Productization:
The process of transforming from customer-specific software development to product software development

Peter Artz
Inge van de Weerd
Sjaak Brinkkemper

Technical Report UU-CS-2010-003
January 2010

Department of Information and Computing Sciences
Utrecht University, Utrecht, The Netherlands
www.cs.uu.nl

ISSN: 0924-3275

Department of Information and Computing Sciences
Utrecht University
P.O. Box 80.089
3508 TB Utrecht
The Netherlands

Table of contents

Abstract

Developing a software product is getting increasing attention in the scientific field and societal field. Organizations are recognizing the potential benefits and importance of developing a product for a market. Product software is defined as a packaged configuration which consists of software components or a software-based service, with auxiliary materials, which is released for and traded in a specific market (Xu & Brinkkemper, 2005). Also organizations which develop software specifically for one customer have identified a need to change their software into a standard product. But how can these organizations transform their software and create a standard software product? Yet, there are few scientific studies reported on such transformation. This study stipulates the need for more exploratory research on the transformation by identifying a productization process.

This research presents several concepts and models, which are developed and validated within this research. The first result which we present in this study is an overview of twenty differences between customized software development and standardized development. The second result is the productization process, which we created as a result of the identified differences and a literature study. The productization process describes the transformation from a customer-specific software development to a standard software product for an entire market. Also a complete graphical and textual description of the entire process is provided, in which the characteristics of each stage are described. The third result of this research is an approach to actually apply the productization process within an organization. The productization approach is a method which consists of three steps in order to define a custom advice how an organization should continue to become a software product business. Finally, the last result is a list with guideline for the implementation of the product management functions from a reference framework for Software Product Management (Weerd et al., 2006a).

Keywords: Productization, software product management, customer-specific software, customizable software product, standard software product, transformation.

1 Introduction

A product manager is a relatively new function within a product software company; this is the result of a transformation of focusing on customized software to developing software as a standard product (Weerd et al., 2006a). Within a small and midsized company a product manager is highly involved in managing the product strategy and the overall delivery process (Dver, 2003). Especially within small product software companies Software Product Management (SPM) is essential. Usually these companies base their whole business on one or two products (Kilpi, 1997). One of the first signs of a failure of product management is insufficient requirements engineering. The main reasons for this failure are the difficulties with misinterpreted needs, changing and creeping requirements, missed deadlines and budgetary commitments and, more globally speaking, failing business opportunities (Ebert, 2007).The result of the study performed by Ebert (2007) showed that the strengthening of a coherent product management role improved the time to market, schedule adherence and handover quality. Ebert also defined guidelines towards successful product management:

- Business objectives and accountability
- Mastering requirements
- Managing risks and uncertainty
- Leadership and teamwork

In order to be able to improve the product management processes, a specific approach is required. The best known maturity models are Capability Maturity Model (CMM) (Paulk et al., 1993), and its follow-up CMMI for implementing the capability maturity model (CMMI Product Team, 2002). However, a number of organizations find it too heavy and difficult to use CMMI (Nawrocki, 2002). In addition, Staples et al. (2007) found several other reasons why CMMI is not adopted: the organization was too small, the services were too costly, and the organization had no time to implement the process improvements. In addition, Brinkkemper et al. (2008) also elaborated on the fact that these methods are too superficial for the specific nature of product software companies. They introduced the Product Software Knowledge Infrastructure (PSKI), which helps product software companies by obtaining a custom-made advice for improving development processes (Weerd et al., 2006b). This infrastructure consists of an online systematic collection of methodical knowledge which can be used for improving the maturity of specific processes within an organization.

A model that describes the product management processes is the Software Product Management (SPM) reference framework, as illustrated in Figure 1. This framework is developed by Weerd et al. (2006a) and is specially designed to give product managers and organizations more insight information about the product management processes and the related stakeholders. The reference framework is a model in which key process areas, stakeholders and their relations are modeled. The four main product management functions are: 1) ‘Portfolio management’; 2) ‘Product roadmapping’; 3) ‘Requirements management’; and 4) ‘Release planning’. Additionally, the internal stakeholders (company board, research & innovation, development, support, services, and sales & marketing) and external stakeholders (market, partner companies, and customers) are also included. In this research we adopt the SPM reference framework in order to support and develop the transformation process to change from a customer-driven software development to a standard software product. We also adopted the SPM Maturity Matrix in order to determine the current maturity levels of specific processes (Weerd et al., 2009).

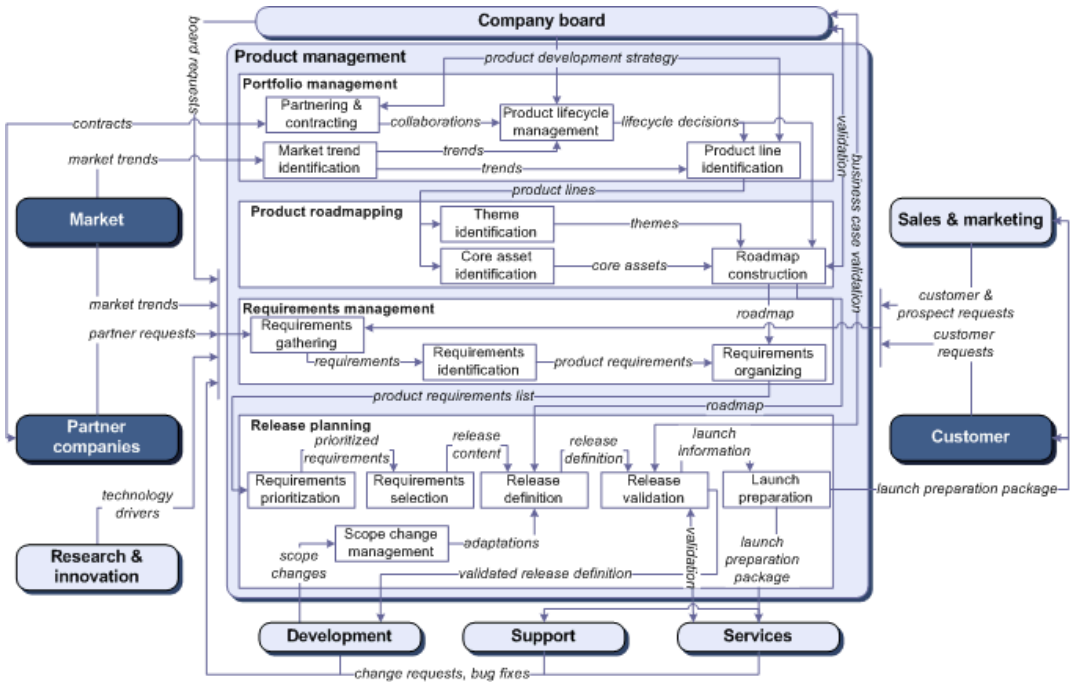


Figure 1: Software Product Management reference framework

1.1 Problem statement

More and more customer-driven organizations recognize a need to create a standard software product. However, main question for such organizations is how they should change their organizational processes in order to create a standard product.

Consequently, the main research question of this research project is: *How can the Software Product Management reference framework support the transformation from developing customized software to a standard software product?*

1.2 Terminology

In conducting this research, we use the reference framework for Software Product Management. For that reason, we first elaborate on the definition of software product management we used during this research. Software product management is: “the process of managing software that is built and implemented as a product, taking into account lifecycle considerations and generally with a wide audience. It is the discipline and business process which governs a product from its inception to the market or customer delivery and service in order to generate biggest possible value to the business” (Ebert, 2009).

Additionally, the definition we use for a software product is: “defined as a packaged configuration of software components or a software-based service, with auxiliary materials, which is released for and traded in a specific market” (Xu & Brinkkemper, 2005).

We identified the definition of the productization process to this approach. The exact definition we define for this process is: *the process of transforming from customer specific software development to a standard software product*. The terminology productization is not widely used in the literature. Hoch et al. (1999) identified it as: “Productization means standardization of the elements in the offering”. In addition, Hietala et al. (2004) also stated that the term productization includes several technological elements from the very early stages of designing a product to the commercial elements of selling and distributing the product. In our terminology, we focused especially on the strategic and operational changes in managing the software and not selling and distributing it.

Initially, we planned to apply the differentiation between 'Project' and a 'Product' within this research. However, a 'Project' can be understood in multiple ways, firstly a project for creating product software, and secondly a project in order to create a customer-based information system. This same issue emerges when we look at the terminology 'Product'; a product can be the result of a customer-driven project. On the other hand, it can also be related to a product designed for a specific market. In order to prevent this misunderstanding, we use the notation of 'customized software development' or 'customer driven' and 'standardized software development' or 'market driven'. In addition, we also provide a list with the used abbreviations on page 85.

1.3 Structure

The structure of this study is divided into three main parts. Within the first part we elaborate on the used research approach and related literature regarding software product management. This is extended by a section which describes the differences between developing customized software (customer-driven) and developing standardized (market-driven) software. Finally, this part also elaborates on the available literature related to the transformation from developing software for a specific customer to a standard software product.

The second part consists of the development and the validation of the main artifacts of this research. Three main artifacts are developed:

1) The entire productization process which consists of seven stages and the description stage;
2) An approach to apply the productization process within an organization; and
3) The guidelines for the implementation of the SPM reference framework. Additionally, the second part also consists of the validation of both artifacts. The validation of the application of the productization can be found in the third part of this study.

The third part of this study is a business cases at MP Objects which describes the adoption of the productization process. The business case consists of three major processes:

1) The identification of the initial position;
2) A gap analysis is carried out which results into several recommendations;
3) The last process is that MP Objects decides which recommendations they plan to execute in order to become fully software product management oriented.

Finally, the last part we present our conclusions of this research project. We also discuss on the results of this research approach and we determined specific topics for future research.

PART I: Theoretical background

2 Research approach

The used the research questions, research method, the research contribution, the validity threats and the related literature are presented within this section.

2.1 Research questions

An increasing number of organizations which develop customer-specific software have a desire to transform, and start developing a standard software product. Weerd et al. (2006a) presented a reference framework for Software Product Management (SPM) which support an organization in managing their product(s) and in which the key process areas, stakeholders and their relations are modeled. As a result, the main research question of this research is:

> *How can the Software Product Management reference framework support the transformation from developing customized software to a standard software product?*

In order to answer our main research question, we defined the following sub-questions which are answered within the upcoming sections of this research project:

1) *What are the specific differences of developing software for a specific customer compared with developing software for a market?*
2) *What kind of characteristics describe the stages of the transformation?*
3) *What are major guidelines that are important for the implementation of the SPM reference framework during the transformation?*
4) *How can an iterative and incremental implementation and the maturity matrix support the adoption of the SPM reference framework?*
5) *To what extent is the implementation of SPM useful and interesting for a small sized organization?*

2.2 Research method

In conducting this research, we apply the design-science research guidelines identified by Hevner et al. (2004). The design-science guidelines "seek to extend the boundaries of human and organizational capabilities by developing new innovative artifacts". A drawback of these guidelines is that no logical hierarchy for applying the guidelines is given; therefore we combined the graphical notation of Vaishnavi & Kuechler (2007) with the guidelines from Hevner et al. (2004).

2.2.1 Design-science

The design-science research method consists of the analysis of the use and performance of designed artifacts to improve on the behavior of aspects of Information Systems (Vaishnavi & Kuechler, 2007). Hevner's objective is to describe the performance of design-science research in information systems via a conceptual framework and clear guidelines for understanding, executing, and evaluating the research. We applied these guidelines in order to describe the problem solving process of this research:

Problem Relevance

Guideline: "The objective of design-science research is to develop technology-based solutions to important and relevant business problems".

The first guideline consists of all initial activities such as the determination of the research question / sub-questions, clarification of the used research methods, and the used approach for maintaining the validity. The problem within this research is related to the transformation from a customer-specific software development to a standard software product. At this moment there is barely any literature available on how companies should transform and which processes should change or how they

should change in order to create product software. Within this research we adopted the Software Product Management reference framework (Weerd et al., 2006a) in order to create product software and become market driven. Additionally, this creates a second difficulty, namely how to implement the processes and product management functions from the reference framework. Currently, no concrete guidelines or references are available for the implementation of the framework.

Research Contributions

Guideline: "Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies".

This research contributes scientifically and socially on several areas. The process of 'How' a company should transform and which actions should be carried out is not yet widely studied in the literature. Furthermore, this research is also another validation of the reference framework for SPM, the SPM maturity matrix, and the situational factors. Finally, the guidelines for the implementation are also part of the scientific contributes. On the other hand, the main social contributions of this study are the recommendations for MP Objects, so that they can continue with their productization process in order to become a software product business. More information about the scientifically and socially contribution can be found in section 2.3.

Design as a Search Process

Guideline: "The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment".

Before we are able to determine the transformation from customized software development to a standard software product, we first studied the differences between these two approaches. By applying this approach we were aware of the characteristics for customer-driven software development and market-driven software development before we started with defining the entire process. Additionally, for the generation of the implementation guidelines we tried to combine several sources of evidence so that reliable guidelines were developed.

Design as an Artifact

Guideline: "Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation".

Focus of this guideline is primarily on the identification and the development of the productization process, a method in order to apply this process, and the guidelines for the implementation of SPM processes within an organization. The design of the productization process is based on a literature study and some exploratory interviews. The application approach is essentially a combination of a number of available methods, such as the maturity matrix (Weerd et al., 2009), situational factors (Bekkers et al., 2008a), and process deliverable diagram (Weerd & Brinkkemper, 2007). The implementation guidelines for the SPM framework are the result of a broad literature study on the different product management areas.

Research Rigor

Guideline: "Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact".

For the development of the artifacts we used several different methods to gain the required information. Main information sources for the generation of the productization process were some exploratory interviews and an extensive literature study. The most important method we used from the literature is the reference framework for Software Product Management (Weerd et al., 2006a). The development of the guidelines for the SPM framework is based on a literature study performed for each of the product management functions.

For validating the correctness and completeness of the artifacts, several rigorous evaluation methods can be selected (Hevner et al., 2004). The methods we used for the validation of the

artifacts are expert interviews, a survey, and a business case. We used an expert panel, because the interviewed experts provide an informed, objective, and unbiased opinion as well as suggestions about the developed artifact (Sandelowski, 1999). Secondly, we created a survey in order to retrieve the opinions of several product managers which participated on a SPM course. The survey was designed in such a way, that we were able to examine the structure of artifact for static qualities (e.g. complexity or readability) (Hevner et al., 2004). Finally, the last method we used is the business case study. By performing a case study, the validation process provides an in depth evaluation in a business environment (Zelkowitz & Wallace, 1998).

Methods applied while carrying out the business case are the maturity matrix for determining the maturity of the (present) SPM processes (Weerd et al., 2009), situational factors for the determination of the best suitable maturity levels (Bekkers et al., 2008a), and the process deliverable diagram for the visualization of the initial situation (Weerd & Brinkkemper, 2007).

Design Evaluation

Guideline: "The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods".

The validation of the productization process is performed multifold; firstly we created a survey for the participants of a Software Product Management course. Secondly, we also interviewed several experts and product managers from the scientific field and practical field. In order to validate the applicability of the identified productization process, a specific case study is carried out at MP Objects. As a result, all three steps of the designed productization approach are executed and presented within section 7. Finally, for the validation of the implementation guidelines we interviewed several experts in the field of SPM. These guidelines are specifically designed for the framework and therefore knowledge of the reference framework is crucial.

Communication of Research

Guideline: "Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences".

Based on the results of this research we should be able to deduce answers to the research questions. In addition, we elaborated on the validity of the SPM reference framework, situational factors, and maturity matrix. We also generated future recommendations for MP Objects as a result of the business case which we carried out. At the end of this research we presented the final results and future recommendations to the board of MP Objects. Additionally, for the Utrecht University we created this study document and we defended the results at the end of the project. Finally, we are planning to submit a paper to present our main artifacts at 'The First International Conference on Software Business' in Finland.

2.2.2 Combined Design Science Model

Due to the lack of structure of the guidelines from Hevner et al. (2004), we combined the guidelines with the design science model (Vaishnavi & Kuechler, 2007). This combination gives a more clear organization to the research method. The combination of these two approaches can be found in Figure 2.

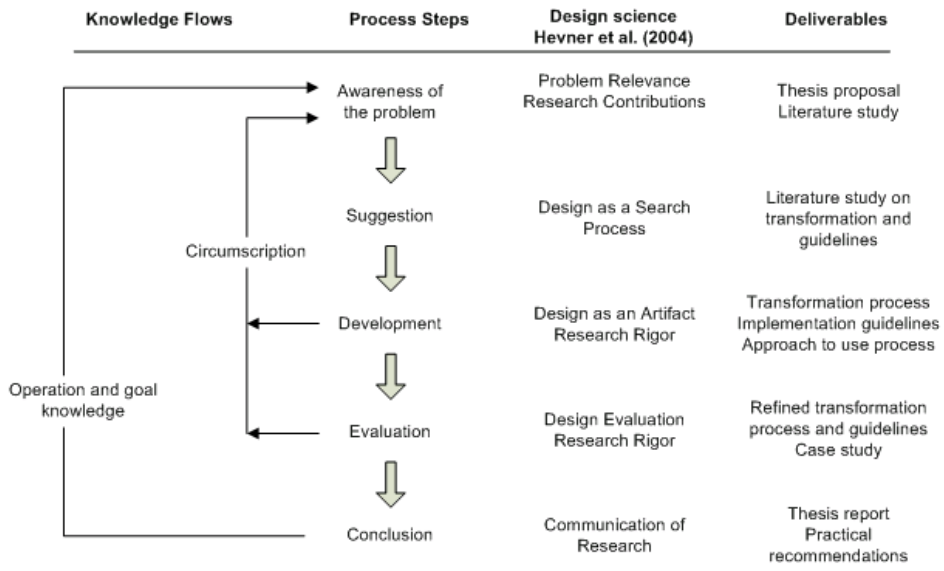


Figure 2: Composed design science method model

The two main knowledge creation flows are 'Circumscription' and 'Operation and goal knowledge'. The circumscription flow is important because it generates a certain understanding that only can be gained from the specific act of construction (Vaishnavi & Kuechler, 2007). Both the development and the evaluation phase are able to send the researcher back to the awareness of the problem phase and then the entire process restarts. The second flow (operation and goal knowledge) is related to the possibility that the conclusion can result into the awareness of a new problem (Vaishnavi & Kuechler, 2007).

The activities we carried out during this study resulted into several deliverables (right column of Figure 2). These deliverables are the outcomes of the following completed activities (an overview is provided in Table 1): Initially, we started with creating a proposal. By performing an extensive literature study we were able to define the exact research question and sub questions of this research. The data collection for answering the research questions was carried out multifold: An additional literature study is performed to retrieve available information related to the differences between customized and standardized software development and available transformations described in the literature. With the gathered information we were able to create an initial concept which graphically and textually described the productization process. During literature study we also carried out an assessment for MP Objects. With the results of this assessment we were able to determine the initial condition of SPM within MP Objects. As a result, we created a conceptual Process Deliverable Diagram (PDD) to visualize the organizational structure. After performing the literature study, we started with definition of the implementation guidelines for the SPM reference framework. By combining results from several different studies regarding software product management, we were able to define conceptual implementation guidelines. Exploratory interviews with experts in the field of software product management and with several employees of MP Objects were carried out to further develop and improve the artifacts. Consequently, this resulted into the final models and descriptions of the productization process and the final implementation guidelines. Additionally, for the validation, we also created a survey for a SPM course and we carried out a case study at MP Objects. The results of the case study are recommendations specifically for MP Objects and these recommendations can be used in order to continue becoming a product software organization.

Activities	Deliverable
- Literature study: background information, define problem description	Proposal
- Literature study: differences, similar transformations - Exploratory interviews - Development - Validation: expert interviews, survey for SPM course, case study at MP Object - Modification / finalization	Implications of productization process Productization process: stages, descriptions, approach for applying the productization process
- Literature study - Development - Validation: expert interviews - Modification / finalization	Implementation guidelines: general guidelines, guidelines per product management function
- Determining initial position: assessment, process deliverable diagram - Gap analysis: determination of suitable situation, specification of actual gap - Determination of recommendations	Case study at MP Objects: several recommendations

Table 1: Overview of activities and deliverables of this research.

2.3 Contribution

In this section, we elaborate to what extent this study contributes to the scientific field and societal field.

Scientific Contribution

The main scientific contribution of this research is the identification of a specific process in order to change from customized software development to standard software development in combination with the adoption of the SPM reference framework. The entire process of becoming a product software business is currently not widely studied and therefore a very interesting scientific topic. In addition, a graphical and textual description for each of the stages of this productization process is designed and validated during this study. Secondly, this research produced implementation guidelines which can be used when specific processes of the framework are put into practice within an organization. Thirdly, this research project is another validation of the SPM framework (Weerd et al., 2006a), the SPM Maturity Matrix (Weerd et al., 2009), and the Situational factors (Bekkers et al., 2008a).

Societal Contribution

The main societal contribution of this research project are the recommendations for the transformation from developing customized software to developing software as a standard product specifically for MP Objects. Additionally, this study provides a textual as well as a graphical description of the entire process which can support organizations while becoming market-driven. Finally, a list of guidelines for implementing the SPM reference framework is provided which organizations can use when they want to adopt the reference framework for software product management.

2.4 Validity

In conducting this research, we considered the four validity threats as described by Yin (2003). We applied these threats especially for the case study at MP Objects and we also considered them while creating the artifacts. The case study we performed is “an empirical inquiry that investigates a contemporary phenomenon within its real-life context, especially when the boundaries between phenomenon and context are not clearly evident” (Yin, 2003). Within this section we elaborate more on three of the validity threats from Yin. The fourth threat is not used because it is not applicable for

this research project. The *internal validity* threat is not relevant because it involves pattern matching, explanation building, and performing time-series analysis. In order to cover this threat, multiple case studies should be carried out.

Firstly, the *construct validity* is covered by using multiple sources of evidence for the data collection of specific information. For example for the determination of the initial position of MP Objects (section 7.3) we applied the viewpoints of a consultant and a developer in order to get a better representative result. Additionally, the assessment was carried out during a session where three employees of MP Objects were present.

Secondly, the *external validity* refers to the approximate truth of conclusions that involve generalizations. As for the exploratory interviews and validation interviews we combined the gathered results of the scientific field with the gathered results of the practical field. Also for the survey we tried to involve organizations which differ in several areas (by using the situational factors). As a result of the validation, we also tried to verify the results of the validation process with references from the literature before we applied them.

Finally, the *reliability* concerns the demonstration that the results of the study can be replicated. For the business case we covered this by creating an approach for the implementation of the productization process (section 7). Additionally, all other relevant information is documented so that the results can be replicated. However, when the productization process will be reused within another business case, it is better when a company is not already at almost the last stage of the productization process. Then the applicability and usability can be validated better.

3 Related Literature

Within this section we elaborate more on the related literature from several topics. The context of this section is designed in such a way that we first present a more or less top-down view of the results. Firstly, we present the terminologies of customized software and standardized software. Secondly, a process model and a reference framework for software product management are studied. Finally, we present an overview of differences which implicate on the productization process.

3.1 Software product

In order to understand the transformation from customized software development to a standard software product, we first identified the differences between these two development approaches. Currently, there is quite some literature available which elaborates on these differences. However, as a result of the literature study, we noticed that several different terminologies for customized software development and standardized software development are used (Table 2).

In terms of	**Customized software**	**Standardized software**
Stakeholder	Customer-driven	Market-driven
Software	Custom information systems	Software product
	Tailor-made software	Packaged software
		Commercial-off-the-shelf (COTS) software
Development	Bespoke development	Market-driven development
	Custom development	Packaged development
Management	Project Management	Product Management

Table 2: Varying terminologies used to identify the differences.

When we look at the different terminologies used in the literature, we identified the following definitions:

- Software product: “defined as a packaged configuration of software components or a software-based service, with auxiliary materials, which is released for and traded in a specific market” (Xu & Brinkkemper, 2005);
- Packaged software: “software sold as a tradable product (purchased from a vendor, distributor or store) for all computer platforms including mainframes, work-stations and microcomputers. Typically, packaged software is licensed for use, not sold” (Sawyer, 2000);
- Bespoke software development: “is also known as developing software based on a contract with a specific customer” (Natt och Dag, 2005);
- Tailor-made software: “is customized software that is developed according to the needs and specifications of individual customers” (Hietala et al., 2004);
- Commercial-off-the-shelf (COTS): “is based on a market driven development and products usually offer more functionalities than customers actually need” (Alves & Castro, 2006).

