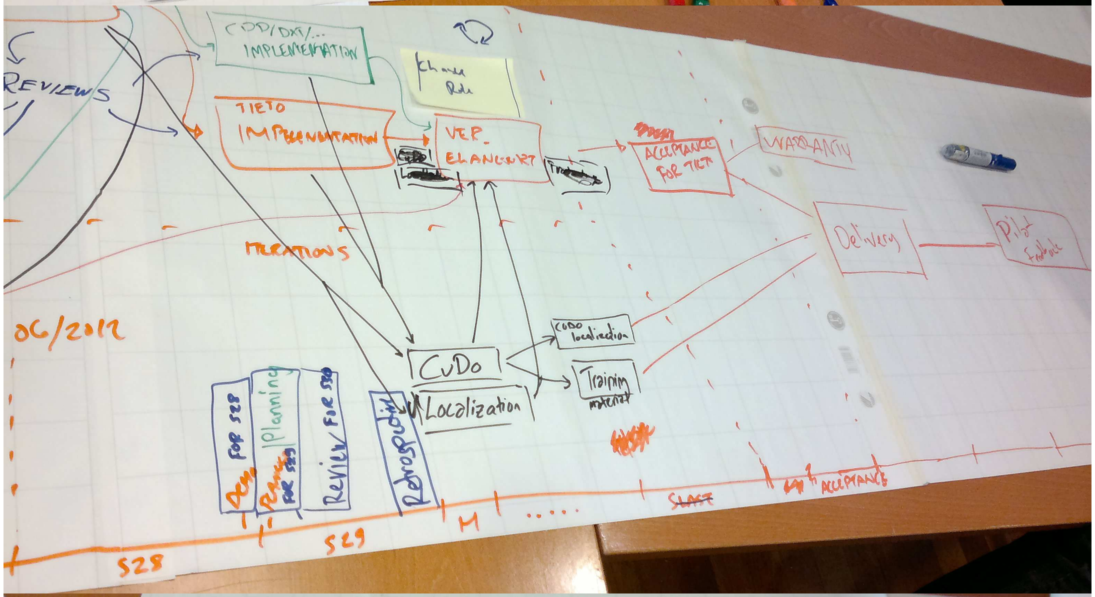


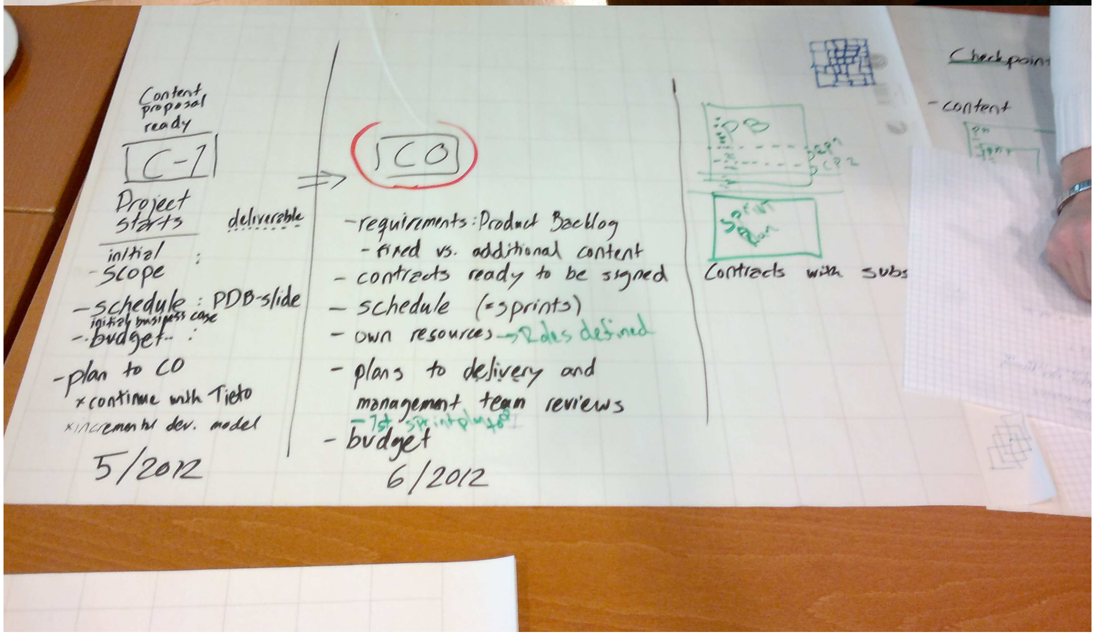
Root Cause analysis. The issue discussed: "C1 Process not suitable for Agile development:


workshop
- lessons learnt (R1)
inputs
First Draft
18.4
Plan created
currently used processes + TIETO model
30.5.
Tact ilon 2 model presented to Mgmt (PDB)
- Define Business Benefits
- 3 Slides
1) Lessons learnt
2) New way
3) Benefits
1 PROBLEM
C1 Process doesn support incremental development.
IMPACT (RISKS/FACTS)
- internal proc. mgmt 100
- confusion, uncertainty
- all stakeholders outside implementation @ Tieto
MWEEKS
FIRES
3 ROOT CAUSE ANALYSIS
Process not taylored
lack of experience of incremental dev.
Support for process implementation for incremental dev
No model Created and presented to Key stakeholders and Mgmt
4 GOAL
Tact ilon Rel 2
Model Created →
Presented → Accepted
→ Trained to all stake-holders

Solution Brainstorming. The issue discussed: "C1 Process not suitable for Agile development:







Appendix 4.