3.2 Software Product Management

Product management is getting more attention within software organizations (Weerd et al., 2006b). Within this section we elaborate more on particularly software product management and other models or frameworks related to product management.

Software Product Management (SPM) consists of the development of software for a relatively anonymous market, instead of developing custom fit information systems. Processes which differ the most from project oriented software development are especially requirements engineering, release management, and marketing & sales (Helferich et al., 2006). In order to have successful product management, it is essential to deliver the right product at the right time for the right market(s). In addition to that, the success depends on many factors and stakeholders. Companies need to have an optimal combination so that software meets customers' needs together with being on time and on budget (Ebert, 2009).

The quality of products can decrease when product management activities do not get enough attention. Ebert (2009) studied the upstream root causes of insufficient product management and identified several early symptoms. These symptoms resulted into tangible problems, which can be found in Table 3.

Upstream Root Causes	**Early Project Symptoms**	**Tangible Problems**
Vague product vision and strategy	Conflicts of interest; commitments not maintained	Rework
Key stakeholders not integrated	Unexpected dependencies between components	Delays, overruns
Unknown project dependencies	Unclear cost / benefit	Scope creep
Business case not evaluated	Incoherent set of features	Wrong content
Needs not understood		

Table 3: Root causes adopted from Ebert (2009).

SPM has become an essential key-process for small companies which base their whole business on one or two product(s) which they develop and sell (Kilpi, 1997). It often occurs that small companies forget the importance of marketing and delivery of software. Kilpi (1998) therefore created the Software Product Management process model (SPM-process model, Figure 3). This model is an extension of the Software Configuration Management with the addition of the areas marketing and delivery. Additionally, the process model consists also of the production and development areas. The four used areas are based on research done by Pine (1993) and the association is designed in such a way that they complete each other in order to reach the company objectives (Kilpi, 1998).

Another model is the Software Product Management (SPM) reference framework. This is based on the results of a performed literature study and field studies with product managers. The structure of this reference framework consists of the key process areas, stakeholders and their relations, as is denoted in Figure 1 (Weerd et al., 2006a).

3.2.1 SPM process model vs. SPM reference framework

When we compare the process model (Figure 3) and the reference framework, we noticed one significant difference. The readability of the reference framework is much more clear compared with the process model. The process model consists of four activity areas with in-between a number of curved arrows which affect the readability and there is not really a logical structure. On the other hand the reference framework consists of a clear / more hierarchical structure with a clear overview of the involved stakeholders.

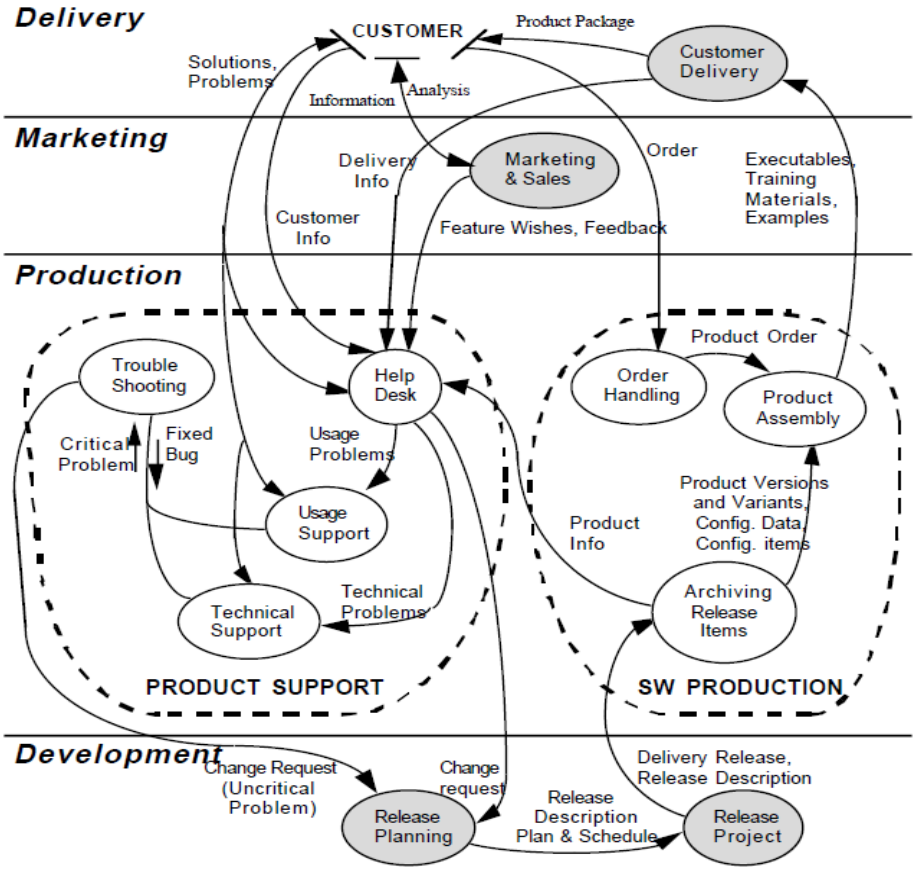


Figure 3: Software Product Management process model adopted from Kilpi (1998).

Secondly, if we evaluate the functional part of the process model we notice that other areas like delivery, marketing and support are also involved within the process model. The reason why these activity areas are involved is because the process model is an extension of Software Configuration Management that includes aspects related to marketing and delivery of software (Kilpi, 1998). Firstly, the 'Delivery' activity manages the customer deliveries. Secondly, the 'Marketing' activity collects and analyses the market and the customer information. It also informs the market and customers of the products and the new product releases. Thirdly, the 'Production' activity supports the customers' problems and hands the orders and deliverables. Finally, the 'Development' activity manages the product change process and plans the release projects. Kilpi states that the activity areas delivery and marketing are vital parts of a business and sometimes easily forgotten in small companies.

The reference framework is not direct related to marketing and delivery of the product software; it focuses on managing the products, releases, and the requirements (Weerd et al., 2006a). It consists of all the involved stakeholders and their relation with product management. If we look at the process model we see that it is hard to determine the involved stakeholders. Take for example the process of gathering requirements; it is not clear from which stakeholder the requirements are gathered. Finally, the product manager plays a central role in the reference framework and he/she interacts and collaborates with the other stakeholders. The process model on the other hand doesn't mention the product manager function at all.

Finally, when we look at the research activities on Software Product Management, we see that Weerd is actively improving and performing further research on this field. Weerd et al. (2009) for example recently developed a method in order to determine the maturity of each of the processes within the SPM reference framework. Kilpi (1998), on the other hand, stopped with studying the SPM process model and other relevant research areas in the field of product management.

3.2.2 Reference framework

The structure of this reference framework is based on its core, the software product itself, and it is structured in a hierarchical way (illustrated in Figure 4). On top of this model is the 'Product portfolio' and this is the complete set of products of a company. The portfolio of a small company usually consists of one product and the activity of managing the portfolio therefore often forgotten or left out. Furthermore, each product has several 'Releases' and each release has its own list of 'Requirements'. Each of the requirements implies the addition of a technical, functional feature or non-functional feature (performance constraints or availability requirement) (Weerd et al., 2006a).

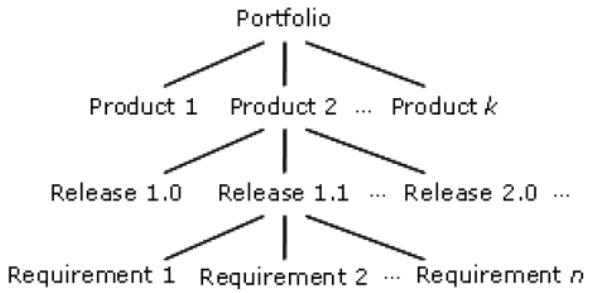


Figure 4: Artifact hierarchy of product management

The SPM reference framework (Figure 1) consists of internal and external stakeholders. The external stakeholders are: the market, the customers, and partner companies. On the other hand the company board, sales & marketing, research & innovation, development, support, and services are the internal stakeholder (Weerd et al., 2006a). Product management consists of the following four functions: 'Portfolio management', 'Product roadmapping', 'Requirements management', and 'Release planning'. We elaborate more on each of these functions in the following sections.

3.2.2.1 Portfolio Management

Research performed by Bekkers et al. (2008b) showed that a remarkable 42% of the interviewees had no processes for handling the high-level management activities (Portfolio Management and Product Roadmapping). Also Vähäniitty (2004) concluded that Portfolio Management (PM) is largely overlooked by many (small) product-oriented software companies. In addition, Vähäniitty mentions that portfolio management within a small company emphasizes the content of upcoming releases for relatively few products. However, management of product development activities as an explicit portfolio is crucial to the long-term success of product software companies (Vähäniitty, 2005). Therefore, Vähäniitty presents an approach for implementing portfolio management in (small) product oriented software companies along with initial experiences. The approach is an integration of portfolio management basics with modern, time-paced software development processes for the small company context.

The PM area of the SPM reference framework consists of the decision making about the set of existing products. Processes of PM are: monitoring market trends, deploying a product development strategy, making decisions about the product lifecycle, and establishing partnerships and contracts (Weerd et al., 2006a). An important key decision making process of the portfolio management function is Product Lifecycle Management (PLM). PLM includes planning and controlling of processes that are required for managing data, documents, and enterprise resources during the entire product lifecycle (Abramovici & Sieg, 2002). Research done by Abramovici & Sieg verified that most of the small and medium sized enterprises (SMEs) are not focusing on PLM. They mention also that it is important to adopt PLM because this could enable an organization to improve operational efficiencies associated with the entire life of a product. Another important part of the PM area is the identification of Product Lines (PL). A PL is a range of software products sharing a common platform

which reduces the development costs, increases the quality of products, and it reduces the time-to-market (Pohl et al., 2005).

3.2.2.2 Product Roadmapping

Vähäniitty et al. (2002) studied the need for Product Roadmapping (PR) by stressing that success requires the release of new products and product upgrades with the right amount of features and quality within an open market window. They present an approach based on PR that is able to support small companies in their product planning function. In practice, PR is often missing due to inexperience, unclear priorities, time-to-market pressures or the lack of suitable process infrastructure (Bosch, 2000). Furthermore Vähäniitty et al. (2002) concluded that product roadmapping can play an important role in bridging the gap between management, marketing and product development. Management should consider both product positioning and development aspects at the same time. Roadmaps can be designed in several forms ranging between the two boundaries technology push (divergent and looking for opportunities) and market pull (aiming for a customer defined product) (Phaal et al., 2004). The roadmap should find a balance and combine these two boundaries, so that a time-based graphical framework for future development and strategic plans can be created. A technology roadmap is a powerful technique to support technology management and planning. Additionally, it is especially useful for exploring the dynamic linkage between technological resources, organizational objectives, and the changing environment (Phaal et al., 2004).

The processes within the Product roadmapping area of the SPM reference framework are related to theme identification and core asset identification. The process theme identification is related to all forecasts concerning technology or market trends. Core assets are components that are shared within multiple products (Weerd et al., 2006a). An important factor for designing the product roadmap is the identification of the market and technology trends (from the portfolio management function). The result of this process is used in order to create an expectation of the content of the future releases.

3.2.2.3 Release Planning

Within market-driven software product development release planning is an essential and complex process (Carlshamre, 2002a). Selecting the requirements for the next release is an inherently complex task due to the varied interests of the stakeholders which are involved. Each of the stakeholders may have their own preferences for the implementation of specific requirements within the next release (Akker et al., 2005). Akker developed an optimization technique to support software vendors in determining the next release of product software. The technique is based on Integer Linear Programming (ILP) which is a method that looks at the required development time / effort and the estimation of the revenue. As a result of that it will generate an optimized list of requirements that should be developed for the next release. Akker et al. (2007) extended this research by also taking different aspects and managerial steering mechanisms into account. They considered: 1) One pool of developers, when for example no different development teams are used; 2) different teams without team transfers, each with its own capacity constraint; 3) different teams with team transfers allowed; 4) hiring external team capacity; 5) extension of the development project deadline; 6) requirement dependency (functional, revenue and cost related). The identified aspects and mechanisms can be easily combined with an integer linear programming model. Another release planning technique is the feature prioritization matrix (Wiegers, 1999), which prioritized requirements based on value, cost and risk. Finally, there is also the cost-value approach which is a technique designed to maximize stakeholder satisfaction (Karlsson & Ryan, 1997).

The Release Planning (RP) area of the SPM reference framework consists of the processes related to the development of a new release. Before developing a release the stakeholder starts with prioritizing and selecting the product requirements. After the selection of the requirements, the

release definition is written. After that the release definition is validated by all the involved stakeholders. Finally when this is approved by the board, a launch preparation is created and sent to the stakeholders (Weerd et al., 2006a).

3.2.2.4 Requirements Management

When creating software for a market, rather than for one specific customer, the pressure on short time-to-market is evident. In order to be able to meet the market demands effective engineering of software requirements is essential (Höst et al., 2001). During the elicitation and analysis of requirements information it often occurs that customers' requirements are poorly understood and inaccurate assumptions are made. This has significant negative implications on the design and development of the product in terms of quality, lead time and costs (Jiao & Chen, 2006). Jiao & Chen also identified that engineers find it difficult to translate the gathered customer requirements into concrete product and engineering specifications. Furthermore, they identified three main processes within RM, 'Requirement Elicitation', 'Requirement Analysis' and 'Requirement Specification'. The Requirements Management (RM) business function consists of gathering, identifying and the organizing the requirements for product software.

When we compare the processes used by Weerd et al. (2006a) and by Jiao & Chen (2006), we see that there is a big similarity between the defined processes. Requirement elicitation is related to the 'Requirements gathering', they both consist of the process of getting requirements of customers and other stakeholders (Jiao & Chen, 2006). However, the main difference between these two approaches is that eliciting requirements is performed before a project starts and gathering requirements is an ongoing process. Furthermore, the requirement analysis phase of Jiao & Chen is related to the derivation of explicit requirements that can be understood by marketing and engineering. This process is roughly similar to Weerd's 'Requirement Identification' process. In addition, Weerd's requirements identification process describes the requirements in a company's perspective and context. Finally, the last process of the RM shows the biggest difference. The 'Requirement Specification' process defined by Jiao & Chen is about the definition of concrete product specification in the functional domain. Additionally it involves continuous interchange and negotiation within a project team regarding conflicting and changing objectives. Weerd's 'Requirements Organization' process on the other hand is related to the association of the requirements per theme and core asset.

3.2.3 SPM Maturity Matrix

The first conceptual method for determining the maturity of processes within a product software company is the Product Software Knowledge Infrastructure (PSKI). The PSKI is an online systematic collection of methodical knowledge which allows product software companies to obtain a custom-made advice for improving specific processes. A fully developed PSKI can offer companies a solution to help mature their processes in an easy comprehensive, incremental way (Weerd et al., 2006b). The PSKI method exists of four different steps: 1) the determination of the current situation; 2) the selection of process alternatives, by selecting the right method fragments for process improvement based on situational factors; 3) putting the method fragments together and implement the advice; 4) give feedback from the company in the PSKI, in order to improve the PSKI (Bekkers et al., 2008b). To identify the initial situation and maturity of the SPM processes within a company, Weerd et al. (2006b) created the capability maturity matrix. This capability matrix is a maturity model in which method fragments are linked with a certain maturity level, in order to be able to provide companies with advice to fix their process need and to mature their processes.

The SPM Maturity Matrix is a follow-up of the capability maturity matrix and further developed and validated by Weerd et al. (2009). The main goal of the maturity matrix is to help identify SPM problem areas and not to give a rating on organizations. The matrix consists of an overview of the

different processes of the SPM reference framework and their related maturity levels. However, the maturity matrix is applicable for all kind of product software companies. In order to determine to what extent certain processes need to be improved, Bekkers et al. (2008a) created a list of situational factors to improve SPM processes in a more optimal manner. With the situational factors a small or medium sized organization is able to determine to what extent they should improve specific processes. The list of situational factors gives the influence weight of each factor on the SPM processes so that it is possible to determine which method is best suited. The factors are initially based on a broad literature study and are validated by fourteen interviews with experts from both the scientific and practical communities. With the factor weights it is possible to determine the desired / best suitable maturity levels 1) of product management in general; 2) for each of the four product management areas; 3) for each of the processes within each product management function. With the calculated maturity levels it is possible to select the best fitting SPM method from the PSKI in order to increase software product management processes.

3.3 Productization

Currently, there is barely any literature available on the explicit transformation from developing software for one specific customer to the development of software product for an entire market. Within this section we elaborate more on the retrieved results from the literature study.

3.3.1 Customer focus

Research done by Codenie et al. (1997) identified the essence of creating software as product, but software should be highly customizable so that the customers can still modify it to their specific needs. Codenie et al. described the transformation to framework-based development so that there is a possibility to respond to new and rapidly changing market opportunities and still customize the software for specific customers. This transformation combines two development process models: Firstly product development, because a product (the framework) is offered and secondly project development, because customers are involved in customizing the product to their specific needs. The transformation described by Codenie et al. is based on a stepwise and iterative construction and it started when the original application (of their first customer) was adapted towards the specific needs and infrastructure of the second customer. However, by reusing and adapting previous projects the overall maintainability decreased and therefore they decided to look into framework technology. Eventually, after a number of iterations the result of this approach is a standardized framework for all customers, with for each customer a customized part.

During the transformation, the following additional problems occurred and the following challenges can be used when the transformation process is carried out:

- *Reuse documentation*: The long life span and the strategic role of frameworks means that good design documentation is essential.
- *Version proliferation*: Important when maintaining solutions is to keep tracking the changes for specific customized solutions. Without reusing information on how a customization relies on the framework, it is very difficult to estimate the effort what is needed to update a customization with a new version of the framework.
- *Delta analysis and effort estimation*: A delta analysis is performed by browsing through a prototype with a new customer and identifies the customers' needs. However, when performing a delta analysis with a new customer, it is difficult to assess what parts of the product (in terms of programming code) can be reused, what parts need to be adapted before reuse, and how much effort is required by the adaptation.
- *Architectural drift*: Due to ignorance, deadline pressure, and the not-invented-here syndrome often solutions that are created are either reinvention of designs already present in the framework or solutions that unnecessarily break the framework architecture. It is necessary to enforce the architecture of the framework without over constraining the customizer.

- *Over featuring*: Migrating features in order to reduce customization might result in over featuring, so applications containing features not relevant for all involved customers. Over featuring makes the framework more expensive, more complicated, and less reusable for future customers.

Transformation described by Codenie et al. (1997) is useful for the initial stages of the transformation process because at a certain moment a company is able to identify the relevance of creating a standardized product. In order to become a product software company, it is essential to change by using a stepwise iterative construction. Keeping the customer satisfied during the transformation is also an important challenge. This challenge is also studied by Vandermerwe (1995) and this study describes briefly the transformation of IBM. The main problem which occurred during this transformation is the fact that IBM failed to recognize environmental changes and customers preferences. As a result of that other companies filled in the empty space left by IBM. IBM had lost sight of its customers and marketplace because their strategy for dominance replaced customer responsiveness with arrogance and total lack of concern. When the new CEO took over the management function they changed this by asking the top 100 customers what IBM was doing right and wrong (Vandermerwe, 1995). So keeping focus on the customers while transforming is very important, otherwise a company might lose their customers.

3.3.2 Requirements Engineering

Another transformation process described in the literature is the transformation of the requirements engineering process from BESpoke software development to MARkeT software development, also identified as BESMART (Bergström & Dahlqvist, 2007). This framework especially focuses on changing the requirements engineering process with a three step framework:

1) *Evaluate the organization's potential to become more Market-Driven.*
   This first step is considering how much more Market-Driven an organization actually can and wants to become. This decision should be based on unique opportunities and threats faced by the individual organization. Bergström & Dahlqvist identified ten issues which must be considered before the transformation takes place. A SWOT (Strengths, Weaknesses, Opportunities, and Threats) analysis must be used in order to determine the opportunities and threats. By specifying the opportunities and threats an organization is able to conclude whether or not a more market-driven approach is the best solution.
2) *Assess the organization's current RE process to determine what inhibits the organization from becoming more Market-Driven.*
   Secondly, an organization should carry out an assessment in order to resolve what areas need to be changed for the organization to become more Market-Driven. The assessment consists of four main areas: Analysis, motivation, transformation, and education.
3) *Recommend practices for becoming more Market-Driven based on the assessment results.*
   The third and final step is to recommended practices for process improvement. These are selected by comparing the results from the previous step with several good practices listed in BESMART.

As a result Bergström & Dahlqvist also identified a number of challenges for the transformation of the requirements engineering process from bespoke to market-driven. These challenges are related with: stakeholding (remote customers, internal stakeholding, different levels of abstraction), gap between the departments, requirement volatility, release planning (prioritization, requirements dependencies, time-to-market, resource allocation), and continuous flow (requirements overload, internal traceability, duplicate requirements).

3.4 Implications on SPM productization

Within this section we elaborate more on how specific differences between customized software development and standardized software development effect on the productization process. In addition, we also identified the specific differences per SPM area (product management function).

Sawyer (2000) performed a study on the difference between packaged software and custom information system development. Compared with software products, packaged software refers more to upscale enterprise software suites which are usually ready-made and rarely come ready-to-run. Furthermore, large package software usually also requires a relative large deployment and implementation period in order to fulfill specific needs of each individual business. Packaged software also involves the instructional materials (such as handbooks, manuals, update information, and support services) for the implementation (Xu & Brinkkemper, 2005). More details on the differences between packaged software and custom information systems adopted from Sawyer (2000) can be found in Appendix A: Differences, Table 15. But according to Sawyer, the two major differences between bespoke (customer specific) software development and market-driven software development are:

- Difference between stakeholders. Compared with bespoke software development, within a market driven development approach, the developer decides about the requirements and the developer selects the stakeholder representatives.
- Major pressure on time-to-market and the software product is often offered to a market by regularly releasing new releases require a careful release planning and requirements prioritization.

In addition to Sawyer, also Keil & Carmel (1995) identified relevant differences between custom software development and packaged software development (Appendix A: Differences, Table 16). This research looked at the explicit link and communication between customers and developers of packaged and custom development environment. The differences are used in order to provide an advice how companies can improve the exchange of information between customers and developers. Keil & Carmel identified three lessons out of an exploratory study of 31 software development projects. First, they found that more successful projects had more links between customers and developers compared with less successful projects. Secondly, they observed a great quantity of indirect links among many of the projects in their sample. However, the use of indirect links is less desirable to use because a customer and developer do not deal directly with one another. This can result in information filtering and distortion. Finally, the third lesson is related to the different types of links used in custom and packaged software environments. Both approaches had one link that was widely used and highly rated, but rarely used in the other environment.

Finally, Natt och Dag (2005) identified that market-driven software development has gained importance because software development companies turn to new and larger markets. They provide a summary of the differences between market-driven and bespoke management (Appendix A: Differences, Table 17) adopted from Carlshamre (2002a) with additions of Lubars et al. (1993) and Robertson & Robertson (1999). These differences are adopted from the results of the PhD which Natt och Dag carried out. The subject of his PhD is related to the area of requirements management in a market-driven environment. Studied were the bottlenecks in market-driven requirements management processes and linking customer wishes to product requirements through linguistic engineering.

3.4.1 Portfolio management

According to Cleland & Ireland (2002), portfolio management for customized software development consists of a strategy that moves the selection and implementation of projects from a random process to one with structure and discipline. Project portfolio management is used in order to align

projects with strategic goals and objectives so that an organization is working more effective and efficient. Furthermore, project portfolio management adds a dimension to an organization's capability and plan for growth. On the other hand, the product portfolio management function is about how an organization spends its capital and resources, and which development projects it should invest in (balancing available resources, making new product and technology choices, allocating resources for R&D operations). Additionally, portfolio management is a method by which an organization is able to operationalize their business strategy (Cooper et al., 2000).

3.4.2 Product Roadmapping

One of the main differences between customer driven and market driven software development, is that a market-driven software development consists of a larger amount of involved internal and external stakeholders. Roadmapping is a technique that software companies use in order to manage the high-level view and to link aspects of business to requirements engineering (Lehtola et al., 2005). Vähäniitty et al. (2002) studied the implementation of a roadmap for a small software company. As a result of this, they found that it sometimes seems 'too much' to establish a solid roadmapping process. However, using roadmaps from an early stage can prove valuable as a company grows and help to maintaining the focus in development. Vähäniitty identified three important values:

- Help in coordinating a complex set of activities;
- Explication of the direction of intent;
- Help in making short-term decisions, long-term decisions and trade-offs.

As a result of the found literature we concluded that the main difference between customer-driven and market-driven software product development is the fact that developing customized software is not based on any form of roadmapping. This process is specially designed for product software development with frequent releases. Developing customized software is based on a fixed (build-to-order) time schedule and has a usually a lifecycle of one release with additional maintenance updates.

3.4.3 Release planning

Release planning is the phase "where requirements engineering for market-driven product software meets the market perspective" (Carlshamre, 2002a). It is a specific process for market-driven development because customer-driven software development is not focusing on releases at all. The release planning function is highly related with the lifecycle of software. Where a customized software solution is based on one release, a standardized software product is based on more (frequent) releases. Research done by Carlshamre (2002a), exposed that the basic planning problems differ significantly between traditional customer-driven software development and market-driven (product oriented) development:

- Delivery of market-driven software is based on a fixed delivery date, frequent releases are planned ahead. On the other hand, customer-driven software development estimates the delivery date based on a set of requirements. Together with the customers the developed software is validated before it is delivered.
- The resources necessary for the development of the software in a market-driven organization is based on a more or less fixed set of available resources. Customer-driven software companies estimate the necessary resources demand based on a set of requirements.
- Requirements selection in a market-driven organization is performed by selecting an optimal subset of requirements for implementation. While selecting the requirements the available resources must be taken into account. In contrast, in a customer-driven organization this process is based on a more or less fixed set of requirements.

3.4.4 Requirement Management

The biggest difference between market-driven and customer specific software development is believed to lie in the Requirements Engineering (RE) process area (Gorschek, 2006). RE involves activities like eliciting requirements, analyzing requirements, prioritizing requirements, and maintaining a database of requirements. It traditionally focuses on bespoke software development where a specific system is developed based on a contract with usually one customer (Höst et al., 2001). On the other hand, market-driven software development is not specific for one customer but for an entire market. This requires proper requirements management so that all customers' wishes can be managed easily. In addition to the regular differences between market driven and customer specific software development, Alves et al. also looked at particularly the differences between the requirement elicitation phase, specification phase, prioritization and negotiation phase, and the validation phase (Appendix A: Differences, Table 18).

BESMART is a framework for shifting from BESpoke to MARkeT-driven RE. Bergström & Dahlqvist identified the general differences between the bespoke (customer-driven) RE and market-driven RE. The result is a list with general differences related to the different areas of the RE processes, which are summarized in Appendix A: Differences, Table 19. In order to cover these differences and become market-driven, Bergström & Dahlqvist created a framework for adopting a more market-driven requirement engineering processes. This framework is designed by using a SWOT (Strengths, Weaknesses, Opportunities, and Threats) analysis and the identified differences.

3.4.5 Summary

Within this section we give an overview of all the identified differences between customized software development and standardized software development. The results are summarized within Table 4 (illustrated on the next page).

	Customized software	Standardized software	Reference
Primary goal / pressure	Compliance to specification / costs	Time-to-market / time	Sawyer (2000), Natt och Dag (2005), Bergström & Dahlqvist (2007)
Measure of success	Satisfaction, acceptance, ROI	Sales, profit, market share, good product reviews	Sawyer (2000), Natt och Dag (2005), Keil & Carmel (1995)
Type of products	New system project; maintenance	New products; new versions	Keil & Carmel (1995),
Lifecycle of products	One release, then maintenance	Several releases, dependant on market demand	Natt och Dag (2005), Bergström & Dahlqvist (2007),
Coordination	Project portfolio	Roadmap	Vähäniitty et al. (2002)
Portfolio	Projects, resource allocation (new customer project, research projects)	Products, resource allocation (making new product and technology choices, R&D).	Cleland & Ireland (2002), Cooper et al. (2000)
Main stakeholders	Customer organization,	Developing organization	Natt och Dag (2005), Bergström & Dahlqvist (2007)
Customers	Usually one	Many	Keil & Carmel (1995), Höst et al., (2001)
Customer identification	Before development begins	After development ends and the product goes to market	Keil & Carmel (1995)
Users	Known or identifiable	Unknown, may not exist until product is on market	Natt och Dag (2005)
Distance to users	Usually small and more involved with development	Usually large and less involved with development	Sawyer (2000), Natt och Dag (2005), Keil & Carmel (1995)
Requirements origin	Elicited, analyzed, validated	Invented by market pull or technology push	Natt och Dag (2005), Alves & Castro (2006)
Requirements specification	Used as contract between customer and supplier (typically document-based)	Verbally communicated and managed individually	Natt och Dag (2000), Alves & Castro (2006), Bergström & Dahlqvist (2007)
Requirements selection	More or less fixed set of requirements, negotiate with customer	Optical selected subset, support requirements selection for release planning	Carlshamre (2002a), Alves & Castro (2006)
Development philosophy	Iterative development is less common, usually waterfall	Iterative development is more common, usually agile	Natt och Dag (2005)
Developer's team	Grows larger over time	Likelier to be small	Sawyer (2000)
Required resources	Estimation of resources based on a set of requirements.	More or less fixed set of available resources	Carlshamre (2002a)
Developer's association	Short-term (until end of project), assigned to multiple projects	Long-term, promoting, involved in entire lifecycle	Sawyer (2000), Natt och Dag (2005)
Validation	Performed with customer before delivery (ongoing process)	After product is released in the market (very late)	Alves & Castro (2006), Natt och Dag (2005)
Deliverance	Estimated delivery date based set of requirements, agreed upon with the customer	Fixed delivery (release) dates, when the requirements best fit a market window	Carlshamre (2002a), Bergström & Dahlqvist (2007)

Table 4: Overview of the identified differences.

PART II: Method construction

4 Productization process

In order to change from developing customized software to a standard product we present in this section the productization process. This section consists of the description of the entire productization process and the adoption of the product management functions from the reference framework for Software Product Management (Weerd et al., 2006a). The entire process is the result of a two dimensional practice. Firstly, the implementation of the reference framework for SPM and secondly the transformation from creating customized software solutions (project, illustrated in red) to developing standardized (product, illustrated in blue) software, illustrated in Figure 5.

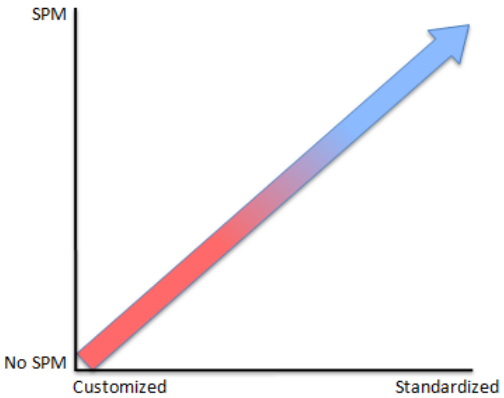


Figure 5: Two dimensional practice of the transformation

The data collection for the development of this artifact is based on the results of the literature study (section 3.3) and some exploratory interviews with experts in the field of software product management. The validation is carried out by interviewing several experts in the scientific field and the practical field. Additionally, also a survey is created to validate the productization process which is filled in by the participants of a SPM course. As a result, several changes are applied to the productization process. We elaborate more on the development of the productization process within this section.

4.1 Productization stages

This section describes the entire productization process in order to become a market-driven product software company. The combination of standardizing the software and still satisfying the customers resulted into seven stages, with two possible end stages (Figure 6).

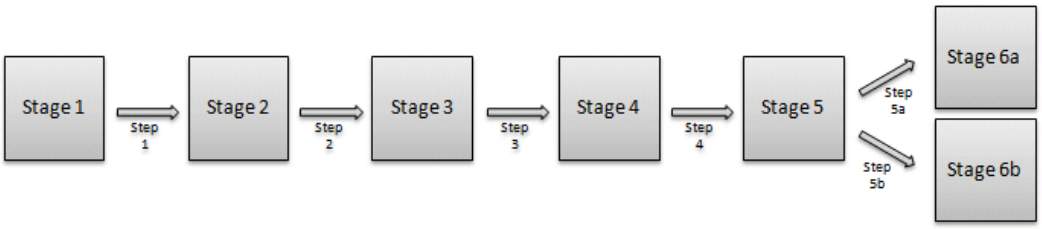


Figure 6: Productization process with seven stages and two possible end-stages.

The productization processes is an evolution from customer specific software to a software product and organizations are able to enter at each stage of the productization process. The productization process continues from the initially entered stage until one of the end stages is reached.

4.1.1 Introduction

The seven models describe the entire productization process for the evolution of customer specific software to a standard product. This entire process is especially applicable to and designed for a

company which is creating software based on a customer-driven development. When for example a company is already focusing on product development this productization process should not be used to improve specific processes. In addition, also when a company is introducing new products, another strategy should be followed. Consequently, this transformation is highly focusing on keeping the customer satisfied while transforming. The first few stages are a logical follow-up of the evolution of project software development. Additionally, we mapped the reference framework for software product management (Weerd et al., 2006a) with several stages of the productization process. Each of the stages consists of several aspects:

- Red (customized features): All red fragments denote a customized development process, which is usually based on a kind of waterfall approach (Natt och Dag, 2005).
- Blue (standardized features): The blue colored fragments are linked to the standardization of the software. Usually a more agile approach is used for the development of this type of software (Natt och Dag, 2005).
- Light gray: the fragments illustrated in light gray, represents the supportive processes which are carried out.
- Dark gray: the dark gray fragments on the right, are the relevant process areas 'Portfolio Management', 'Product Roadmapping', 'Requirements Management', and 'Release planning' from the Software Product Management reference framework. These fragments are initially transparent (stage 2) and during the productization process, the maturity of each function should increase. Eventually all functions should be as matured as possible.
- Relations: the relation between the entities and nodes represents the information flow between fragments. The relation between the product management functions are depicted from the reference framework for SPM (Weerd et al., 2006a).

The models describe the productization process from being customer-driven (customized software) to becoming market-driven (product software). We identified two possible end stages: customizable software product (stage 6a) and standard software product (stage 6b). The reason why we applied two end stages is based on the fact that there is often a need for some customization in order to integrate software for a customer specific situation (Hietala et al. 2004). It takes substantial time and effort to get enterprise product software up and running and therefore pieces need to be customized (Hoch et al., 1999). Cusumano (2004) recognized the intension of selling software products and providing support services. They identified this approach as a "hybrid approach" which consists of a customized (customer specific) part so that the product is better applicable within current infrastructures. We therefore applied the terminology of 'stage 6a / stage 6b' and not 'stage 6 / stage 7'. Otherwise, the intention of transforming from stage 6 to stage 7 can emerge. Within the scope of this research we did not studied the possibility to change between the two possible software product types.

4.1.2 Stage 1: Independent Projects

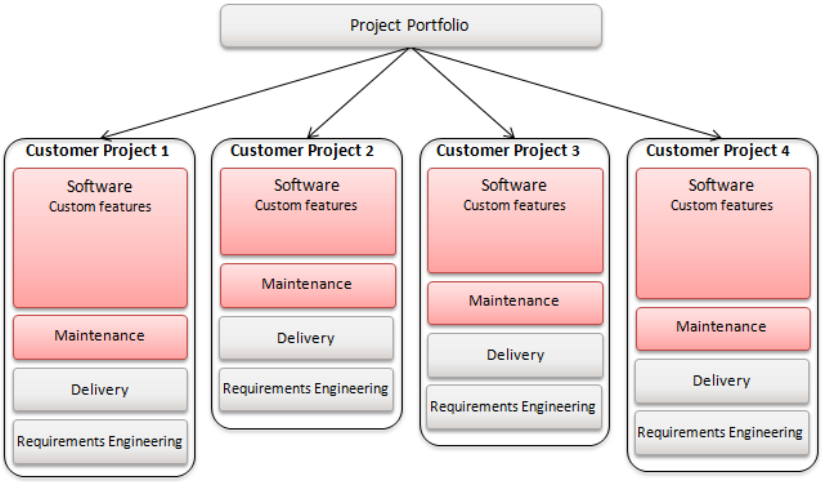


Figure 7: Stage 1 – Independent projects.

Characteristics:

- *Projects are executed independently*
- *Projects differ in budget, technology, and functionality*

General

This first stage represents a company with a project portfolio of projects which are executed independently. In this case a company is creating information systems which completely differ in size, budget, technology, and functionality. These projects have barely any standard common (shared) functions or features. The projects are especially driven by the customers and therefore they are the main stakeholders within this phase. The success of the projects is measured by the customer satisfaction and acceptance. Consequently, there is usually a small physical distance between customers and the developers (Keil & Carmel, 1995). Traditional approach used for the development of customized information systems is often based on sequential development process, which is also known as the waterfall methodology.

Project Portfolio

Usually customer driven organizations execute a number of independent projects simultaneously, also denoted as a portfolio of projects. The more projects a company executes, the more complex it becomes to manage them. The required resources for the development of the systems are estimated based on the customers' wishes which are elicited from the customer.

Requirements Engineering

The requirement engineering process consists of the activities related to the elicitation, specification, negotiation, and validation of the requirements (Alves et al., 2006). Keeping the customers involved is essential in order to satisfy them and therefore all customers' requirements must be obtained. Based on the (more or less fixed) number of requirements the delivery date and the required resources can be determined (Carlshamre, 2002b). After the software is developed it is validated together with the customer and after acceptance it is implemented. The lifecycle of these projects is based on one release and after that maintenance updates can be applied when the software consists of some critical bugs. When a customer wants new features added to the software, a new quotation should be created and a new project starts.

4.1.3 Stage 2: Project Feature Reuse

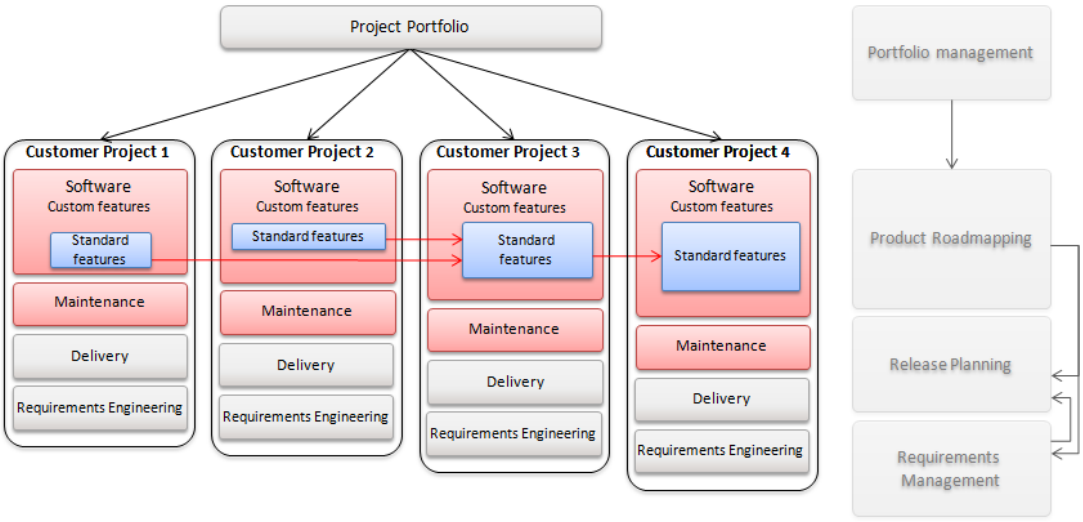


Figure 8: Stage 2 – Reuse across projects.

Definition of Project Feature Reuse: *Projects are executed differently and features or functionalities are reused across projects.*

Characteristics:

- *Focus on feature reuse across projects*
- *More custom than standard features*

General

The process starts with stage 2, when a company is developing new projects for different customers. The possibility of reusing specific already developed features, functions, components or modules from previous projects becomes more obvious. An advantage of reusing functionalities from finished projects is that the overall quality and reliability of the software can increase because specific parts have already been tested within previous projects. This pattern of reusing features from previous projects can continue throughout the upcoming projects and it starts forming a basic set of standardized features or core functionalities.

Important when producing new code which potentially could be used in future projects, is that the code should be created with the intention to reuse it. Otherwise, it will be difficult to merge the code with upcoming projects and it will not have any advantage later on during the development of a product. Within this stage the projects still have a dominant part of custom (customer specific) features which need to be developed. Reusing the standardized features from other projects is essential for the production of the product's core functionalities. In the model this is illustrated by reusing specific standardized features from 'Customer Project 1' and 'Customer Project 2', which eventually are used as a basis for 'Customer Project 3'. In addition, the emerged part of standardized features from 'Customer Project 3' is used for 'Customer Project 4'. This process continues until stage 3 is reached.

When we look at the stakeholder involvement during the development of the software we know that the customer is in this stage the main stakeholder. This implies that the software is developed by a build-to-order approach, which means that a software business develops information systems which are designed for a specific customer. The result of a build-to-order (contract based) approach is that the intensity of sales and marketing activities is rather small. Additionally, the focus on research and innovation projects on new technologies is not really required in this stage. Looking for new technologies is only performed when customers have a specific need for it. Consequently, main

involved stakeholders in this stage are the customer and potential partner companies (external stakeholders).

Project Portfolio
When an organization begins to base new projects on previous (finished) projects the overall complexity of the portfolio increases. Proper management of all projects within the portfolio is essential at this stage and future stages. Management of the project portfolio at this stage is related to the allocation of the company's capital (i.e. related to partnering and contracting) and resources (Cooper et al., 2000).

Requirements Engineering
There are minor changes within the requirements engineering process as a result of the increasing similarities between the different projects. The development approach is still based on build-to-order which implies that there should be as many customer requirements realized as possible in order to satisfy the customer.

4.1.4 Stage 3: Product Recognition

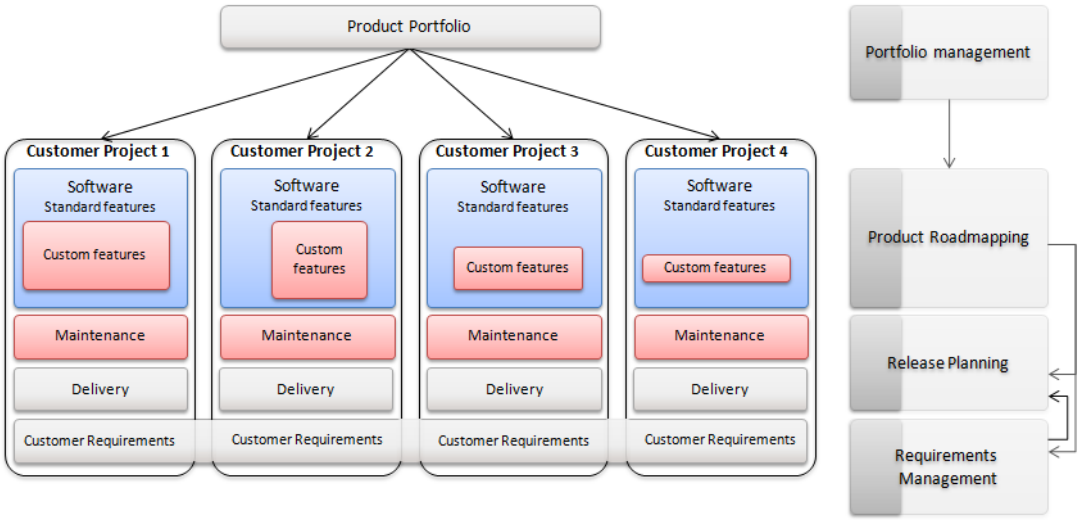


Figure 9: Stage 3 – Product recognition

Definition of Product Recognition: *Large parts of projects are reused and a product scope starts to emerge from the reused functionalities.*

Characteristics:

- *Shared features between projects*
- *More standard than custom features*
- *Customer specific maintenance*

General
The main difference with stage 2 is that at a certain moment the standardized part of the projects is bigger compared to the customized parts. Basically a company is able to identify the similarities of customers' wishes which results into a more generic basis for / recognition of a product. Important in this stage is that an organization should evaluate on the potential to become market driven. A SWOT analysis can be used in order to determine the strengths, weaknesses, opportunities and threats of becoming market driven (Bergström & Dahlqvist, 2007). When it is decided to start transforming to a standard product, stage 3 is the first step of creating a product. During this stage the main boundaries of the product emerge and based on that a company could identify a certain product for a specific market / purpose. Additionally, in this stage the Requirements Management should have

some attention, so that all new emerging customers' requirements are stored and can be managed from one central place. Managing the requirements of the customers is necessary because when a company starts to recognize a specific product there is still a need to satisfy the customers. As a result of that, the projects consist of more standard features than custom features. This implies that at this stage the focus is still based on a customer-driven approach and certainly not yet standardized enough for an entire market. Each project still has its own customized part merged into the main structure of the software in order to satisfy the customer. This merged customized part is different for each customer and this means that we cannot speak of a product platform. Also the maintenance and the delivery of the software is still customer focused.

For the productization process, also the introduction of the Product Manager (PMngr) function is essential. During this stage this function should be introduced within an organization and an employee should be assigned to this function. The PMngr is accountable for the strategic and operational decisions of the product and the PMngr is the first one to know when there is a problem. Also for the upcoming stages the introduction of the PMngr is important because he/she is also responsible for management of the portfolio, management of the requirements and product requirements, and management of the release plan.

Project Portfolio

No major changes occur in relation with the project portfolio. Each of the projects still consist of a customer specific part for each customer and the resources need to be allocated based on the complexity. Determining the required resources can be difficult within this stage, for example when customized features need to be merged into the standardized part several unforeseen changes can occur. Still the project consists of a fixed delivery date based on a more or less fixed list with requirements. The lifecycle of the project still is based on one release and the potential maintenance activities.

Requirements Management & Delivery

Based on the fact that at this point we still speak of customer-driven software development an organization should satisfy the customer by developing as many as possible wishes. At this stage we can speak of the initial point of becoming market-oriented and recognizing a specific product. The difference between the requirements activities of this stage and the previous stage is based on the fact that all customers' requirements are managed by implementing the requirement management function. This changes the process of eliciting all requirements from the customers separately to gathering requirements from all customers and storing them at one central place. Also the stakeholder involvement changes when transforming into stage 3.

Together with the internal stakeholders, the PMngr should look at the additional changes which emerge when specific customer requirements will be implemented. Basically the involvement of the stakeholders starts to change during the transformation, from focusing primary on one external stakeholder (the customer) to focusing on the internal stakeholders. This means that the customer involvement in the development of the product is decreasing and the involvement of the internal stakeholders is slightly increasing. The internal stakeholders should determine the scope and boundaries of the product. All additional (customer specific) functionalities are developed in the customized layer.

4.1.5 Stage 4: Product Platform

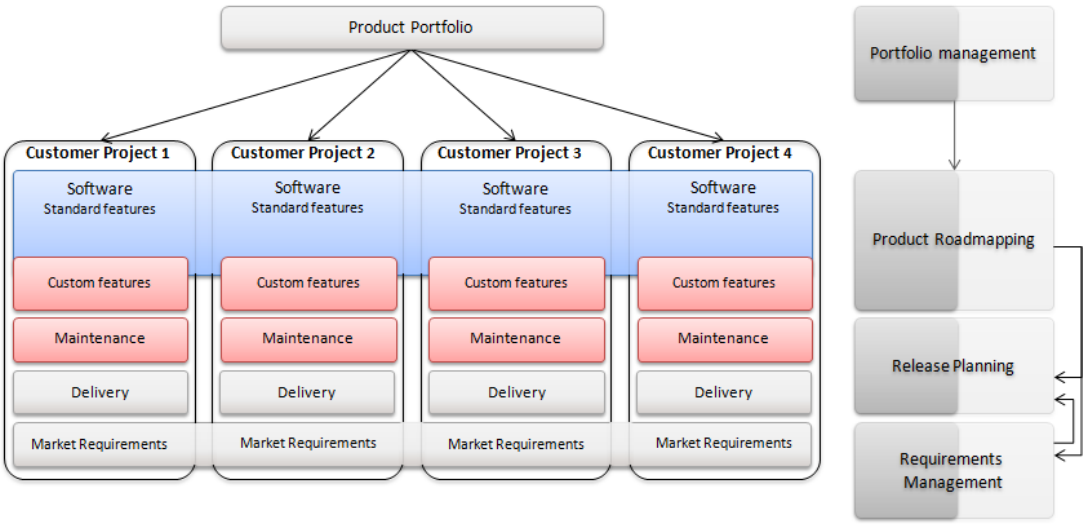


Figure 10: Stage 4 – Product platform.