C1 process: Gate criteria mapping

During the prototype creation and development of the proposed model, all the C1 process gate criteria were mapped to build the new model. As an example, this appendix presents an excerption of the implemented mappings (for Gates C-1 and C2).

C-1 GATE

Project	Pilot project for new model
Gate /	C-1 / May

C Gate	Gate Criteria	Functional Process Deliverable	Explanation for criteria	Core Team Member	Valid in project yes/no	Pilot Gate
C-1	Needs and Initial requirements identified including service requirements	F1 Stakeholder needs	The needs and performance are identified, some service requirements may also be identified: e.g. requirements for SW downloading and SW installation time, service break during upgrade etc.	Product Manager	YES	C-1
C-1	Initial schedule and effort estimates until C5		C Gates times and work load	Product manager	YES	C-1
C-1	Plans and resources for C-1 - C0	Planning team in place	Feasibility activities plan Needed plans and named resources for next phase exist. Acquire & Mobilize the Planning team	Product manager	YES	C-1
C-1	Core team members to reach C0 nominated	Planning team in place	Typical core roles until C0: project manager, product manager, architect, delivery capability manager, controller	Product manager	YES	C-1
C-1	Initial business case / financials	F5 Financial analysis business plan / business case	F5 valid if there Commercial Contract under preparation. Use LoB Business case / Financial template	Product Manager	YES	C-1
C-1	Product and solution content in line with roadmap and business and product strategy	M4 R&T/D Portfolio Dashboard	Project content checked against roadmap	Product Manager	YES	C-1
C-1	Feedback from previous product/solution projects (C1) taken into account	Lesson learnt	Lessons learnt from previous projects, products and solutions taken into account	Project Manager	YES	C-1
C-1	Features or functions which require interoperability with other products (e.g. end user equipment) identified.			System Architect	YES	C-1
C-1	Preliminary strategic suppliers list available		Supplier candidates short listed and related risks recognized. Procurement must be involved from the very beginning of the screening. Delivery Capability study under work	Delivery Capability Manager	YES	C-1
C-1	Feasibility study for Manufacturing & Repair Process		Delivery Capability (NPI) Plan under work	Delivery Capability Manager	NO	NA
C-1	Product cost target		Cost target reviewed with Delivery Capability Manager (NPI project)	Product manager	YES	C-1
C-1	Sourcing cost draft		Sourcing cost follow up table from Delivery Capability Manager prepared and reviewed in Core team.	Delivery Capability Manager	YES	C-1
C-1	Product portfolio consistency verified (phase-outs of other products)		If new product will replace existing product initiate C6 study	Product manager	YES	C-1

C2 GATE

C2	Integration test plans approved and test cases reviewed	F1 Integration plan		IVV Manager	YES	In sprints
C2	Detailed SW specification, SW coding done, module/unit level testing completed	F2 Verified Software Solution	SW solution fully (source, information system, builds) developed, and unit tests completed.	Development Project manager	YES	In sprints
C2	SW integration completed	F2 Integrated Software Solution	Integration test implementation available, test cases passed, SW version description available	Development Project manager	YES	In sprints
C2	SW verification completed	F2 Software Verification Plan and F2 Software Verification & validation report	SW Verification plan approved. SW verification test cases passed. Build instruction and SW version description available.	Development Project manager	YES	In sprints
C2	HW prototypes available and satisfying C2 criteria defined in C1.5	F3 Packaged HW Prototypes		Development Project manager	NO	NA
C2	Configuration Management Plans updated	F4 Configuration Management Plan, F4 CMP status reports, F4 Configuration Audit Report		Configuration Manager	NO	NA
C2	Product configurations defined		Product configuration and structure for commercialization defined. Commercial materials structure exists	Product Manager	YES	CP 2
C2	Final supplier selection done		Delivery Capability Plan updated	Delivery Capability Manager	YES	C0
C2	Maintenance plan draft	Maintenance plan	Maintenance elementary items (spare parts) defined and approved - preliminary list of spare parts available, list of repair tools available	Service capability manager	NO	NA
C2	Production testing specification defined		Development makes specification to operations how to test the product in production. This document is input for production test planning	Delivery Capability Manager	NO	NA
C2	Product cost LE for C2 reviewed and agreed in Core team		Corrective actions for product cost agreed if needed and NPI plan updated	Delivery Capability Manager	YES	CP 2
C2	Implementation manufacturing process to production			Delivery Capability Manager	NO	NA
C2	IPR activities checked against IPR plan			System Architect	YES	CP 2
C2	Open Source Components adaptation as planned in Usage Plan and documented as commercial use requires	F2 Software Version Description	Ensuring that 1) linking to own code does not contaminate whole developed software 2) Open Source licence conditions are fullfilled, and 3) Open Source licence information to customer documentation created	Development Project manager	YES	In sprints
C2	Deliverable list reviewed and status update approved by core team		See last C0 criteria	Project Manager	NO	NA

Appendix 5.

PRODUCT BACKLOG FOR THE NEW MODEL

ID	Category	User story	Acceptance criteria	Responsible	Schedule	N.N.- Specification		N.N. - Sub. - implementation		N.N. - Internal Implementation		N.N. - I&V		N.N - CuDo		N.N - Services		N.N - Quality	
						Schedule	Status	Schedule	Status	Schedule	Status	Schedule	Status	Schedule	Status	Training	Localisation				
1																					
2																					
3																					