Definition of Product Platform: *a set of features that form a common structure, from which a stream of derivative products can be efficiently customized, developed and produced.* This definition is based on the definition from Meyer & Seliger (1998).

Characteristics:

- *Generic product platform*
- *Customer specific maintenance*
- *Customer requests are handled as market requirements*

General

One of the major differences between developing customized and standard software product is the change in lifecycle of a product. In order to determine the future of a product a company should start focusing on future releases so that they start gaining a bigger market share. Due to the introduction of the roadmapping function a company can generate a long-term plan. The roadmap consists of the relevant information (in terms of themes and core assets) for the future development of the product. It should be designed in such a way that the customized part of the project decreases during the upcoming stages. This means that the main focus on satisfying the customer is decreasing and the focus of gaining more market share is increasing. The requirements management functions must start focusing on gathering all market requirements instead of only the customer requirements. These market requirements are needed in order to determine the content of the future software product for the rest of the process because eventually the customized part should be as small as possible.

Nevertheless, we still are not able to speak of a first release of a product at this stage; the amount of customized features within the projects is at this point too large. The standardized part of this stage can be seen as the initial start of creating the product by generating a standardized core. This core can also be identified as a generic product platform which is extended by a large customer specific layer.

Product Portfolio

From this point on we speak of a product portfolio instead of a project portfolio. However, each of the projects still consists of a specific customer specific part for each customer. While creating the product platform a company should consider collaborating with partner companies. The entire product doesn't initially have to be developed in-house; there is also the possibility that a product or

service from a partner company is adopted. An advantage of using products or services of partner companies is the fact that the reliability and quality of the product increases.

Product Roadmapping
During this stage main attention is towards the product roadmapping processes. A roadmap consists of the long-term planning which describes the future of a product and it is based on the market trends. Main processes next to the creation of the roadmap are the identification of themes and core assets. Themes are used for the organization of product requirements, which are identified based on the market requirements. Additionally, these themes are used to give the roadmap a clear direction. On the other hand there are the core assets, this are components that are shared by multiple products. The themes and core assets are used in order to plan the future mutations for the generic product platform in order to become a software product.

Requirements Management
Compared with the previous stage, the requirements management function should now focus on gathering market requirements. This means that from now on the customer requests are handled as market requirements. Additionally, also the requirements from the market (potential customers) must be gathered. The intensity of customized features must decrease in order to start developing a standard product for all customers / for an entire market. The result of that at this stage is a product platform which can be customized for each customer separately. The maintenance activity and the delivery of the product are still customer based. Also at this stage is the involvement of the internal stakeholders increasing. Together with the product manager they should consider whether or not specific features fit in the product platform or it will be custom developed for a customer. The decision of implementing specific parts within the product platform should be supported by the defined roadmap. However, while deciding which requirements will be included within the platform, the PMngr should consider "over featuring" (Codenie et al. 1997). Otherwise the product platform can consist of features which are not relevant for all involved customers or potential customers.

4.1.6 Stage 5: Standardizing Product Platform

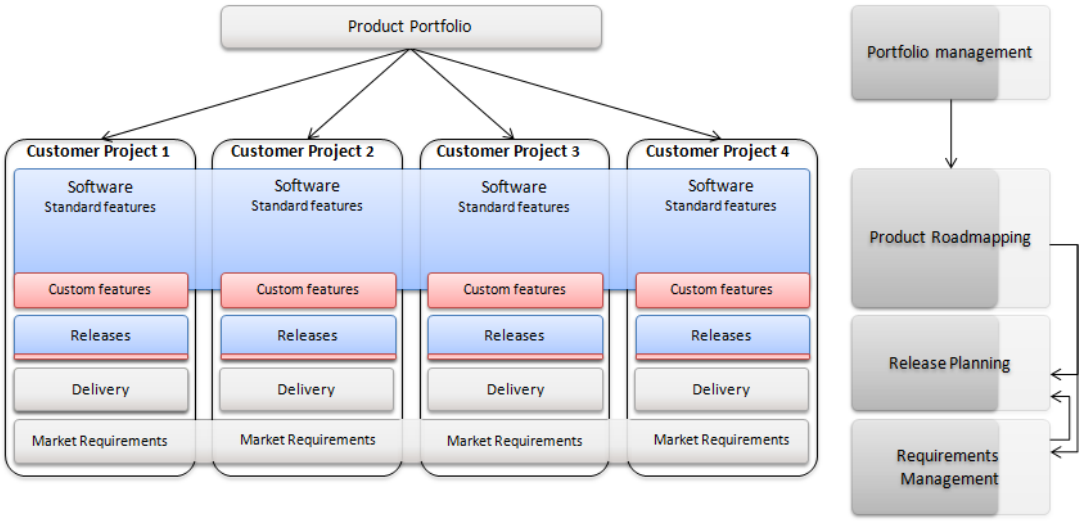


Figure 11: Stage 5 – Standardizing product platform.

Definition of Standardizing Product Platform: *increasing the set of features that form a common structure and introducing releases, from which still a stream of derivative products can be efficiently customized, developed and produced.* This definition is based on the definition from Meyer & Seliger (1998).

Characteristics:

- *Focus on generic product platform*
- *Requirements gathering based on market trends*
- *Event-based customized releases per customer*

General
The main focus of this stage is still the customer-orientation, but it starts to change towards a market orientation and bringing the emerging product to the market. Therefore a company should create a more standardized product which is applicable for more customers in the market. In addition, they can still, if necessary when a customer has got specific requirements, create for each customer an own customized part which is positioned on top of the product. The result of this approach is a standardized main product with additional event-based releases. The reason there is still a customized layer in place is because keeping the customer satisfied while transforming remains important.

Due to the introduction of event-based release, the lifecycle of the product is starting to increase. In order to manage this longer lifecycle, main attention of this stage is at the portfolio management function. One of the important decisions that must be made is the determination whether a company wants to focus on selling standard product software or selling product software with an additional customizable layer so that the product can be integrated within a company's situation. This decision influences the last stage of the productization process and should be made within this stage. As a result of that, a company should define strategic directions of the product and this involves the goal, market, and scope of the product. Consequently, these decisions determine the final end stage of the productization process.

Portfolio Management & Product Roadmapping
The main focus of the portfolio management function is changing the main business strategy from developing software for one specific customer to developing a product for an entire market. This change is essential, because Ebert (2009) identified a vague product vision and strategy as one of the five main root causes of insufficient product management. Together with the partnering decisions and the market & technology trends, the PMngr must identify a certain product line. The organization should also decide what products and which domains will be present in the product line. This product line is of influence on the designed roadmap, and therefore the roadmap must be updated based on the strategic decision.

Requirements Management & Releases
Within this stage the requirements gathering activity should also register the identified market trends and technology trends. Specific research projects or competitor analysis reveals new requirements. Additionally, also in this stage the customers' requests should be handled as market requirements. The customer requests which are not initially implemented should be documented so that they can be implemented in the near future. An important issue when adopting the requirements management and release planning functions is that it is possible that customers are not satisfied with the features of the first release. In order to satisfy the customers it is essential to develop in a balanced manner and still be customer focused. While selecting requirements for a release, the added (future) value of developing and implementing specific features should be considered.

Due to the continued development and increasing lifecycle of the product new features emerge and the maintenance is changed into event based releases. The frequency of delivering new releases is not scheduled within this stage, this is mainly ad-hoc and focused on events (for example a critical bug fix). In addition, not everybody gets the same release due to the fact that there is also a custom part for each customer. The portfolio management function is especially important for maintaining the customer specific solutions and keeping track of the changes for each specific customer (Codenie

et al. 1997). Otherwise, it is difficult to determine the effort what is needed to maintain a customized product with a new release.

4.1.7 Stage 6

This section describes the similarities and differences of the software product for stage 6a and for stage 6b. The main difference between these stages is related to the possibility of customizing the product software (stage 6a) so that it can be integrated to the customers' current information systems or infrastructure (Hietala et al. 2004). On the other hand, product software can also be designed in order to serve the mass market (stage 6b), with configuration possibilities and without any form of customization. The decision of selling product software as service or as license should be determined in an earlier stage of the productization process.

General

The main change in this stage is the change in the release planning function. Where in the previous stage the release was specified for each customer separately, this stage will generate one generic release for all customers without any customized features. Therefore, the requirements prioritization and requirements selection processes should focus on all stakeholders and not specifically on one customer in particular. Important for successful release planning is the involvement of the stakeholders with the composition and validation of the releases definition. As a result of this change, the 'Delivery' of a release specific for a customer (stage 5) is changed into 'Launch & Delivery' of the releases for all customers / entire market. The process of launching & delivering of the releases is further explained in the following section where it is specified for each end-stage. The involvement of the internal stakeholders is still increasing within this stage. The internal stakeholders have at this point a higher influence for the determination of the content of the releases. In addition, the marketing and sales department should focus in this stage on bringing the product to the market. Finally, this results in a decrease of the involvement of the external stakeholders on the development of the releases. Although, because of the customizable layer on top of the software product the contribution of the external stakeholder is higher in stage 6a compared with stage 6b.

Portfolio Management & Product Roadmapping

No major changes occur within the portfolio management and product roadmapping processes within stage 6a nor stage 6b; on the other hand the lifecycle, product line and roadmap should be updated based on made changes. In addition, the new release planning activities should be based on the identified roadmap. Central for this stage is the activity of bringing the product to the market and selling the product. Also the process of identifying the market trends and market requirements is important at this stage. Specific wishes of the market should be taken into account so that the new functionalities for the next releases can be determined. The product line specified earlier should be updated based on the identified trends.

Requirements Management & Release Planning

Compared with the previous stage the intensity of customized features should decrease in order to start developing a standard product. Therefore, the maturity of the requirements management and release planning activities should increase in such a way that the requirements prioritization and requirements selection processes are more based on a product oriented viewpoint. The requirements prioritization process should be based on the gathered request from all the customers and these requests should be handled and stored as market requirements. The generated roadmap should be used as basis for the selection of a subset of requirements for a release. The releases are launched and delivered for all customers when it is developed, tested and accepted. Compared with previous stage; this process consists of the implementation of the releases for all customers at once. More information for each end stage specific can be found in the following sections.

4.1.7.1 Stage 6a: Customizable Software Product

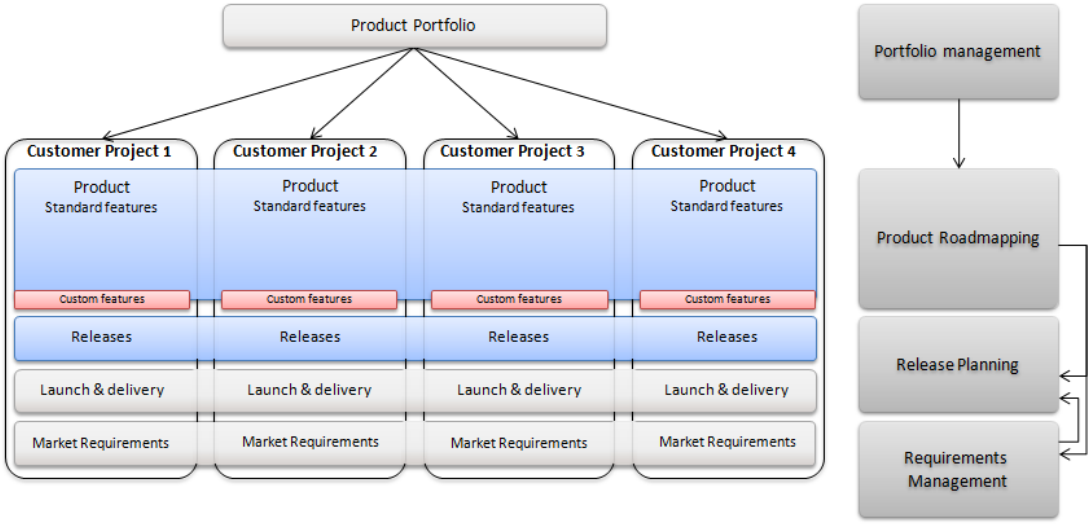


Figure 12: Stage 6a – Customizable software product.

Definition of Customizable Software Product: *a packaged configuration of a software-based service, with auxiliary materials, which is released for and traded in a specific market and customized for a specific customer*. This definition is based on the definition for a software product from Xu & Brinkkemper (2005).

Characteristics:

- *One standard product with customized layered part*
- *Structured releases*
- *Software business aiming at selling services*

General

For some companies there is still a need to be able to customize the software product so that it can be integrated within specific situations. This product type is characterized by software that is too complex to be sold 'off-the-shelf' and that requires customization or special integration and installation work (Cusumano, 2004). Therefore, there is the need for a customized (small) layer on top of the product so that the required functionalities can be added. Also Codenie et al. (1997) identified the essence of still having a customizable part in order to be able to apply the product to different situations. Additionally, Hoch et al. (1999) identified this type of product software as "enterprise solution systems". Usually this type of product software is developed for other enterprises and not specifically for individuals.

Requirements Management & Release Planning

This stage has still a small customized layer on top of the product and this layer is special designed for each customer so that the software product can be integrated within for example existing infrastructures. However, when we compare this layer with the previous stage, this layer should be considerably smaller and related to providing a service. Due to the addition of the customized layer on top of the product, the complexity of integrating the product for new customers might take considerable more time and effort. New customers will have a more complex process of launch and delivery due to the integration and collaboration within existing information systems. According to Hoch et al. (1999), when for example an ERP system is installed, 30% of the total costs is spend on the product license and 70% for providing professional services to implement the product.

4.1.7.2 Stage 6b: Standard Software Product

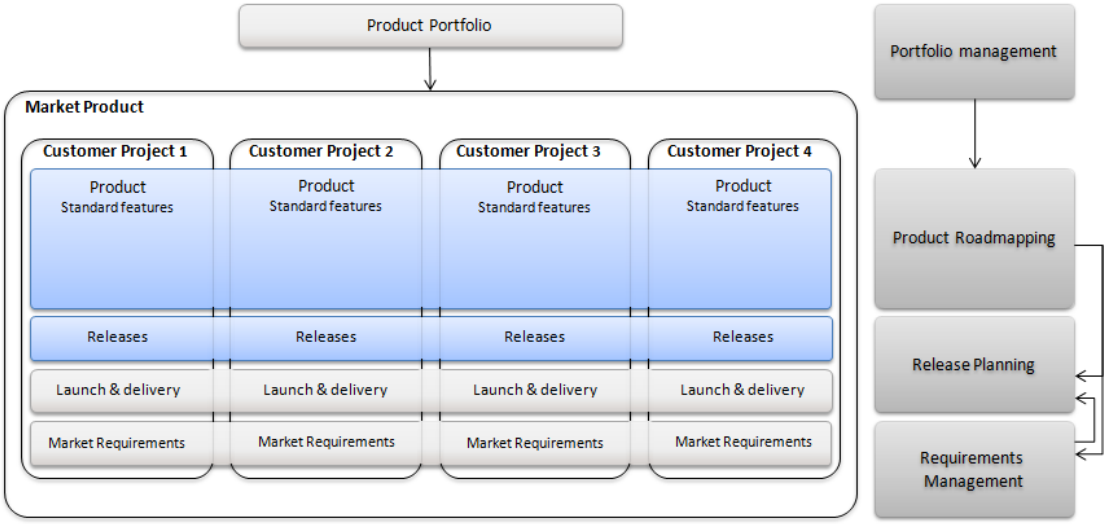


Figure 13: Stage 6b – Standard software product.

Definition of Standard Software Product: *a packaged configuration of software components, which is released for and traded in a specific market*. This definition is based on the definition for a software product from Xu & Brinkkemper (2005).

Characteristics:

- *One generic product for all customers and build for a specific market*
- *Structured releases*
- *Software is completely configurable*
- *Software business aiming at selling licenses*

General

After several stages still focusing on the customers, this stage only focuses on a market in general. By performing active marketing and sales activities, the company should start to sell the product to a mass-market and start looking at the wishes of that market. In order to bring the product to the market, it is required that there are no customized features included within the product and the product is completely configurable. Furthermore, the main measurement in order to determine success is mainly based on getting a bigger market share and having a shorter time-to-market. Still the customer satisfaction gives a proper indication whether or not the entire transformation is a success or not, but this is not the main success indicator anymore. An advantage of selling product software is that no additional development is required in order to be able to sell the product to new customers. This also results in a much bigger potential market and more customers (Hietala et al. 2004).

Requirements Management & Release Planning

Compared to the previous stage, this stage consists of the development of a product for an entire market segment. As a result, the release planning process should serve the identified market frequently with new releases. The requirement management process must be changed from gathering requirements from specific customers to gathering requirements from the entire market segment. Additionally, the requirements gathering process is driven by a market pull and technology push approach. Due to the release planning function, the main focus when prioritizing and selecting requirements is changed to create an optimal (subset) of requirements in order to increase market-share. While determining this subset of requirements it is essential to consider the available resources. At this stage the product is launched & delivered per market and the product is delivered to a bigger amount of customers compared with stage 6a. All of the customers within this stage get

exact the same product without any customizable features at all. Some examples of this kind of products are: operating systems or business software. Main objective of this type of software product is selling as many licenses as possible.

4.2 Validation

For the validation of the productization process we interviewed a number of experts, we created a survey for the participants for a SPM course and we performed a case study at MP Objects. The results of the case study can be found in section 7, and the actual validation results in section 7.6. We elaborate more on the objectives and the results of the expert validation and the survey within this section.

4.2.1 Objective

The objective of the expert validation and the survey is multifold. What we tried to achieve with the (practical) expert interviews, is mainly focused on the acceptance and recognition of such a process in the past of organizations. Key questions are related to how companies actually transformed in order to become a software product company. We also considered what order is followed to implement the product management functions and whether firms had skipped stages during their transformation. Additionally, for the scientific field we tried to validate whether or not the productization process is technically sound in relation with the viewpoints of the SPM experts and the reference framework for SPM.

Alternatively, we created a survey which is carried out by the participants of a Software Product Management Course to validate the readability, understandability, and applicability of the productization process in a wide range of companies. The situational factor questions are used in order to determine the variation of companies. An important result of the survey is whether or not the participants are able to empathize in such process and the possibility to select a stage(s) which is applicable for their product(s).

4.2.2 Expert panel

The validation of the productization process is performed by interviewing a number of experts from the scientific field and practical field.

Participant	Organization	Remarks
Rudy Katchow	Credit Tools	Currently Product Manager, former student at Utrecht University and also studied the SPM reference framework.
Marc Remmers	BusinessBase	Product Manager and participant of a SPM course
Martin van Mierloo	GX	Product Manager and participant of a SPM course
Inge van de Weerd	Utrecht University	Doctor at Utrecht University and founder of the SPM reference framework
Sjaak Brinkkemper	Utrecht University	Professor, doctor at Utrecht University and co-founder of the SPM reference framework

Table 5: Interviewed experts from scientific field and practical field.

4.2.2.1 Clarification

As a result of the expert validation, several changes are applied to the initially designed productization process. We clarified the stages textually and visually. For each of the stages we created several bullet points which describe the models and its characteristics in order to clarify the stages. Additionally, we also identified concrete names for each of the stages within the productization process. By using the characteristics we were able to determine accurate and

representative names. We also added a list with definitions, in order to avoid misconceptions about the terminology which we use in the models and in the associated descriptions of each stage. We also changed several visual aspects of the models in order to increase the readability of the productization process. For example, we removed the abbreviations of a couple of terms we used. By using the full name the readability increases and then there are no misconceptions about the terminology.

4.2.2.2 End stages

Initially, we defined one end stage of the productization process, when product software management is fully applied and the product is fully standardized. However, as a result of the interviews we identified the possibility that an organization can also be product software oriented when still a small customized layer is added to the software. This also implies the possibility to attach new (customized) functionalities to the product software so that it can be integrated within existing environments. We verified this suggestion by looking at the available literature. As a result, also Cusumano (2004) identified the division and identified it as a "Hybrid approach". This approach is the combination of selling products and coordinated support services. Additionally, also Hoch et al. (1999) identified the division of product software designed for a mass market ("Packaged software") and product software which needs customization ("Enterprise solutions") (illustrated in Figure 14). They studied the degree of productization which ranges from the packaged software product delivered "as is", to customer specific software.

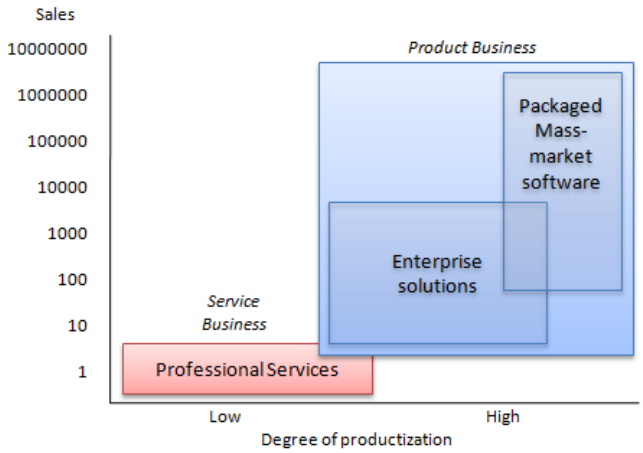


Figure 14: Software Product and Service Business (adopted from Hoch et al., 1999)

We changed the end stage productization process into two possible stages (6a & 6b) due to the results of the interviews and the recognition in the literature. The main difference between these two possible end stages is based on a different approach for selling the product and providing a service. As a result of the introduction of the two new possible end stages, the experts from the other interviews also identified the option to switch between these two end stages. However, in order to be able to apply this possibility we need to look deeper into the relevant factors which are involved in this process. For example the scope and domain of the software can negatively influence this process.

Additionally, the product manager from Credit tool identified the possibility to move backwards in the productization process. There is always a possibility that an organization decides to change back to a customer specific focus. The decision of changing back to developing customized software is also studied by Cusumano (2008) and is identified as "Servitizing products". Eventually, markets for software products can ultimately become saturated and in order to maintain revenues, organization might have to invert new products or start switching to services. However, the scope of this research focuses on the transformation towards becoming a product software company. Consequently, we are

not looking deeper into the possibility of changing back to a service business. Future research should determine the possibility to actually transform between the two end stages and the possibility to change back.

4.2.2.3 Applicability

All product managers were able to select a stage which was applicable for their organization. Moreover, all product managers also recognized earlier stages of the productization process in the history of their organization. For example for GX a clear transformation from stage 4 to stage 6a was recognized. However, during this process they sort of skipped stage 5 due to a high commitment of management, a clear vision and strategy, and a degraded customer focus. On the other hand, when we look at BusinessBase, the product manager recognized that their legacy system is transformed until stage 5. In order to continue with the transformation of becoming market driven, they completely redesigned their product and developed it based on another programming language. As a result of that, their new product is positioned in stage 6a.

4.2.2.4 Merging stages

During the evaluation of the productization process, several experts identified a need to combine stages. The added value of stage 5 was considered by some experts. The product manager of GX for example pointed out that they skipped stage 5 due to a high commitment of management and realizing that customers might leave. For GX, the main strategy was to create a software product as soon as possible. They rather focused on new (potential customer) instead of their current customers. In contrast, as mentioned earlier, the product manager of BusinessBase recognized that their legacy system is positioned in stage 5, which indicates on the need to keep this stage.

In our opinion when stage 5 is left out, the gap between stage 4 and stage 6a or 6b is too large. Customer satisfaction is one of our main concerns while changing from developing customer specific software to a standard software product. We think that during the productization process, an organization should first focus on creating a product platform (stage 4) and still provide customer-driven maintenance activities. Additionally, the next point of attention is that an organization should introduce releases. This implies that the build-to-order approach changes into a more (long-term) organized development. Main difference when implementing releases is that customers might have the desire to have a specific requirements or requests implemented. But what if a company decides to develop that specific functionality later on in the lifecycle of the product? We expect that there is a possibility that customers might not agree on this approach. Using releases should therefore be subtly introduced in an organization so that also the customers are able to get acquainted with this approach. The reason for this is that the release planning function is a very complex problem which includes different stakeholder perspectives, competing objectives and different types of constraints (Ruhe, 2005). Consequently, we use stage 5 for the introduction of releases and still satisfy the customers by providing the possibility to develop customized features. Figure 15 represents the implementation of the releases to change the release time and still keep the customer satisfied.

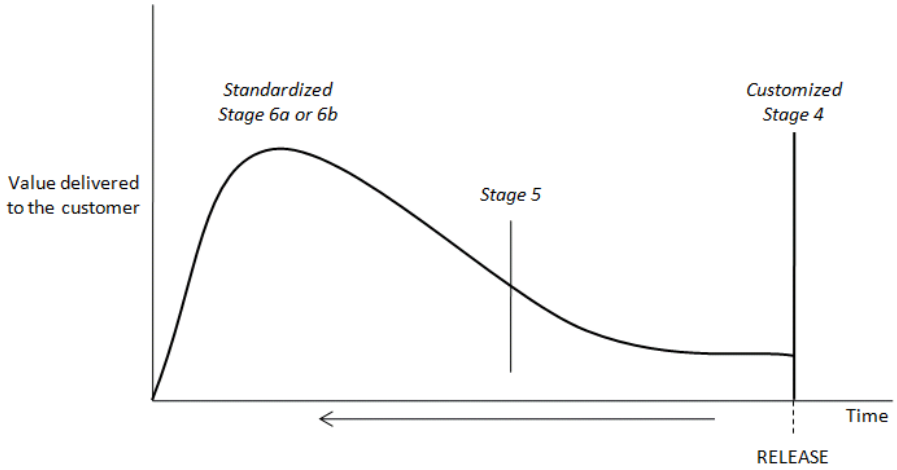


Figure 15: Change in delivery of releases, adopted from Koponen (2008).

However, this does not imply that an organization must follow stage 5. When a company's strategy is to release new products as soon as possible, there is a possibility to skip stage 5. As a result, this might have a negative effect on the customer satisfaction. Future research should be carried out in order to determine the actual implications of leaving out stage 5.

Finally, the product manager from Credit Tool also suggested merging the first three stages into one stage which generally describes a customized software development situation. However, due to illustration purposes we did not applied this aggregation. The productization process presented within this research shows a clear evolution of the software.

4.2.2.5 Integration of the reference framework

Initially, we presented a specific order for adopting the product management functions. However, an explicit approach to apply these functions can not be identified because each organization desires to use another approach for implementing the product management functions. The experts indicated during the interviews that another approach to realize the specific product management functions can also be used and that it is not possible to specify a fixed sequence for implementing the functions. As well as a top-down approach as bottom-up approach were mentioned during the validation with the product managers. Therefore, the integration of the right size of the stages (reference framework for SPM) was reconsidered. As a result, we changed the visualization of the right size of the stages in such a way that an increase in maturity of the product management functions is visualized. By using this approach, organizations are able to determine their own adoption strategy of the reference framework.

4.2.3 Survey

In addition to the expert interviews we also created a small survey which was filled in by ten of the (fourteen) participants of the Software Product Management course given at the Utrecht University. This course provides the product managers (participants) with more insight information about the software product management activities and involved stakeholders. This survey is filled in by ten respondents of eight different companies. The size of these companies varies from 10 till 130 employees and one outlier of 350 employees (Table 6).

Company	**1**	**1**	**2**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Business unit size (fte)	135	100	70	60	35	350	10	18	80	20
Customer satisfaction	7	7	6	8	7	7	9	6	7,8	7
Potential market size	500-1500	500-1500	0-500	500-1500	3500 +	3500 +	0-500	0-500	3500 +	1500-3500
Release frequency (days)	31	365	180	183	500	40	100	30[1], 90[3]	120[2] 450[3]	30
New req rate	300	50	12	100	200	300	100	400	600	300
Lifetime (years)	5	10	3	8	5	3-7	10	>4	3	5
Customer Involvement	M	M	M	L	L	L	H	L	M	H
Legislation	Loose	Loose	Loose	Loose	Loose	Strict	Loose	Strict	Loose	Non-existing
Applicable stage	6b	6b	4	6a	6a	6a	4,5,6a,6b	6a	6a	6b

Table 6: Companies which filled in the survey and their applicable stage(s).
[1] Bug fixes; [2] Minor releases; [3] Major releases.

We introduced the entire productization process to all the participants and afterwards we asked which of the stages is the best applicable for their company. The results of that were quite surprising:

- **Combining stages:** One respondent suggested combining stages 2 & 3 and 4 & 5 due to the possibility that companies might have the required knowledge in house to perform such change. We can imagine that smaller companies with one or two product and a small amount of customers can combine steps in order to become market driven.
- **Possibility to go back:** Another participant stated that the product which they developed was positioned in stage 6a and they deliberately changed back to stage 4. At that moment they modified the product for one big customer and based on the success of that system they restart transforming to stage 6a.
- **Differentiation:** Two respondents of one company picked both a different stage (4 and 6a) while they both aimed at the same product. Additionally, it is also strange that both respondents gave different answer for the situational factor questions. This can indicate that one of the respondents is not fully aware of the current position or that there are too many similarities between stage 4 and 6a.
- **Multiple stages:** One of the respondents picked four different stages for the four products which they develop, namely stage 4, 5, 6a and 6b. When we look at this particular company we saw that they provide a CRM module, planning module, a service module designed for PDA's & laptops and an administrative & reporting module. It is therefore understandable that these modules each are differently managed.
- **Applicability:** All respondents of the survey were able to depict a reprehensive stage which is applicable for their software and a remarkable seven of the ten respondents have already a product which is positioned with stage 6a or 6b.

Consequently, we are pleased with the results of the survey, but we did expect that the respondents would elaborate more on the productization process itself. Quite regularly only brief answers were given, but a result of that we can conclude that the entire process is clear.

5 Productization approach

At a certain moment organizations starts to recognize a need to start developing a standard software product. This section determines from which position companies start and how large the difference is with the best suitable end situation. The following steps should be followed when a company is willing to transform into a product software company (also illustrated in Figure 16):

1 The first step of the transformation is the determination of the *initial position* and this activity begins with carrying out the Software Product Management assessment. The assessment consists of three major parts; the first part contains questions related to the situational factors. The second part contains questions for the determination of the maturity levels of (possibly present) SPM processes. Finally, the last part consists of the criteria for determining the initial position within the transformation process.

2 A *gap analysis* must identify the distance from the initial position until to being fully Software Product Management oriented. We use the situational factors (Bekkers et al., 2008a) to determine the best suitable maturity levels for the product management functions. Based on the initial maturity levels, it should be determined which processes need to be implemented or improved. In addition, we also use the Process Deliverable Diagrams for the determination of the organizational differences.

3 Finally, the *recommendations* can be used by an organization so that they continue in transforming to become market-driven. During this step, also the guidelines (section 6) can be used for the implementation of processes which are not in place.

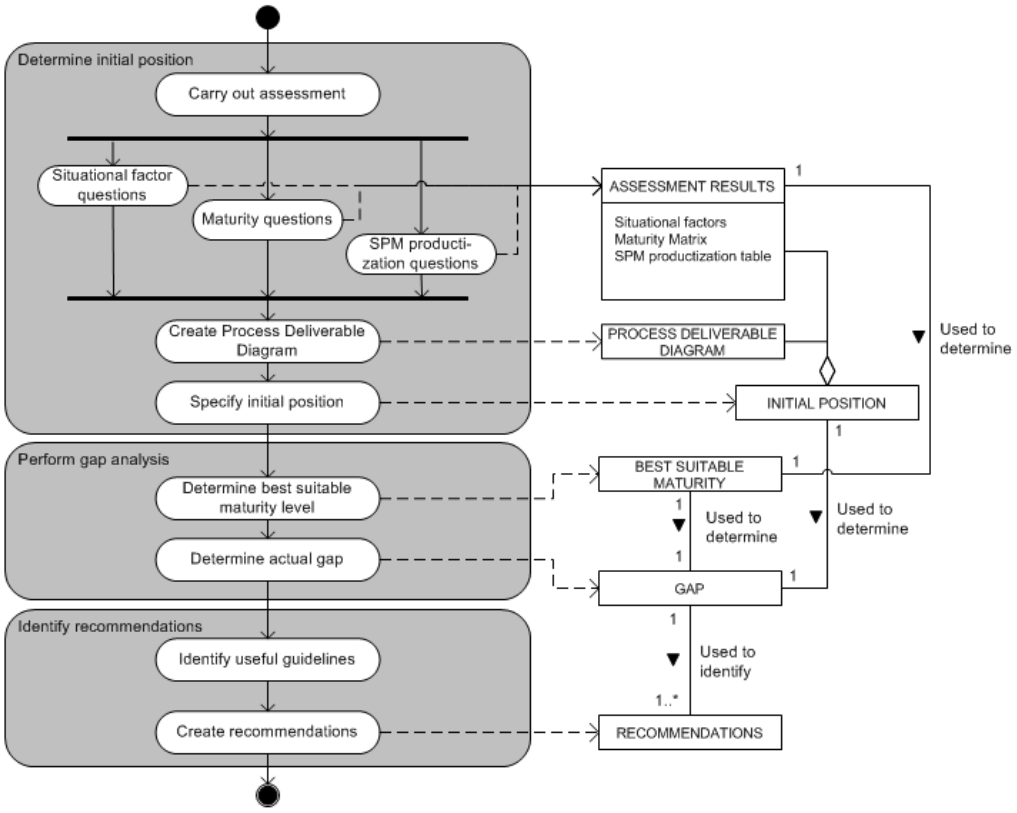


Figure 16: Process Deliverable Diagram for application productization process

5.1 Initial position

The determination of the initial position is determined by carrying out the assessment and by creating a Process Deliverable Diagram (PDD). More information on both of the approaches can be found in this section. We elaborate more on applying this approach within the business case which can be found in section 7.3.

5.1.1 Assessment

The assessment which was carried out can be found in Appendix D: Assessment and it is a follow-up of the assessment created by Weerd et al. (2009), which is designed to identify the maturity of the SPM processes within an organization. The assessment consists of a couple of general questions, situational factor questions, maturity questions and the maturity matrix. The final part is related to the determination of the initial position when a customer-driven company is starting to transform to develop a software product. This additional part is used in order to be able to determine the initial position of an organization.

Situational factors

The questions related to the situational factors describe the context and the environment of an organization. The factors are divided into five categories: 'business unit characteristics', 'customer characteristics', 'market characteristics', 'product characteristics', and 'stakeholder involvement'. The situational factors are especially designed for small and medium sized companies so that they can improve their SPM processes in a more optimal manner. The list of situational factors gives the influence weight of each factor on the SPM processes so that it is possible to determine which level of maturity is best suited.

Maturity questions

As a result of the maturity questions from the assessment, the SPM Maturity Matrix can be filled in. The maturity matrix (Appendix D: Assessment, III. Maturity questions) has got thirteen columns with maturity levels from 0 to 12, where 0 represents the lowest maturity level and 12 the highest level. The rows are equal to all processes within the SPM reference framework divided into the four product management functions: Requirements Management (RM); Release Planning (RP); Product Roadmapping (PR); and Portfolio Management (PM). The maturity levels are marked gray and show in what extent a specific process is implemented. Within the maturity matrix there are three different maturity levels: 1) General Maturity Level (GML), which consists of the overall maturity level of all four product management functions; 2) (Product Management) Area Maturity Level (AML), which is related to each of the four product management functions; 3) Process Maturity Level (PML), is the maturity of each process within a function. For each of these levels holds that the lowest maturity score shows the actual maturity level.

SPM Productization

The last part of the assessment consists of some criteria in order to determine at which stage an organization is initially positioned. The entire transformation is based on seven different stages and transformation steps in-between the stages. In order to determine at which stage a company is positioned, we generated a table with criteria (Appendix D: Assessment, IV SPM Productization). Each step consists of some criteria which determine the initial stage. As a result, of the designed models we identified six criteria areas which characterize the different stages. The identified criteria areas are 'Software', 'Sequel', 'Requirements origin', 'Requirements selection', 'Main stakeholder involvement on development' and 'Objective' (Table 7). The content of this table is determined by looking at the differences between developing software for specific customers and developing product software for an entire market (section 3.4). Based on the differences and the designed models we specified the criteria between these two extreme values.

Area	Customized software	Standardized software
Type of products 'Software'	New system project; maintenance	New products; new versions
Lifecycle of products 'Sequel'	One release, then maintenance	Several releases, depended on market demand
Requirements origin 'Requirements origin'	Elicited, analyzed, validated (identified per customer)	Invented by market pull or technology push (based on the entire market)
Requirements selection 'Requirements selection'	More or less fixed set of requirements	Optical selected subset of requirements
Main stakeholders 'Main stakeholder involvement on development'	Customer organization (external)	Developing organization (internal)
Measure of success 'Objective'	Satisfaction, acceptance, ROI	Sales, market share, good product reviews

Table 7: Differences used for determining criteria, adopted from Table 4.

During the assessment, an organization should encircle or select the applicable values based on their situation. The initial position can be determined by taking the best suitable (average) step based on the given answers. The initial stage is the start point of the transformation and it continues until stage 6a or 6b is reached and all Software Product Management processes are properly implemented.

5.1.2 Process Deliverable Diagram

The second approach we use in order to determine the initial position of the transformation is the development of a Process Deliverable Diagram (PDD). Weerd & Brinkkemper (2007) have described a technique which can be used to model activities and artifacts of processes. This PDD consists of two meta-models: 1) a meta-process model (left side of the PDD model), in which all the processes can be viewed; and 2) a meta-deliverable model (right side of the PDD model) in which all the deliverables can be depicted. The two models are based on the UML activity diagram and the UML class diagram (Weerd & Brinkkemper, 2007). The PDD is used in order to support the determination of the initial position for the transformation and to determine the organizational gap/difference between the initial situation and the fully adopted product management situation.

5.2 Gap analysis

After the determination of the initial position, we use a gap analysis in order to identify the distance from the initial position until the end situation of being a product software business. However, first we must determine at which maturity level the product management processes within an organization should operate. This decision is based on the answers of several situational questions. With the results of these questions we can calculate the best suitable maturity levels of each of the product management functions. Finally, the gap analysis provides several recommendations which will be suggested to a company's board and after that the board can decide which recommendations will be applied. We elaborate more on applying this approach within the business case which can be found in section 7.

5.2.1 Situational Factors

The best suitable maturity levels can be calculated by using the some of the situational factors defined by Bekkers et al. (2008a). Based on these factors a company is able to determine how mature specific product management function should be. Research of Bekkers (2008b) consisted of the identification and validation of a list with situational factors. For each of the factors Bekkers identified the weight factor for each different product management functions (PM, PR, RM, RP). For each of these functions we defined two situational factors based on their influence weights

(Appendix B: Situational factors, Table 20). In this table we identified PM', PR', RP', and RM' these values show the difference with the highest factor weight. The last column shows the difference with the second value and based on this we can see how much a factor specific contributes to an explicit product management function.

The first factor we identified is 'Product lifetime' because this factor has got the biggest influence on PM (4.55) as well as PR (4.36) when we compare them with the other factors. Second factor we use for PM is 'Market size', because this factor has a significant influence on only PM (difference of at least 0.63 with other factors) and the second factor we use for PR is 'Customer satisfaction'. The difference with the other factors is not that big, but we choose this situational factor because it has the second highest weight (4.00) on product roadmapping. For RP we use 'Legislation' (0.73) and 'Release frequency' (0.54) because these two factors have a big influence on the RP processes in combination with the biggest difference compared with the other functions. Finally for RM we use the factor 'New requirement rate' (0.37) and 'Customer involvement' (0.27) because also these factors have the highest weight in combination with the biggest difference compared with the other product management functions. An additional elaboration on the influence of each factor can be found in (Appendix B: Situational factors). In order to decide the importance of the chosen situational factors we used the values which are adopted from Bekkers et al. (2008c) technical paper as *reference values* (Table 8). This paper consists of 14 cases studies on the situational factors. By using the variety of the values, the relevance of each factor can be determined.

Factor	**Value ranges (Bekkers et al., 2008c)**	**MP Objects' value**
PM_F1. Product lifetime PM_F2. Market size	Range from 1 to 15 0-500 (8x), 500-1500 (1x), 1500-3500 (1x), 3500+ (3x) one outlier	> 15 1500-3500
PR_F1. Product lifetime PR_F2. Customer satisfaction	Range from 1 to 15 Range from 6 to 8	> 15 8
RP_F1. Legislation RP_F2. Release frequency	Strict (7x), Loose (7x), Non-exiting (0x) Range from 7 to 180 days, and one outlier: 365 days	Loose 90
RM_F1. New requirements rate RM_F2. Customer involvement	Range from 2 to 500, and one outlier: 1500 Low (1x), Medium (6x), High (6x), and one combination (M/H)	250 High

Table 8: Reference values adopted from Bekkers et al. (2008c)

With these values the gap analysis can be carried out in such a way that an organization is able to determine the intensity of implementing specific product management functions. An example on how to apply this is given in the business case (section 7). A drawback of this approach is that reliability of this approach should be considered. Therefore, in order to give suitable recommendations we also use the results of the maturity matrix and the PDD.

5.2.2 Maturity Matrix

In addition to the determined situational factors, we also use the SPM Maturity Matrix in order to determine the essence of improving / implementing specific product management functions. Within this matrix the results of a company are marked gray and in addition to that, the red line in this matrix shows the results that Weerd obtained from an Internet survey among product managers in January & February 2009. In this survey, product managers were asked to indicate which capabilities they carried out in their own organization. We use this line as a reference point in order to determine the essence of improving specific processes. This red *reference line* is calculated by looking how often each capability was implemented within the companies. Weerd et al. took the mode score for each process (mode score is the highest value that occurs the most frequently). The maturity levels of the business case should be compared so that can be determined whether or not it is likely.

5.2.3 Process Deliverable Diagram

Where the previous approaches focused especially on the presence of specific product management processes, we use the PDD during the gap analysis to determine the hierarchical and organizational relation between the processes. There is a possibility that specific processes are present but, because that software is developed for a specific customer, the hierarchy is not as defined within the reference framework for SPM. Furthermore the PDD also shows the deliverables of the current situation. Based on this we can decide which organizational changes are required so that the product management processes are properly in place. So basically, we are able to support the other part of the gap analysis and identify recommendations for the resulting steps of the productization process.

5.3 Recommendations

After performing the gap analysis we provide several recommendations which an organization can adopt in order to become fully software product management oriented. We use the models of the productization process in order to determine remaining activities of the resulting phases to transform into a product software business. Additionally, we can use the generated implementation guidelines for the realization of the improvements. Finally, the board of an organization can decide which recommendations they apply in order to become market driven.

6 Guidelines

In order to implement the reference framework for Software Product Management within an organization, it is valuable to apply guidelines so that a positive result is guaranteed. There is little literature available on the specific implementation of product management. Therefore, we identified a number of guidelines for implementing the product management functions within an organization. The definition which we used for a guideline is: that a guideline "aims to streamline particular processes according to a set routine and is never mandatory". Important while defining the guidelines is the effect of guidelines, they can be both favorable and restrictive. Guidelines which are too rigid and strict could have counter unhelpful effects (Bohl et al., 2002). We created an initial list with guidelines based on available literature. Research fields which we studied are related to software product management, the product management functions, implementation hierarchy, and research regarding software process improvement. After that, we validated the guidelines by interviewing experts which are known with the reference framework for SPM. We elaborated on retrieved feedback and we made several adjustments. Consequently, the final guidelines for the implementation of the SPM processes are presented within the section.

Before we start with the elaboration of the determination of the implementation guidelines we first look at the definitions and characteristics (Table 9) for the used terminology.

- Policy: "A plan or course of action, as of a government, political party, or business, intended to influence and determine decisions, actions, and other matters"[1]
- Procedure: "An act or a manner of proceeding in any action or process"[2]

	Policy	**Procedure**
Goal	Guiding principle used to set direction.	Particular way of accomplishing something.
Structure	Course of action to guide and influence decisions.	Steps to be followed as a consistent and repetitive approach or cycle to accomplish an end result.
Scope	Serve only as a guide, therefore the scope is limited.	Determine a sequence of activities to do particular work, therefore a broad scope.

Table 9: Characteristics of the used terminology (Singla, 2006).

6.1 General

This section consists of the identification of general guidelines for the implementation of the SPM processes. The general guidelines are related to the function of Product Manager and the actual implementation of the processes.

Product Manager

When a company starts to implement the processes of the reference framework of SPM, they also have to consider the introduction of the 'Product Manager' (PMngr) role. References we used in order to determine the guidelines for the implementation of the product manager function are: 1) Dver (2003) and 2) Ebert (2007), the actual results can be found within Appendix C: Guidelines from literature, I. General.

For the determination of the guideline for the implementation of the PMngr function (G1) we used the results identified by Dver (2003): Understanding the product is one of the major responsibilities of the PMngr (1a), this is important because he/she is also accountable for the overall decisions of the product (1b). Other responsibilities of the PMngr are managing the portfolio and identifying potential partner relationships (1f). Parts of the portfolio are related to the product lifecycle management and product line identification (1c). The PMngr is also accountable for management of

[1] http://dictionary.reference.com/browse/policy
[2] http://dictionary.reference.com/browse/procedure

the requirements and this involves the specification of the market requirements (1d) and product requirements (2a). Finally, the last area in which the PMngr is responsible is the release planning function. Within this function the PMngr is responsible for the creation of the release definition (2a). In general, the PMngr is accountable for the strategy definition and operational execution (2b).

We did not used 1e, because the PMngr is not specifically responsible for bringing the product to the market and creating the go-to-market plan, this is the responsibility of the product marketing manager or the (internal stakeholder) sales & marketing department. Based on the found literature we identified the following guideline for the introduction of the PMngr function:

G1. Introduce the Product Manager function to the company, and assign this function.
In order to become market-driven, the function of Product Manager (PMngr) should be introduced. The PMngr is accountable for the overall product decisions (in terms of strategic and operational decisions) and the PMngr is the first one to know when there is a problem. Some of the responsibilities of the PMngr function are: identifying market needs, managing the entire product line / life cycle, preparing and implementing a business case, managing requirements, writing specifications, and monitoring development projects.

Software Process Improvement
When we implement specific processes within an organization this requires a specific approach and a certain attitude of the staff members. Therefore we also looked at the success factors of Software Process Improvement (SPI). We redefined some of the results so that they are applicable for an iterative implementation process for the product management functions. References we applied in order to identify the guidelines for improving software processes are: 1) Niazi et al. (2005) and 2) Stelzer & Mellis (1999), which can be found within Appendix C: Guidelines from literature, I. General.

In order to use the phases from Niazi et al. (2005) for the identification of the guidelines, we combined the following phases 'Pilot implementation' and 'SPI implementation action plan' into 'Planning' (as described in G2). We combined them because the phases from Niazi are related to SPI across the entire organization. For an iterative process improvement approach it is too complex to perform first a pilot implementation and design a SPI implementation plan on the result of the pilot. The other phases from Niazi are renamed so that they are better suited for the SPI of the product management processes. Research carried out by Stelzer & Mellis (1999) also refers to the SPI across the entire organization; therefore we are not using the factors 'Tailoring improvement initiatives', 'Change agents and opinion leaders', and 'Unfreezing the organization' for guideline G3. The commitment of senior management, involvement of staff members (Stelzer & Mellis, 1999) and awareness of the organizational changes (Niazi et al., 2005) is important for each iterative implementation process (G3). Especially, the awareness is important for every phase of the SPI approach and we therefore created a separate guideline (G3) for this rule. As a result of the used literature, we identified the following guidelines for the implementation process of each iteration:

G2. Each iterative implementation process should follow a controlled approach.
All involved stakeholders should understand what the consequences are when organizational changes take place, because the implementation is a comprehensive process. For the implementation of the product management function, the following approach can be used:

- Learning: Initially, managers and employees should completely understand of the necessary details and they also should understand how the implementation of specific processes adds value to the organization. This involves providing the participants with the required knowledge.
- Planning: Before the implementation of a SPM process area it is important that a plan or some kind of roadmap is developed. This plan should also consist of relevant and realistic objectives which must be achieved at the end of the implementation process.

- Implementing: After the preparations, the changes need to be applied within an organization. Management during the implementation of the process improvement practices is essential; otherwise it can lead to ad hoc, often inefficient, and sometimes chaotic practices.
- Maintaining: The organization must continually support maintenance and improvement of the implemented practices at a local level after the implementation. This will prevent that a changed process slides back to the old situation.

G3. Make sure that senior management is committed and employees are involved during the implementation of a SPM function.
Senior management must be committed and supportive before, while, and after the realization of a product management functions. They must encourage staff members of different teams and departments to communicate and collaborate in order to prevent misunderstandings and to create a coherent organizational culture.

6.2 Requirements Management

The requirements management function must be related to market-driven software development, this means that all requirements from a broader domain should be collected and not elicited from one specific customer. These identified requirements must be translated into product requirements so that they are described in the company's perspective and context. References we used for determining the guidelines for requirements management are: 1) Sawyer et al. (1999), 2) Natt och Dag et al. (2004), and 3) Jiao & Chen (2007), which can be found within Appendix C: Guidelines from literature, II. Requirements Management.

According to Jiao and Chen, requirements should be gathered from a number of stakeholders (3a), it should be made sure that all stakeholders are involved properly (RM1), so that only the real needs of the customers / market are gathered (3d). Additionally, all stakeholders should be able to fill in their need (by prioritizing) to have a specific requirement implemented (3f). For managing and storing the requirements (RM2) we identified the desire to store all the requirements on one location (2d). Possible storage techniques which can be used are a database (1a) or a requirements management tool (2d). For both of these techniques it is necessary to identify all requirements uniquely (1b). Furthermore, requirement management policies must be defined for the identifying, storing, and managing the customer requirements and product requirements (1c & 2e). A specific part of the policies must ensure that the requirements are documented properly (3e) and that also the rejected customer requirements are stored (1e). Ideally, there are also traceability policies defined (RM3) (1d). These policies should ensure that there is a distinction and a link between customer requirements and product requirements (2a & 2b) so that a bi-directional traceability is in place. Also Jiao and Chen identified the importance of the differentiation between requirements described in a customer's domain and described in a functional domain (3b & 3c). After identifying the product requirements it is vital to organize them so that dependencies and traceability are specified.

We did not apply 3g (Classifying requirements), due to the fact that the requirements are not specially classified to help the designers. The requirements must be organized in relation with the themes and core asset information. As a result we identified the following guidelines:

RM1. Create a procedure for retrieving the wishes, requests, and bug fixes from all the involved stakeholders.
The company must implement a procedure to retrieve the requirements from especially the involved external stakeholders, but also the internal stakeholders. Some kind of procedure should be defined or a system should be used to be able to retrieve all requests and wishes. Specify which stakeholders

are involved in the requirements gathering process and make sure that they are properly participating in the gathering process.

RM2. Define policies for requirements management and therefore storage of the requirements.
All gathered customer requirements and indentified product requirements need to be documented at one central place (database or requirement management tool). The storage should be designed in such a way that storing market requirements and product requirements is preformed separately. However, there must be the possibility to link them with each other, so that all requirements are easily traceable. Use a standard (template) to document (used and unused) requirements and describe them clearly, because often they are poorly understood and expressed in abstract, fuzzy or conceptual terms.

RM3. Design a procedure for transforming the customers' requests into product requirements and define traceability policies for organizing these requirements.
Convert market requirements into business understandable requirements by identifying product requirements. An organization should derive explicit product requirements that can be understood by marketing and engineering departments. The involvement of the internal stakeholder is essential during this process, because they should support the identification of product requirements. Furthermore, the market requirements need to be connected with all the related product requirements, so that the dependencies of implementing a specific requirement and the traceability of the requirements are known. The traceability of the requirements should be described within these policies so that a bi-directional traceability is maintained.

6.3 Release planning

In terms of the SPM reference framework the release planning activities must be related to the process of making software releases available to its users or market. This process should consist of the prioritization and selection of the requirements, creating and validating a release definition and (if necessary) managing scope change. The references we used for identifying the guidelines for the release planning functions are 1) Berander & Andrews (2005), 2) Ruhe (2005), 3) Ruhe & Saliu (2005), 4) Sawyer et al. (1999), and 5) Momoh & Ruhe (2006), which can be found within Guidelines from literature, III. Release Planning.

The prioritization of the requirements (RP1) should be designed so that the optimal set of requirements for a new release is determined based on the added value and the lowest cost (1a). During the prioritization the product manager should also take into account the existing dependencies between features (3d) and try to satisfy stakeholders (5b). Possible approaches which can be adopted are: value based (concerning the impact) or urgency based (concerning the time-to-market) (2a). Furthermore, the selection of the requirements (RM2) should focus on selecting an optimal set of requirements (1b) by looking at the added value, costs and risks. A subset of the requirements must be defined in order to minimize rework and schedule slippage (1d). Main constraints such as schedule, budget, resources (3c), time-to-market, and quality should be considered when selecting the set of requirements for a release (1f & 5c). Additionally, the satisfaction of the customers can be taken into account during the selection (1e); however this is not the main concern. According to Sawyer, to prevent requirement change management, a company should identify global system requirements while selecting the requirements (4a). As a result, it finds requirements likely to be most expensive to change. Furthermore, an organization should also identify unstable and unpredictable requirements in order to simplify requirements change management (4c).

After the selection of the requirements, a release definition should be created and validated (RP3). Due to a changing environment there is a possibility that the release plans need to be updated

frequently (2b). Therefore, a procedure and standard (template) for the development of the release plans improves the maintainability of the release definition. While creating the release definition a roadmap can be used in order to determine the best possible blend of features (3a & 5a). The development and the validation of the release plans should be executed in collaboration with the involved internal and external stakeholders (2c & 5d). A specific approach for the communication (RP3) with the stakeholders should be used so that all stakeholders are properly involved with the development of the releases and when the content or schedules change this should be reported to the stakeholders (5e). In addition, when the scope of a release changes (RP4), it is useful to follow a specific approach, so that specific actions can be executed when a scope change occurs (4b). Finally, when the development of a release is ready, a special set of instructions to launch the release should start, so that all internal and external stakeholders are informed and involved regarding the delivery of a new release. The result can be a launch preparation package or a specific procedure for the implementation of the product software for each customer (when specific integration is required).

We did not apply 1c (Estimate expected customer satisfaction), because the main focus while prioritizing and selecting the content of the new release should be on the added value, costs and risks. Expected customer satisfaction is not the main concern. We also did not use 3b (Satisfy the most important stakeholders), because the main idea of the release planning is selecting the optimal set of requirement with the biggest added value and not satisfying the most important stakeholders. Customer satisfaction can be taken into account, but is definitely not the main driver. As a result of found literature, we identified the following guidelines for implementing the release planning:

RP1. Create requirements prioritization procedure or adopt a specific prioritization technique.
Choose between prioritizing the requirements based on their value or urgency. For a value based approach, an organization should prioritize each requirement and balance the business benefit of each requirement against its cost. On the other hand, when an organization uses the urgency based approach they should prioritize requirements based on market or customer needs in order to address a shorter time-to-market. While prioritizing, also existing dependencies between requirements should be considered.

RP2. Define a requirements selection procedure or adopt a specific selection method.
Plan and select an ordered and optimal subset of requirements for a release. Main focus when selecting this subset of requirements is increasing market share and secondly try to satisfy customers. Strive to reduce rework and schedule slippage, so that the need for scope change is minimal. The selection procedure should be feasible with conflicting constraints such as schedule, budget, resources, time to market, and quality. In order to prevent scope change, the product manager should identify and reconsider global system requirements because these requirements are likely to be most expensive to change. Lastly, unstable or unpredictable requirements also need to be identified in order to simplify scope change management.

RP3. Generate procedures for developing, validating, and communicating release plans.
Before defining the procedures for the release plans, a standard (template) should be created and it should be designed in such a way that it can be updated frequently. The release plan should consist of a list of the prioritized and selected requirements that will be implemented, a time path, required resources and the needed capacity for developing the release. While creating the content of a release definition the product manager should try to provide maximum business value by offering the best possible blend of features in the right sequence of releases and base the content of the release on the roadmap. Also ensure that the stakeholders are involved when creating a release plan. This can be achieved by creating a communication procedure to involve all stakeholders and this procedure should keep track of the communication with the stakeholders when the release plan is updated or finished and finally for the validation of the release plan.

RP4. Define change management policies and define a launch procedure.
In order to cope with scope change, an organization should create scope change policies. These policies should describe what the consequences are, what actions must be executed and how stakeholders should be informed when the scope changes. Secondly, when a release is completed, a special launch activity should start, so that all internal and external stakeholders will be formally informed about the launch of a new release. The result can be a launch preparation package or a specific procedure (dependent on type of product) for the implementation of the product releases. This procedure depends on whether a release needs to be implemented for an existing customer or for a new customer (more complex integration).

6.4 Product Roadmapping

The main criterion for product roadmapping is that a roadmap should describe the planning and forecasting of features for future releases. The roadmap should fill up the gap between the defined strategic and organizational elements. References we used for the guidelines for product roadmapping function are: 1) Lehtola et al. (2007), 2) Lehtola et al. (2005), 3) Phaal et al. (2004), 4) Lee & Kang (2004), and 5) Ebert (2007), which can be found within Appendix C: Guidelines from literature, IV. Product Roadmapping.

In order to determine the directions of the products, the commonality and variability of the product should be identified (PR1) (4a). By specifying themes and core assets the future of the product can be determined. Additionally, the process of identifying themes and core asset information is an ongoing process and therefore policies for managing and storing the features should be created. As a result, the identified themes and core asset information is used to determine the future of the product (PR2) (1a). The roadmap is a long-term plan and this implies the necessity to focus on a shorter time period (2f) and to plan more releases in advance (1c). The roadmap shows when the planned functions / features arrive and what their dependencies are (5b). When designing the roadmap a company should create a balance between market pull and technology push (3a). Due to the frequent updating of the roadmap it is useful to create policies for documenting, storing and updating the roadmap. The roadmap is a tool which is used to communicate with the involved stakeholders and therefore it may not be outdated (2d). The development of policies for storing and managing the roadmap should avoid this. The main purpose of creating a product roadmap (PR3) is linking the business decisions (business side) with requirements engineering (operational side) (1b & 2a). Responsible for the development and maintenance of the roadmap is the product manager (1d). During the development of the roadmap the product manager should take different aspects into account (3b) such as the market and technology trends (3d), available resources (2e), scope, trends, strategy, and budget. Otherwise it can result in unreachable releases and an inadequate roadmap. As mentioned earlier, the roadmap is a tool which can be used to inform the involved stakeholders (2b). In order to keep the stakeholders informed a specific approach for communicating the roadmap should be considered. The roadmap must be designed by collaborating with the internal and external stakeholders, and a procedure for communicating the roadmap can support this process. In addition, this includes the external stakeholders and not only internal stakeholders, as described in 5c.

We did not apply 2c (Emphasizing developer viewpoint), because for the development of the product roadmap the viewpoint of all internal stakeholders is important and not only from the developers. Secondly, 3c (Performing market analysis) and 3e (Identification of technology) are also not used, because these are related to the portfolio management and particularly the identification of the market trends and technology trends. Thirdly, 4b (Assets design) is not applied, because this guideline focuses too much on the actual development of the assets. In the roadmapping function only the identification of the asset information is desired. Finally, we also did not use 5a (Technology roadmap), because a technology roadmap is a possible form of roadmap and it not necessarily has to be a

technology roadmap. As a result of the found literature we identified the following guidelines for implementing the product roadmapping processes:

PR1. Define policies for identifying and storing the themes and core asset information.
Specific ideas or requirements with a similar nature need to be wrapped up as themes and they should provide certain functionality of the product. In addition, the commonality and variability of a domain perspective should be recognized as reusable core assets. These identified themes and core asset information gives the roadmap and upcoming releases a clear direction. The development and maintenance of the roadmap is a continuing process and a roadmap should never get outdated. By creating a standard (template) and specifying policies for managing and storing the themes and core assets, this process can be maintained. Preferably the roadmaps are stored at one central place.

PR2. Identify features for the roadmap and create the roadmap.
For the identification of the features of the roadmap, a company should try to find a balance between a market pull and a technology push approach. The main purpose of creating the roadmap is to make the link of the defined strategy and the requirements management activities explicit. The roadmap is a plan which consists of the forecasts concerning time of implementation of specific trends, new products, product lines, resources or the entire company. While creating the roadmap consider: the level of available resources, nature of the issue being addressed, and the market and technology trends. By creating a roadmap, a long-term planning (three to five years) for the future releases is created which should be communicated regularly with the involved stakeholders and it therefore needs to be updated frequently.

PR3. Define policies for storing and procedures for communication the roadmap.
The development and maintenance of the roadmap is a continuing process, a roadmap should never get outdated. By creating a standard (template) and specifying policies for this process of designing and updating the roadmap should be simplified. Additionally, the roadmap should be stored at one central place, so that all staff members are able to reach and manage it.
One of the main purposes of using a roadmap is that it provides a layout of the upcoming product releases and it can be communicated with the stakeholders so that they are acquainted with the upcoming product changes. In order to communicate the roadmap with the stakeholders it is essential to create a specific procedure so that stakeholders are up-to-date about short-term and long-term releases. Additionally, the communication of the roadmap should be standardized by creating a template email for example.

6.5 Portfolio Management

Essential for the determination of PM guidelines is that they must be related to products (designed for a market), this because PM also can be used in project oriented (customer driven) software companies. In order to define the guidelines, the terminology PM must be related to decision making about the set of existing products; introducing new products by looking at (upcoming) trends and the product development strategy. The strategic decisions must be supported by specifying possible collaborations with partners, the product lifecycle and the product line (Weerd et al., 2006a). References we applied in order to identify the guidelines for improving software processes are: 1) Cooper et al. (2001), 2) Stark (2004), 3) Clements et al. (2005), and 4) Srinivasan (in press), which can be found within Appendix C: Guidelines from literature, V. Portfolio Management.

Before determining the strategic decisions first the business goal(s) should be determined (PM1). As a result, a specific approach should be selected to realize these goal(s) (2a). This involves the determination of the product line goals, scope and in which domain the products are positioned (3b & 3c). The strategic decisions should be based on relevant focus areas (1a). These areas outline a basis for the identification of specific market trends and technology trends (PM2). Based on the

identified trends a company can define the directions of the product (1b & 3d) which involve the product lines and the product lifecycles (PM3). The product lifecycles and product lines should make a clear link with the business strategy (2d). Understanding the lifecycle of the products (2c) is essential during this process. Additionally, as the main target of these strategic decisions is to gain market share, therefore it is required that products are positioned in such a way that they take an advantage of upcoming trends (3a). It is also useful to use customer feedback while determining the product life cycle to enhance and maintain customer satisfaction (2e). An organization must make sure that there is no pipeline gridlock between for example existing products and the introduction of new products (1d). The portfolio should also be balanced (1e), this in order to increase market share. A company should not focus on only low value projects which are relatively easy with a low risk, and without using new technologies. Furthermore, it is important that the strategic directions are feasible and based on the available resources (1c). If not enough resources are available a company can consider collaborating with partners. By collaborating, the overall quality can improve and the time-to-market can decrease (4a). A make-or-buy decision should be used to determine whether or not to collaborate with a partner company (4b). Finally, communicate the strategic decisions both internally and externally with stakeholders (2b).

We did not use 3e (Establishing organizational readiness), because establishing organizational readiness is not relevant when the portfolio management processes are implemented. Before considering product management an organization should be already prepared from an organizational viewpoint. Therefore, it is more applicable for the guidelines related to the software process improvements guidelines. As a result, we identified the following guidelines for implementing the Portfolio Management function:

PM1. Define a specific business goal of the product(s) within the portfolio.
The first thing a company should determine is the specific goal / target of product(s) within the product portfolio. While defining the business goal the product manager can consider the following approaches

- Strategic enterprise-wide initiative which focuses on new market-leading products and full control across the product lifecycle;
- Cross-functional projects for achieving tactical benefits (by implementing new lifecycle processes across several functions;
- Targeting some very precisely defined improvements to achieve benefits in specific operational areas.

PM2. Actively start looking and documenting market trends and technology trends.
Before starting with determining the specific strategy, a company must start looking at the market trends, general industry movements, technological developments and competitor actions. Specific Research and Development projects can be started in order to identify these existing or new upcoming trends. Documentation of the identified trends is important because they are used while specifying the product lifecycle and the product line. In addition to that, the trends should also be used for the development of the roadmap.

PM3. Determine strategic directions of the product(s) in terms of product lifecycles and product lines and also determine storage policies.
While defining the strategic choices, the company should make sure that current and planned products are positioned to take advantage of upcoming trends. A company has to create a balanced portfolio of high value projects such as high risk versus low risk, across markets and technologies, and make sure that there is no pipeline gridlock in the portfolio. Furthermore, while creating the strategy for the product management function, a company must be reasonable with defining the directions. In order to define the directions properly it is essential to understand the lifecycle of the product and make a clear link with the business strategy. While determining the strategic directions also consider

the collaboration with partners companies by using a make-or-buy decision. When it is decided to buy a product or service, standard Service Level Agreements (SLAs) should be used. Finally, all documents related with identification of the strategic decisions need to be documented. Management of the strategic directions (which involve the product lifecycles and the product lines) also includes the communication with the internal and external stakeholders.

6.6 Overview of the guidelines

Within this section we give an overview of all the identified guidelines from the literature, which are summarized in Table 21. Also an exploratory implementation hierarchy is given per product management function.

Guideline	Implement hierarchy
G1. Introduce the Product Manager function to the company, and assign this function.	1
G2. Each iterative implementation process should follow a controlled approach.	3
G3. Make sure that senior management is committed and employees are involved during the implementation of a SPM function.	2
RM1. Create a procedure for retrieving the wishes, requests, and bug fixes from all the involved stakeholders.	2
RM2. Define policies for requirements management and therefore storage of the requirements.	1
RM3. Design a procedure for transforming the customers' requests into product requirements	3
RP1. Create requirements prioritization procedure or adopt a specific prioritization technique.	1
RP2. Define a requirements selection procedure or adopt a specific selection method	2
RP3. Generate procedures for developing, validating, and communicating release plans	3
RP4. Define change management policies and define a launch procedure.	4
PR1. Define policies for identifying and storing the themes and core asset information.	1
PR2. Identify features for the roadmap and create the roadmap.	3
PR3. Define policies for storing and procedures for communication the roadmap.	2
PM1. Define a specific business goal of the product(s) within the portfolio.	1
PM2. Actively start looking and documenting market trends and technology trends.	2
PM3. Determine strategic directions of the product(s)	3

Table 10: Overview for the implementation of the guidelines.

6.7 Validation

Also the implementation guidelines are validated by carrying out interviews with experts. These interviews were combined with the expert validation of the productization process. Within this section we elaborate more on the objectives and results of the evaluation of the implementation guidelines.

6.7.1 Objective

Main objective of the validation process is to determine the validity of the guidelines within a business environment. Consequently, by interviewing the product managers we tried to validate the recognition that organizations need to capture specific policies and procedures for product management. We also validated the guidelines by interviewing the founder of the reference framework for SPM. With that interview we validated the composition and completeness of the identified guidelines. Especially the notion of creating guidelines for the actual implementation of the SPM processes is validated. This notion did not consist of the approach to improve specific processes. A shortcoming of the validation interviews which were carried out was that it did not consist of the explicit validation of the usability and applicability of the identified guidelines.

6.7.2 Expert panel

For the validation of the guidelines we interviewed three experts (Table 11), which all have knowledge of the reference framework for software product management. Within this section we briefly elaborate on the results of the validation.

Participant	Organization	Remarks
Rudy Katchow	Credit Tools	Currently Product Manager, former student at Utrecht University and also studied the SPM reference framework.
Martin van Mierloo	GX	Product Manager and participant of a SPM course
Inge van de Weerd	Utrecht University	Doctor at Utrecht University and founder of the SPM reference framework

Table 11: Interviewed experts for the validation of the guidelines.

Main change as a result of the validation process is the specification of the terminology used in the guidelines. Several of the guidelines use either the term ‘policy’ or ‘procedure’. For each of these approaches we identified the definitions and the main characteristics. Understanding the differences between the terminologies is also of importance, because each term also involves specific acts and ways to operate. Consequently, we reconsidered the used terminology within each of the guidelines.

As a result of the validation process, we concluded that the respondents agreed on the essence of capturing actions which describe certain activities or proceedings. For example GX introduced this while they were changing in becoming a product software company. The entire process from gathering requirements until the actual delivery is documented in a special workflow plan. All results of that, rules and protocols are designed and managed in JIRA. By using this configuration, there is a possibility to communicate the conditions of requirements easily with the involved stakeholders.

Further validation and the actual adoption of the guidelines need to be executed in order to prove the added value of using the guidelines. The usability of the guidelines should be validated by applying them in several case studies at different organization. Preferably these organizations are also different in business size, so that they are validated at small but also large companies.

PART III: Business case

At MP Objects

7 Business Case

A business case is executed at MP Object in order to validate the productization process and specifically the productization approach. Finally, at the end we present our recommendations for MP Objects.

7.1 Objective

The main objective of the case study for MP Objects is to validate the applicability of the productization process, by using the identified application approach (section 5). This approach is especially designed for companies which already identified a need to start developing a standard software product. With the case study, we tried to evaluate the artifacts when they are applied within a business environment. Furthermore, we applied the productization approach and as a result we defined several recommendations for MP Objects in order to continue with their path to create a software product.

7.2 Company description

MP Objects is a young and growing specialist in providing ICT solutions for the logistic sector. They are a partner for stepwise realization of unique IT solutions for Supply Chain Management within organizations and support companies in the:

- Integration across internal and external systems
- Collaboration between people in the order cycle
- Optimization of processes in the supply chain.

MP Objects' main product, the online Supply Chain Suite (SCS) consists of: 1) The Order Management Systems (OMS) that helps to manage the entire order lifecycle. 2) The Transport Management System (TMS) to streamline all processes of the transportation lifecycle across multiple carriers. 3) The Crossdock Management System (CMS) for efficient deconsolidation of incoming shipments and consolidation of outgoing shipments. The suite is used by 1250 users from 60 different companies, which result in 100.000 weekly transactions.

Currently, MP Objects is focuses on a software development approach more towards specific customers. After they participated in a Software Product Management (SPM) course given at Utrecht University, they got acquainted with the reference framework for SPM. As a result of that, MP Objects wants to transform from developing software specific for a customer to developing software as a product for an entire market. Consequently, the main result of the SPM course is that they already introduced the release management function and they started creating and launching releases each quarter.

7.3 Initial Position

For the determination of the initial position we carried out the assessment and we created the Process Deliverable Diagram (PDD) for MP Objects. As a result of the first step for putting the productization process into practice, we concluded that the initial position of MP Objects is at stage five (Figure 19). We elaborate more on this decision within this section.

7.3.1 Software Product Management Assessment

The assessment was carried out at MP Objects in April 2009, present during is session were: Inge van de Weerd, Martin Verwijmeren, Joost Fieggen, Pascal Durr, and Peter Artz. The purpose of the assessment was to determine the maturity of the (present) software product management processes within MP Objects. The results of this assessment can be found within Appendix D: Assessment. Based on the results we concluded that the best applicable stage for the initial position is stage five

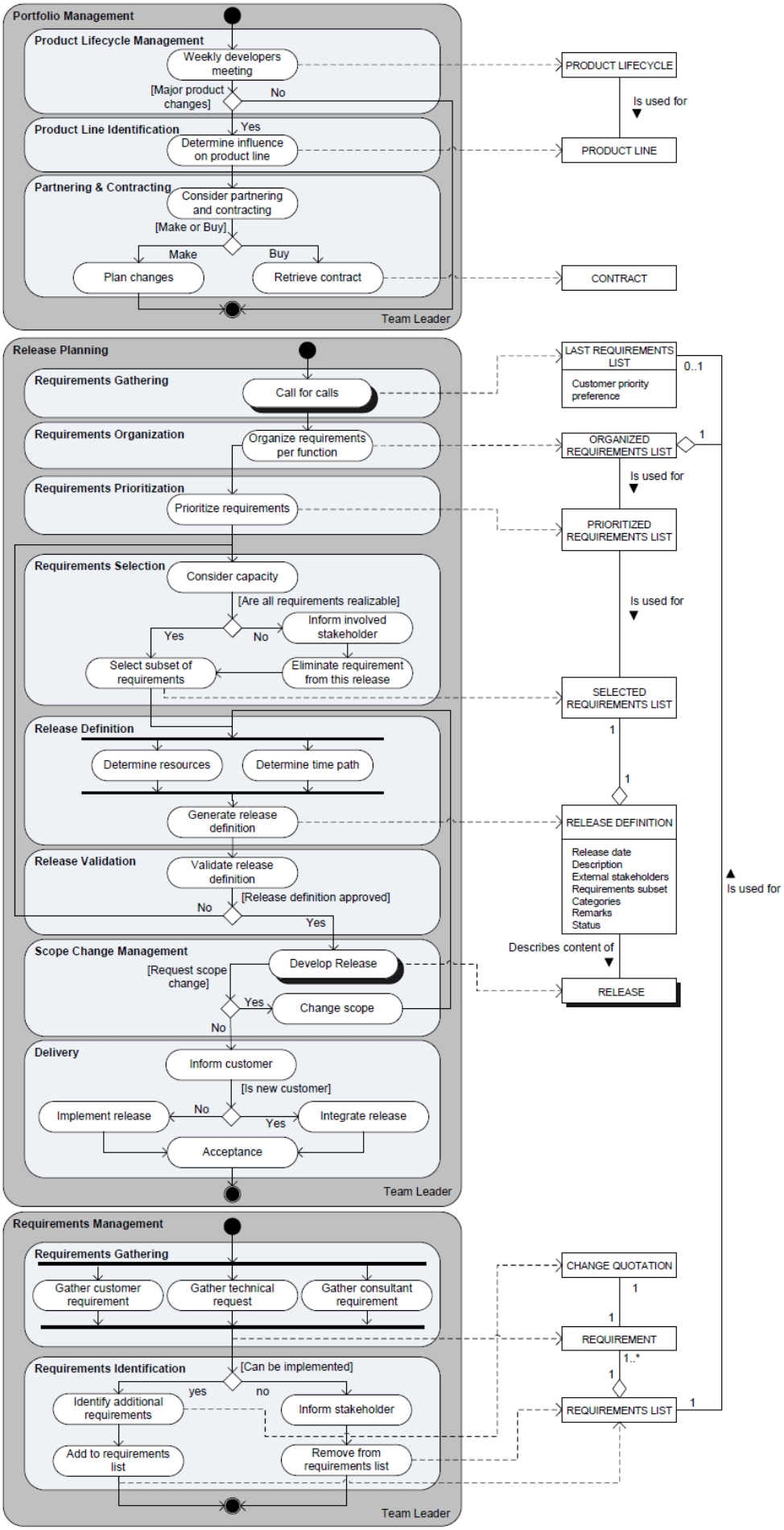
(step 4). We based this decision on the average score in the SPM productization criteria table (Table 25). The reason why stage six (6a or 6b) is not yet applicable, is due to the fact that there is still a specific customer focus in the release planning function. Factors which indicate on this customer focus are the 'Requirements selection' and the 'Main stakeholder involvement on development' of the software.

The last part of the assessment was carried out later in this study and was filled in by two employees of MP Objects (a consultant and a developer). As a result, the two different views were combined in order to get a representative result of the current situation. Remarkable to see was that in several areas both participants differ in their view on the product.

7.3.2 Process Deliverable Diagram

After we carried out the assessment, we created a Process Deliverable Diagram (PDD) (Weerd & Brinkkemper, 2007). This section elaborates briefly on the generated PDD of the initial situation at MP Objects.

Figure 17: Process Deliverable Diagram of MP Objects.
Note: the PDD can be depicted on the next page.


Portfolio Management
Product Lifecycle Management
Weekly developers meeting
[Major product changes]
No
Yes
Product Line Identification
Determine influence on product line
Partnering & Contracting
Consider partnering and contracting
[Make or Buy]
Make
Buy
Plan changes
Retrieve contract
Team Leader
PRODUCT LIFECYCLE
Is used for
PRODUCT LINE
CONTRACT
Release Planning
Requirements Gathering
Call for calls
Requirements Organization
Organize requirements per function
Requirements Prioritization
Prioritize requirements
Requirements Selection
Consider capacity
[Are all requirements realizable]
Yes
No
Inform involved stakeholder
Select subset of requirements
Eliminate requirement from this release
Release Definition
Determine resources
Determine time path
Generate release definition
Release Validation
Validate release definition
[Release definition approved]
No
Yes
Scope Change Management
[Request scope change]
Develop Release
Yes
Change scope
No
Delivery
Inform customer
[Is new customer]
Implement release
No
Yes
Integrate release
Acceptance
Team Leader
LAST REQUIREMENTS LIST
Customer priority preference
0..1
ORGANIZED REQUIREMENTS LIST
1
Is used for
PRIORITIZED REQUIREMENTS LIST
Is used for
SELECTED REQUIREMENTS LIST
1
1
RELEASE DEFINITION
Release date
Description
External stakeholders
Requirements subset
Categories
Remarks
Status
Is used for
Describes content of
RELEASE
Requirements Management
Requirements Gathering
Gather customer requirement
Gather technical request
Gather consultant requirement
Requirements Identification
[Can be implemented]
yes
no
Identify additional requirements
Inform stakeholder
Add to requirements list
Remove from requirements list
Team Leader
CHANGE QUOTATION
1
1
REQUIREMENT
1..*
1
REQUIREMENTS LIST
1

The related activity tables can be found within Appendix E: Process Delivery Diagram (I. Activity Descriptions) and these tables consist of the explanations for all the activities and sub-activities depicted on the left side of the process deliverable diagram. Also the concept definitions can be found in Appendix E: Process Delivery Diagram (II. Concept Definitions). This table elaborates more on the right side of the process deliverable diagram and it consists of the definitions for each of the deliverables. Preferably, each definition is adopted or supported by a reference from the literature (Weerd & Brinkkemper, 2007). While we created the PDD, we concluded that there are already quite some SPM processes present. However, several processes are still customer focused (e.g. the prioritization and selection processes). This implies that the external stakeholders have a bigger influence on the development of the product. Also the requirements gathering process characterizes this customer focus. Currently, MP Objects is gathering and identifying requirements in a continuous process. In addition to that, before a release is developed there is a last 'Call for calls' so that all customers are able to specify and prioritize specific requests which they want to have integrated in the upcoming release.

When we compare the PDD with the different stages of the productization process, we noticed that the product management areas which are in place are portfolio management, requirements management and release planning. As a result of the created PDD, we concluded that the most valid stage for MP Objects is stage five. Main reason for this decision is due to the remaining customer focus. Additionally, when we look at the other product management functions, we concluded that product roadmapping function should be in place. The reason for the absence of the roadmapping activities is understandable, because it is partially based on the fact that the portfolio management function is not actively managing the long-term lifecycle. There are weekly developer meetings where in particular the current development progress is discussed. The absence of the explicit identification of market trends and technology trends also points out that there are no concrete future decisions and directions determined and registered. In addition, the release planning function is also affected hereby. The content of the releases is specified when a previous release is launched (Figure 18). Due to that, the organization of the requirements is not part of the RM function, but it is performed just before a new release starts.

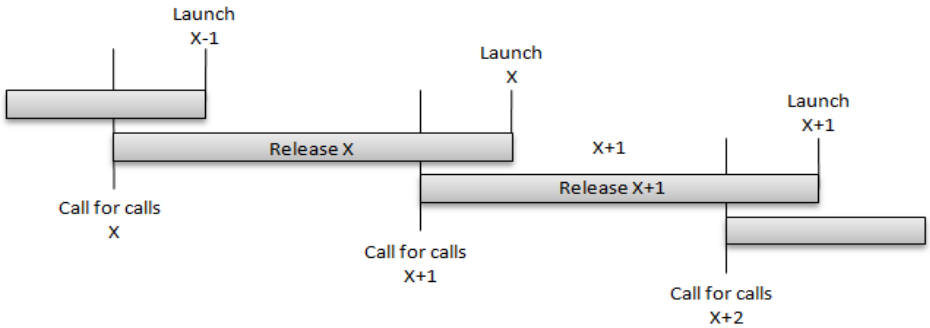


Figure 18: Release planning cycle for MP Objects

Based on the deliverables identified at the right side of the PDD, we also discovered a lack of concrete documentation of a couple of the SPM processes. For example, decisions made for the product lifecycle and product line are not actively documented, managed and monitored. Additionally, the requirements management function is not involving all stakeholders evenly. Mainly the requirements from the customers are registered and additional requirements from developers and consultants are communicated verbally. Ideally, all requirements are managed by using a requirements management tool or database, which stores all gathered requirements in one central place. By applying such environment, also the distinction between market requirements and product requirements can be made and applied more easily.

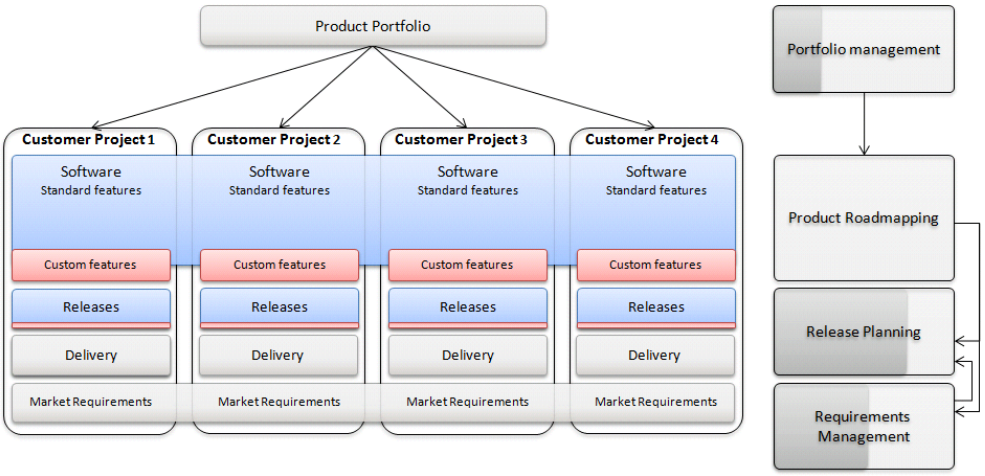


Figure 19: Initial stage MP Objects mapped with the maturity levels

7.4 Gap analysis

After determining the initial position, the next step is to determine the distance with a fully adopted product management situation. In order to determine this distance, we performed a gap analysis which resulted in tangible improvement areas.

7.4.1 Situational Factors

Before we carried out the gap analysis, we first determined the best suitable maturity levels for each of the product management functions for MP Objects. These maturity levels are based on several situational factors adopted from Bekkers et al. (2008b) (Table 23). A detailed elaboration for each product management function can be found in this section.

Portfolio Management

MP Objects develops two different products, the first one (which has their main focus) is the Supply Chain Suite (SCS) and the second product is Export Document Online (EDO). The expected lifecycle (PM_F1) of MP Objects' software is more than fifteen years. When we compare this with the other cases we distinguished that the lifetime is longer compared with the values adopted from the case studies of Bekkers. Basically, this means that even with a small amount of products, PM is rather important for MP Objects due to the long product life. In that case it is advisable to manage the lifecycle and product line in a sufficient way. Secondly when we looked at the market size (PM_F2), this factor supports the need for improving the PM function due to large amount of potential customers within MP Objects' market. Moreover, when we look at the reference value of the market size we noticed that a potential market size of 1500-3500 customers is reasonable. Also this factor illustrates on a certain need to manage the product portfolio, even when a portfolio consists of a small amount of products.

Additionally, also Vähäniitty & Rautiainen (2005) identified the need of managing the product portfolio within a small organization with a small amount of products. They suggest that by using the PM function the awareness of what projects and other development activities are ongoing increases. It also gives an idea of how the resources are allocated and it helps in making informed decisions and trade-offs when this is required.

Product Roadmapping

The situational factors both identified also a need to implement the PR processes within MP Objects. When MP Objects wants to develop products for an entire market instead of a few customers, a roadmap is useful to manage expectations, plans, themes and core assets of the product(s). The first factor which we used to determine the essence of adopting the PR function is the product lifetime

factor (PR_F1). A longer lifetime of a product involves a bigger need to create and update a roadmap actively. The expected lifetime of MP Objects' main product is more than fifteen years and this indicates a significant need to implement the PR function and to start forecasting the future of the product(s). Moreover, also the customer satisfaction (PR_F2) should also be considered. The customer satisfaction can be maintained or improved, by keeping the customers informed about the content of future releases. The roadmap is used as a tool to communicate to the market. For MP Objects the customer satisfaction is eight and this value is at positioned at the end of the sample range from Bekkers et al. (2008c). So both factors indicate on a significant need to implement the PR function.

Also for PR we found literature which recognizes the need to implement the PR function within a small organization. Vähäniitty et al. (2002) proposed that the PR activities can help bringing together the perspectives of business management and software development. In addition, using roadmaps from an early stage is valuable when a relative small company grows and it can help to maintain the focus on development. Due to the fact that MP Objects is currently growing and starting to build a product for an entire market, the need to implement the PR function is increasing.

Release Planning

Currently, the release planning function is the most matured product management area. The two situational factors which are used for the determination of the need for improving the RP function are legislation and release frequency. The legislation shows "the level of influence imposed upon the software product by government bodies" (Bekkers et al., 2008a). For a 'Loose' legislation (RP_F1) is 'release validation' one of the most important processes, based on the results of the assessment we noticed that it is useful when MP Objects improves their RP function in this particular area. The other factor is the release frequency, when the release frequency (RP_F2) has a large amount of days (less product releases a year), the difficulty of managing the releases increases. MP Objects launches a new release each quarter (four releases per year). Based on these factors we can conclude that there is still room for improvements within the release planning function. However, there is no need to implement the RP capabilities within MP Objects to their full extent. Consequently, several release planning processes can be improved, so that management of the releases is performed better and more efficient.

Requirements Management

When we look at the situational factors we distinguish the need for also improving the RM processes. The new requirements rate of MP Objects is around 250 per year (RM_F1), which results into 20 requirements each month. This rate makes it unnecessary to implement all RM capabilities in the organization to their full extent. However, managing the requirements by using a requirements management tool or database is preferred. When we compared a new requirements rate of 250 per year with the reference values we noticed that this rate is not particularly intense. The second factor, customer involvement (RM_F2) is related to the development and determination of the content of the product. A high customer involvement requires a proper RM function because the customers are the main drivers for the identification of new product requirements and this requires a proper requirements gathering process. By installing a requirements management tool also the traceability and maintainability of the requirements improves.

Conclusion

As a result of the situational factor outcomes and the acknowledgement within the literature, we recommend that MP Objects should definitely implement and/or improve the product management functions for PM and PR. For the Portfolio management function we advice a maturity level around 6 – 7. It is not required to have the PM implemented to their full extent, because MP Objects is a small (growing) company with currently two main products. Main improvement for the PM function is related to the identification of market and technology trends.

Secondly, the suitable maturity level for the Product Roadmapping function should be near the 7 – 8. Due to the long lifecycle of the product and the recognition to implement PR in a small company in the literature, we recommend to definitely implement the PR function. Also for the PR function it is not required to have it implemented in its full extent.

Thirdly, the Release Planning function should have a maturity level around 7 – 8. Especially, the requirements prioritization and selection should be changed so that it starts focusing on developing a product for a specific market. Additionally, the development and validation of the release definition can also be improved. This also includes the communication of the release definition to the involved stakeholders.

Finally, the Requirement Management function should have a maturity level around 7 – 8. This is based on the fact that currently no centralized environment is used to manage the requirements. Ideally, a database or requirements management tool should be implemented to support the RM function.

7.4.2 Maturity matrix

We used the SPM Maturity Matrix for the identification of the real need to improve specific functions. By taking the lowest maturity levels of the maturity matrix (Table 24) we were capable to identify the initial General Maturity Level (GML) which is 1. This level is based on PR3 and PM1, which are the lowest scores within the matrix. The (Product Management) Area Maturity Levels (AML) are: for PM level 1; for PR level 1; for RP level 5; and for PM level 4 (depicted from Table 12).

	0	1	2	3	4	5	6	7	8	9	10	11	12
RM1. Requirements gathering		A		B				C		D			
RM2. Requirements identification			A			B			C				
RM3. Requirements organizing					A		B		C				
RP1. Requirements prioritization		A				B		C				D	
RP2. Requirements selection			A			B			C		D		
RP3. Release definition				A			B			C			
RP4. Release validation				A			B		C				D
RP5. Launch preparation					A			B			C		
RP6. Scope change management					A				B		C		
PR1. Theme identification				A	B					C			
PR2. Core asset identification				A					B				C
PR3. Roadmap construction			A			B				C			
PM1. Market trend identification			A		B				C				
PM2. Partnering & contracting			A		B					C			
PM3. Product lifecycle management			A			B		C					
PM4. Product line identification			A			B		C					

Table 12: Initial maturity levels for MP Objects.

Portfolio Management

When we look at the maturity matrix we noticed that there are already three PM capabilities in place. The reason why the PM function is not that mature is quite typical for a small and young software company like MP Objects. The essence of managing the product portfolio in an organization which focuses on a small amount of products is often seen as too complicated. Due to the main focus on one product, there is no need to fully apply the PM processes. Active identification and registration of market trends is the only portfolio management activity which is not in place. On the other hand, in order to become market-driven it is essential that all future trends are identified and documented. In contrast, from the average product software company (red line in the maturity

matrix) we concluded that the essence of improving the management of the product lifecycle and the product line is not that sufficient compared with the 'Market trend identification' and 'Partnering & contracting' capabilities. However, in the ideal situation also these processes need to be improved.

Product Roadmapping
Based on the maturity matrix we confirmed that there are no product roadmapping activities are present. Consequently, the implementation of the roadmapping function is an essential improvement activity for MP Objects. Also the reference line indicates on the relevance to implement the product roadmapping function.

Release Planning
From the maturity matrix we concluded that the RP function is the area with the highest AML. Therefore, the essence of improving the capabilities is much lower compared with the other product management areas. When we compare MP Objects' values with the average product software company we distinguish that the requirements prioritization, requirements selection, and release validation still have a lower maturity level. Consequently, we concluded that there is still some room for improvement of the RP function. Especially the requirements prioritization and requirements selection processes should be supported by the roadmap. By using the roadmap as a guide, the dependencies between the different requirements and releases can be taken into account.

Requirements Management
Within the maturity matrix all the RM processes have a lower maturity level compared with the average product software company. The need for improving the RM function is therefore relatively high. The main reason for this is that when MP Objects want to become more market oriented, the requirements should be managed properly so that all requirements from the involved stakeholders can be managed and organized per release. Also for the RM function, it is not required to implement this function to its full extent.

7.4.3 Process Deliverable Diagram

Also a PDD is used to determine the gap between the current state and the best suitable situation. We identified several organizational changes which are used for creating the recommendations.

Portfolio Management
Based on the PDD of the PM function, we identified the short come of managing, monitoring and documenting the product lifecycle and product line effectively. One of the main factors relevant for the size of the gap is that MP Objects is currently not identifying and registering market or technology trends. These trends should be used in order to determine the future directions of the product lifecycle and product line.

Product Roadmapping
At this moment MP Objects is not managing or maintaining any of the roadmapping processes or other kind of activities related with forecasting the future of the products. Currently, they use a sequential release planning approach which means that after a release is launched and delivered, they start focusing on the new release.

Release Planning
Main difference between the release planning processes which are in place and the processes of the SPM reference framework is the customer focus. Before a release starts, all customers are asked to give their last requests ('Call for calls') which they want to have implemented within the upcoming release. When the customers are providing the list with requests, they also assign a priority to each request. This implies that the involvement of the internal stakeholders while selecting the

requirements is considerably lower than the external stakeholder (customers). As a result of this, more requirements selected which satisfy the customer instead of creating a product designed for a bigger market.

Requirements Management
The first thing we noticed from the PDD is that MP Objects is currently only focusing on the processes of gathering and identifying requirements as an ongoing activity. By eliciting the requirements via the last 'call for calls' process, a rather customer specific development approach is still present. They identify the complexity of the customers' wishes, and if necessary they also identify the related requirements. However, in terms of the reference framework for SPM, they should also make a distinction between market requirements and product requirements. By identifying product requirements, it is possible to describe the requirements in the company's perspective and context. MP Objects is not specifically focusing on an entire market and the requests of potential customers are not actively documented and stored. Finally, the organization of the requirements is performed when a release starts and not when the requirements are managed. The cause of this organizational difference is based on the fact that the real customers' needs are gathered in the beginning of the release planning function.

7.4.4 Conclusion

The actual gap between the initial position and the best suitable adoption of the software product management reference framework (Table 13) is especially focusing on the PM and RP functions. We concluded that the best suitable general maturity level (GML) for the entire product management function is 6. This advice is based on the recommended (product management) area maturity levels for each product management functions separately. Consequently, the following maturity matrix (Table 13) shows the suitable end result for MP Objects. The dark-gray marked areas are the areas which need to be improved. For MP Objects is it not desirable and required to have all product management functions applied in their full extent. First, MP Objects should focus on having an aligned maturity, so that after that they have the possibility to excel in specific areas.

	0	1	2	3	4	5	6	7	8	9	10	11	12
RM1. Requirements gathering		A		B				C		D			
RM2. Requirements identification			A			B			C				
RM3. Requirements organizing					A		B		C				
RP1. Requirements prioritization		A				B		C				D	
RP2. Requirements selection			A			B			C		D		
RP3. Release definition				A			B			C			
RP4. Release validation				A			B		C				D
RP5. Launch preparation					A			B			C		
RP6. Scope change management					A				B		C		
PR1. Theme identification				A	B					C			
PR2. Core asset identification				A					B				C
PR3. Roadmap construction			A			B				C			
PM1. Market trend identification			A		B				C				
PM2. Partnering & contracting			A		B					C			
PM3. Product lifecycle management			A			B		C					
PM4. Product line identification			A			B		C					

Table 13: Best suitable maturity levels for MP Objects.

When we look at the type of the software that MP Objects is developing, we are able to conclude that the desired end stage is 6a. There is no relevance to be in stage 6b for MP Object because there is always a need to integrate the software into an existing infrastructure or communicate with other software. This means that first the maturity of all product management function should be improved before we are able to say that MP Objects product is positioned within stage 6a. However, there is

the possibility to introduce a new product which is designed in such a way that stage 6b is reached. Currently, there are some emerging interests in developing a 'light' version of the Supply Chain Suite. The potential benefits of this product are that it will be completely configurable and it will consist of significantly less functionalities without any required customization.

7.5 Recommendations

Within this section we provide recommendations for MP Objects in order to improve their product management functions. The biggest challenge for MP Objects is the implementation of the product roadmapping function. We also take the adoption of Scrum into account while defining the recommendations because MP Objects is considering applying an agile development approach in the near future. Consequently, this section elaborates more on the defined recommendations for MP Objects.

7.5.1 General suggestions

In this section we present some general recommendations for MP Objects.

Product Manager

Firstly, we recommend that MP Objects should adopt the role of product manager. One person should have the full responsibility for managing the product. The application of this function, can be supported by guideline G1 (section 6.1). This guideline explains in more detail the main responsibilities of the product manager.

Communication

Based on the results of the case study we identified the need of integration of and communication with the consultants. The consultants are sometimes not fully aware of the development of the product. By adopting the Scrum development method, the communication with the internal stakeholders can be improved. A property of Scrum is that several meetings should be carried out in order to evaluate on sprints and releases. By carrying out the release meetings in collaboration with the other internal stakeholders, they are better aware of the changes and new functionalities developed within a release.

Generally speaking, a main concern for MP Objects is the communication with involved stakeholders. A good example of the communication issue is the launch preparation process. Currently, while launching a release a standard template is used. This template describes the exact procedure to launch the releases and main subjects of this template are:

- Customer specific (contact) information;
- What to prepare before launching the release;
- What to carry out on the day of implementation;
- What to do after the implementation (after care).

However, main concern for the launch preparation process is the communication of the content of the releases. Currently, customers and other internal stakeholder are not fully aware of the new functionalities and changes which are included within a release. A proper communication procedure should make sure that all stakeholders are informed about the development status of a new release. This should also include the possibility that specific stakeholders are informed about the status of their request / requirements.

Also the communication of the designed roadmap is recommended. By publishing a part of the roadmap in for example the quarterly, new potential customers might get interested in the product. This involves the short-term planning (upcoming releases) and long-term planning (new core functionalities or techniques).

7.5.2 Portfolio Management

The process which is important for MP Objects and should have the main focus is the market trend identification. The relevance of implementing the other processes seems considerably lower at this moment.

Market trend identification

Currently, MP Objects is already paying some attention on new technical improvements. In the futures active market analysis must reveal new directions for the future of the product, for example by looking at new standards or other opportunities through technological combinations. Important after the identification of certain trends is the documentation of these trends. Future releases should cover these trends so that a competitive position is achieved. Retrieving these trends is an ongoing process and they should be identified by the consultants as well as the developers. Ideally, all trends are stored at one central place so that everybody is familiar with the already identified trends and that they are easy to manage. The introduction of the trends identification can be supported by using guideline PM2 (section 6.5) for the realization of this process.

7.5.3 Product Roadmapping

For the adoption of the product roadmapping function we recommend to use the guidelines described in section 6.4. Furthermore, the implementation of the roadmapping function can be supported by several different approaches which are available in the literature. In order to communicate the roadmap to MP Objects' customers and market, it is interesting to publish (part of) the roadmap within the quarterly brochure. Currently, each quarter a brochure is send to all contacts of an already composed mailing list. By publishing the roadmap, potential customers are getting acquainted with the future development of the product.

T-Plan Fast-start

For the roadmapping activity of MP Objects we suggest the adoption of the "T-Plan fast-start" approach (Phaal et al. 2004). This approach is developed as part of a three-year applied research program and it is already widely used. It consists of a clear process which describes the development of the roadmap by facilitating four different workshops (Figure 20). Additionally, according to Holmes & Ferrill (2005) this method is also applicable for the creation of company-specific product/technology roadmaps for Small and Medium Enterprises (SMEs). Benefits of this approach are (Phaal et al. 2004):

- Supporting the start-up of company-specific roadmapping processes;
- Establishing key linkages between technology resources and business drivers;
- Identifying important gaps in market, product and technology intelligence;
- Developing a "first-cut" technology roadmap;
- Supporting technology strategy and planning initiatives in the firm;
- Supporting communication between technical and commercial functions.

T-plan fast-start is suitable for MP Objects because it provides a quick, systematic approach and it makes maximum use of the time committed by senior management and the participants. The first-cut technology roadmap produces clearly connects technology development & acquisition with business drivers & strategy (Holmes & Ferrill, 2005). In addition, this approach also integrates the market and business drivers to prioritize the items in the roadmap (Phaal et al. 2004). These business drivers represent the combination of the underlying customer and business motivations. A technology roadmap is better suitable for MP Objects because then stakeholders are able to see when specific functions or new techniques will be integrated in the product. Additionally, roadmapping delivers also positive benefit to the SMEs, which range from the staff members being able to step aside from daily pressures to plan with the assistance of a neutral facilitator, through to

the integrated identification of current operational problems and solution development (Holmes & Ferrill, 2005).

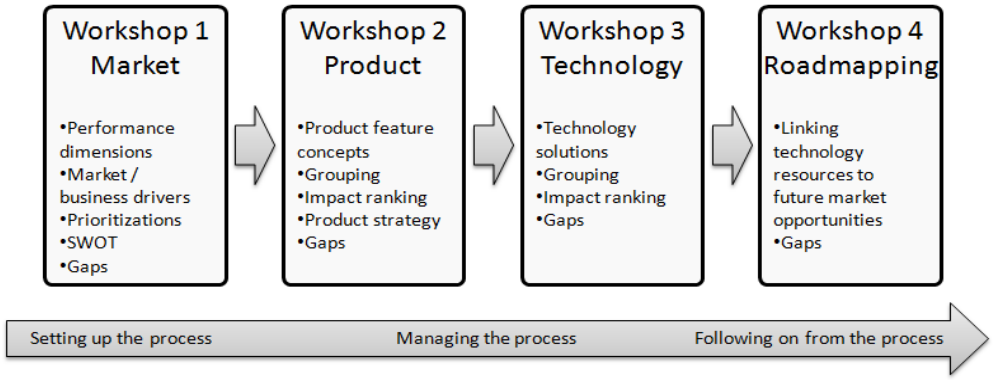


Figure 20: T-Plan fast start process (adopted from Phaal et al., 2004).

The T-Plan approach is achieved by facilitating the four workshops (Figure 20). As a result of the four workshops, two matrixes are designed which cover the three layers of the roadmap (as illustrated in Figure 21). The first workshop consists of the identification of the market drivers and business drivers for MP Objects and each driver should be prioritized. Examples of the main market drivers especially applicable for MP Objects are: 'Time to market', 'Green supply chain', and 'Rising transportation costs'. Additionally, example business drivers are: 'Lower logistic costs', and 'Higher logistics quality'. The second workshop identifies a set of product feature concepts, which should grouped into ranked themes. The defined themes should be combined with the market and business drivers from Workshop 1. By combining these two sets a Market-Product grid emerges which consists of the weighted scoring of the identified desirable product features. Examples of some themes for MP Objects are: 'Security', 'Transport Planning', 'Event Management', and 'Transport Execution'. The third workshop identifies the technological solutions that could deliver the desired product features. Also the technological solutions should be grouped into ranked themes. The combination of the results of the second Workshop and the technology themes produce the second matrix. Finally, the last workshop combines the two designed grids into a complete "first-cut" which covers the three layers of the roadmap. While defining the roadmap a combination between market 'pull' and technology 'push' must be made so that a balanced roadmap is created. Do not focus on either one of these values. More detailed information on designing the T-Plan can be adopted from Phaal et al. (2003).

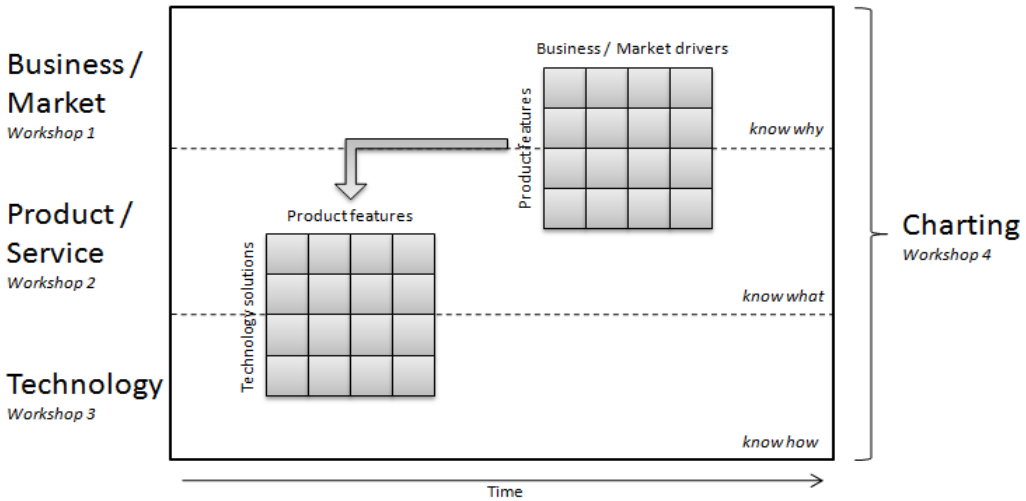


Figure 21: Technology roadmap based on the workshops (adopted from Phaal et al., 2004).

After the adoption of the product roadmapping function, MP Objects should make sure that: 1) the roadmap is updated frequently (also described in Guideline PR3) and 2) that roadmap is rolled-out within the organization. Consequently, there are two possible methods in order to roll-out the roadmap and communicating the results to the other parts of the organization. Firstly, the *Top-down* approach, where the requirement for roadmaps is prescribed by senior management – the particular format may or may not be specified. Secondly the *Bottom-up* approach, where the benefits of using the method are communicated and support provided for application of the method where a potential fit with a business issue / problem is identified (Phaal et al., 2004).

In relation to the guidelines (section 6.4) which we identified during this research, we suggest to use guideline RP1 for workshop 2 and workshop 3 and guideline RP2 for workshop 4. Finally, for rolling out the roadmap we recommend to use guideline RP3 to communicate the results internally and externally.

7.5.4 Release planning

The release planning function is the most matured product management function. However, when MP Objects is planning to adopt an agile development method like Scrum, their current release planning function should change in several areas. But first we provide some recommendations based on the results of the gap analysis for the lower matured processes.

Release definition: Development and validation

Main concern for these processes is the involvement of the internal and external stakeholders. Currently, the consultants are not fully aware of the content and functionalities of the releases. Therefore the creation of a release definition template is of essence for communicating it to the internal stakeholders. Also for the validation of the release definition is it important to communicate the actual developed content of a release with the involved internal stakeholders. Before the developments starts, it first needs to be formally approved.

Requirements Prioritization

The requirements prioritization process is currently (mostly) driven by the customers and not by the internal stakeholders. As a result of the 'Call for Calls', each customer is able to provide their (prioritized) most important requests. The first three of these requests are always included in the upcoming release. This means that the product is currently developed by the 'paying' customers and not particularly for the potential customers within the market. In order to create a standard product, the prioritization of requirements for a release should be changed. In the literature there are several methods available for the requirements prioritization process. The best suitable method for prioritizing the requirements is based on making a trade-off between the different (internal and external) stakeholders. Changing the requirements prioritization process cannot be done in one instance and therefore MP Objects should first apply another approach in which a trade-off is made between all involved stakeholders. This approach still keeps the customer quite involved in the prioritization process, but then they are not the main driver.

In order to create a software product for a market, we recommend that other factors are also considered while prioritizing the requirements. This means that the approach MP Object currently uses should change in such a way that other factors are also used. Lehtola et al. (2004) provide a clear overview of the three points which affect the requirements prioritization process.

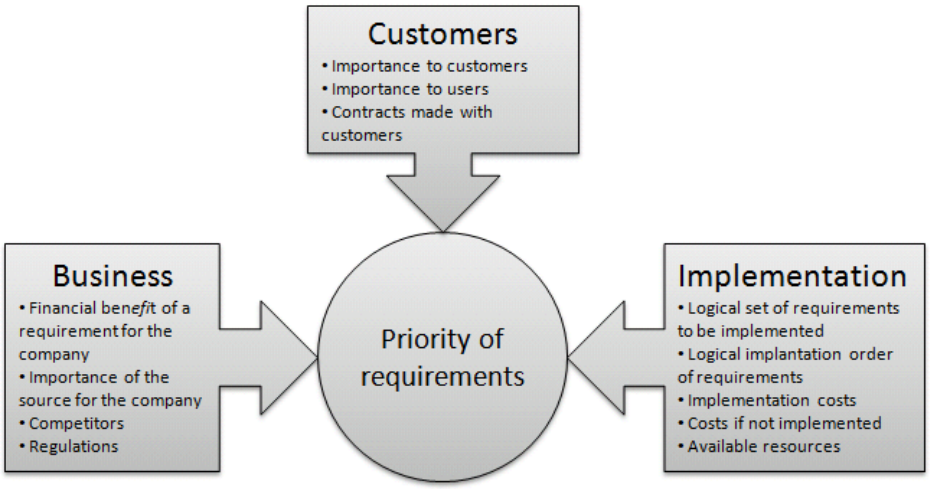


Figure 22: Three points view, adopted from Lehtola et al. (2004).

Scrum

Recently, MP Objects was introduced to the (agile) Scrum development method and they are considering implementing this method in their engineering department. However, in order to combine Scrum with the release planning function a specific configuration is required. Agile Scrum development process is designed to improve productivity and reduce time to market for new product. Scrum has evolved into three different types: type A) Isolated cycles of work, type B) Overlapping iterations, and type C) All at once (Sutherland, 2005).

	Type A	**Type B**	**Type C**
Iteration / Sprint overlap	Down-time between sprints/no overlap (insufficient Product Owner engagement	Slight overlap (Planning/Prep)	Complete and multiple
Level of involvement	Development team	Product management team	Entire organization
Release	Evert 4-6 iterations	Every 2-4 iterations	Every iteration/sprint
Iteration lengths	Fixed	Fixed	Multiple overlapping / varying lengths
Cycle time for new requests delivery	4-6 Months	2-3 Months	Monthly
Release per year	Total – 2, two major per year and patch as needed	Total – 4, Quarterly major, patch as need	Total – 12, weekly patch, monthly update, quarterly major

Table 14: Characteristics of the different Scrum types (Sutherland & Schwaber, 2007).

For MP Objects we recommend to implement Type B Scrum because this fits better in terms of releases, sprints and level of involvement (as described in Table 14). Currently, MP Objects is developing four releases per year and there is already an overlap between each release (as illustrated in Figure 18). By adopting type B, the sprints can be executed continuously with the sprint backlog always full at the beginning of each new iteration. In terms of the reference framework for SPM, the backlog can be compared with the requirements management function (section 7.5.5). Within this type, the product backlog must be fully loaded at all times so that a developer never questions what to do next (Sutherland, 2005). The overlap between the sprints is designed by adding the product definition tasks for the next sprint into the current sprint. This allows working smoothly from sprint to sprint (Sutherland & Schwaber, 2007). Essential for the implementation of Type B Scrum is that a company should have a sustainable development policy and process. This implies that the following key indicators should be visible (Sutherland & Schwaber, 2007):

- Team autonomy: the Scrum team is totally responsible for their product and no outside agency impacts the work plan of the team inside a sprint.
- The Product Owner is part of the Scrum and affects product design and implementation within a Sprint without disrupting self-organization.
- Self-transcendence: individuals move beyond self-gratification to focus on team performance.
- Cross-fertilization: expertise is regularly shared across team members and no single person is a bottleneck.

An advantage of implementing the Type B Scrum approach is that after each iteration a complete (workable) version of the product is created (as illustrated in Figure 23). This approach is very practical for MP Objects, mainly because it increases the involvement of the stakeholders during the development and not only at the end when a release is ready to be launched. Stakeholders can be better informed about the status of new features.

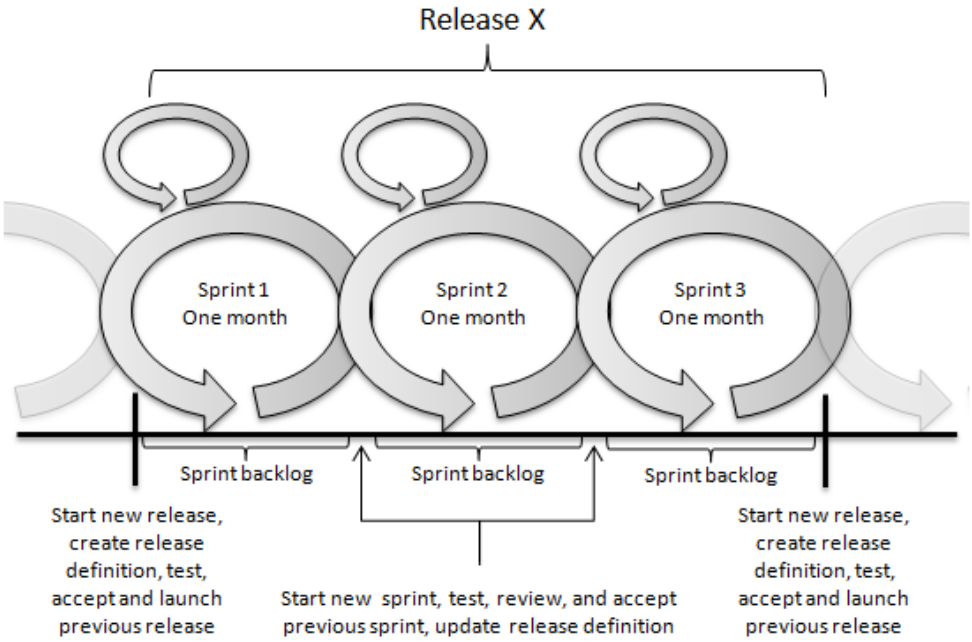


Figure 23: Type-B Scrum, a continuous flow of sprints (adopted from Sutherland, 2005).

Consequently, the requirements selection process should be based on the Scrum approach. This is performed by selecting the optimal list of features (requirements) from the product backlog (requirements management) into a sprint backlog. For each sprint the estimated development effort (velocity) must be determined so that at the end of the sprint (during the sprint review meeting) it can be compared with the actual development time. The results of this comparison should be taken into account when the new sprint is defined. Additionally, the creation of the release definition will also be slightly more complex. Each release is build upon two (workable) demos which consists of a selected list of features (requirements). Eventually, the release definition is the combination of three sprint backlogs. Therefore we recommend that when a new release starts, a release definition is created for all three sprints (as illustrated in Figure 23). At the moment a sprint is finished, the release definition must be updated based on the results of the finished sprint and the upcoming sprint. Furthermore, within the release definition a division should be made between the different sprints so that clearly is described which requirements will be implemented within which sprint/iteration. The sprint backlog consists of the features broken down into specific tasks.

Furthermore, the Scrum process consists of a number of meetings (Paasivaara et al., 2008) and the most important meetings for MP Objects are the ‘Sprint Planning Meeting’, ‘Sprint Review Meeting’ and the ‘Sprint Retrospective’. The Sprint Planning Meeting starts at the beginning of a sprint cycle

and during this meeting the features (requirements) for the upcoming sprint are selected. The selected features should be described in a Sprint Backlog which consists of the details regarding the required time and resources for that specific sprint. At the end of a sprint cycle, two meetings are held: the Sprint Review Meeting and the Sprint Retrospective. The Sprint Review Meeting evaluates on the work that was completed and not completed. Also the workable version is shown to the stakeholders so that they are aware of the development status and changes in the product. Additionally, during the Sprint Retrospective meeting all team members reflect on the executed sprint so that a continuous process improvement can be achieved.

7.5.5 Requirements Management

Depending on the decision to implement Scrum, we provide some recommendations for both situations when Scrum will be implemented and when it will not be implemented. Based on the fact that MP Objects is already using JIRA for tracking down bugs and errors, we recommend that this tool is also used for managing the requirements.

Market requirements and Product requirements

Alternatively, another approach for storing the requirements is by making the distinction between 'market requirements' and 'product requirements'. When requirements are received from the internal stakeholders as well as the external stakeholders, they must be managed as 'market requirements'. Additionally, for each market requirement should be determined which specific technical requirements are required in order to realize it. These technical requirements are also identified as 'product requirements' and describe the requirement in a business (technical) perspective. The differentiation between requirements in a customer perspective and a product perspective improves the traceability.

Therefore we provide a configuration (which is also applicable for JIRA), so that it is to link the market requirement with one or more product requirements. For storing the market requirements we propose the following fields:

- ID: each requirement must have a unique id.
- Label: this field consists of a clear name for the requirement
- Description: additionally, also a short and concise description must be given
- Priority: for each market requirement the source can give the requirement a priority
- Source: this field shows who provided the requirement e.g. a customer, consultant, developer, partner company, etc.

For the product requirements there must be a possibility to link it with (zero or more) market requirements. Also the possibility to connect product requirements with each other should be included, so that dependencies between requirements are made clear. As a result we propose the following fields:

- ID: each requirement must have a unique id.
- Label: this field consists of a clear name for the requirement
- Description: additionally, also a bright description must be given
- Priority: for each product requirement the product manager determines a priority
- Status: what is the current development status of the requirement
- Version: which release(s) are affected by the product requirement
- Attachment: also the possibility to assign an external files must be possible

An advantage of configuring JIRA for requirements management is that also another part of the reference framework for SPM can be implemented. For example the organization of the requirements can be included. By storing the identified themes from the roadmap it is possible to

organize specific requirements to themes and core assets. Creating these connections is achievable in JIRA, but using them takes more effort and time.

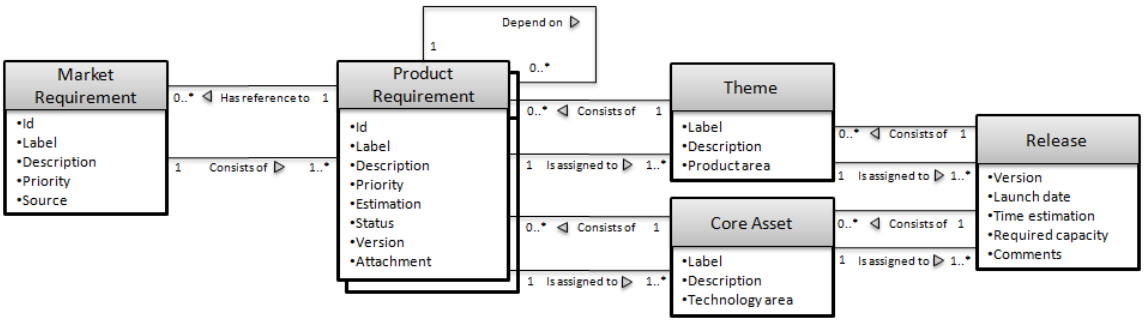


Figure 24: JIRA configuration.

Scrum

When the Scrum development method will be applied, the requirements management configuration should be based on this method. This implies that the processes of the requirements management functions slightly change. All features, wish-list items (market requirements in terms of SPM reference framework) need to be prioritized by business value and stored in the product backlog. Additionally, the sprint backlog consists of the features which will be implemented within an upcoming sprint. These features are broken down into tasks (product requirements in terms of the SPM reference framework); as a best practice and the necessary hours of work should be estimated after that.

For JIRA there are several available plugins which are specially designed to support an agile development method. Two available free plugins are: Agile Wall Report[3] and Agile Velocity Tracking Plugin[4]. The agile wall report plugin is designed to create an overview of all tasks (or user stories or issues) which are grouped by their status into three columns: 'not started', 'in progress' and 'done'. The other free plugin, Agile Velocity Tracking Plugin, generates velocity tracking charts over versions. The velocity is the amount of effort a team can handle in one sprint. This process is helpful for determining the capacity of the development team in order to optimally use the resources within each sprint. Atlassian also sells a plugin designed to manage agile development, which is GreenHopper[5]. This product provides agile project management to simplify both sprint planning and task tracking for a sprint team. Features which are included in the GreenHopper plugin are: 1) Card Management, 2) Sprint Planning, 3) Task Tracking, 4) Charting Progress, and 5) Tool integration. This plugin is a very useful tool to support the requirements management and release planning functions. Also keeping track of the status of the development can be managed easily within GreenHopper.

Alternatively, it is also possible to manually configure JIRA for requirements management and the Scrum development method. However, several functionalities of GreenHopper cannot be reproduced due to the complexity and integration within JIRA.

7.5.6 Final remarks

Recently, Atlassian released the latest version of JIRA, which is version 4. This version has a new type of license which is based on the number of users. For a small amount of users the license costs are considerably lower. MP Object should therefore think about updating the current version into the latest version because also new configuration possibilities are implemented within that version.

[3] http://confluence.atlassian.com/display/JIRAEXT/Agile+Wall+Report
[4] http://confluence.atlassian.com/display/JIRAEXT/Agile+Velocity+Tracking+plugin
[5] http://www.atlassian.com/software/greenhopper/

7.6 Validation results

This section describes the last validation approach which is used in order to validate the applicability of the productization process and their stages. Firstly, the initial position was determined by carrying out the assessment and creating the process deliverable diagram. This is followed by a gap analysis which determines the actual gap between the initial condition and the best suitable situation. The best suitable situation is determined based on the results of the situational factor questions from the assessment. Finally, based on the results of the gap analysis we provided several recommendations specifically form MP Objects.

By executing this case study, we were able to validate the applicability of the productization approach. Additionally, based on the results of the case study we are able to conclude that the productization process is also applicable for small software companies. However, main concern while adopting the reference framework for SPM is in which degree the processes need to be implemented or in other terms: how high the maturity levels of product management function need to be. Based on the situational factors we were able to determine that a fully adopted references framework is not necessary. Future research should look deeper into the integration of the approach which Bekkers is currently developing.

PART IV: Conclusion

8 Conclusions, Discussion & Future Research

In this section we give an overview of the answered research questions. In addition, we also explain in detail the limitation of this research and the future research opportunities.

8.1 Conclusion

In conducting this research, we have identified a process which describes the transformation from developing customized software to a standard software product. We also included the validation of the artifacts, which we validated by using different methods. Based on the results of validation process and the applied adjustments, we are able to conclude that the artifacts presented within this research provide a substantial contribution to the scientific field. Consequently, based on the presented artifacts we were also able to answer our main research question of this research:

How can the Software Product Management reference framework support the transformation from developing customized software to a standard software product?

We have answered this question in PART II: Artifact construction, were we developed and validated the productization process and the productization approach. The identified productization process is a useful guide for organizations which want to become a product software business. All the stages of the productization process are also mapped with the SPM reference framework. As a result of that, it can be depicted how for the entire process the product management functions mature during the transformation. Additionally, per stage a brief description is presented which explains in more detail the characteristics of that particular stage. The stages represent the different phases which can occur during such transformation. Eventually, when an organization has reached their desired end stage, all product management functions should be in place. The actual maturity levels can be determined by adopting the productization approach. In this approach an assessment is used to determine the initial situation and the best suitable end situation of an organization. By using these two results, a gap analysis determines the actual distance in order to become a software product business. The productization approach is specially designed to apply the productization process within an organization.

Additionally, we also defined five sub-questions which are used in order to be able to answer the main research question of this research:

1: What kind of characteristics describe the stages of the transformation?

The answer to this question can be found within section 4. There we provide graphical and textual descriptions of the entire productization process. For each of the stages of the productization process we defined some characteristics which distinguish the main differences between the stages. The textual descriptions consist of the events which presumably can occur within a stage. The descriptions also provide useful information of the related product management functions. As a result of the validation process, several adjustments were made to the characteristics and textual descriptions.

2: What are the specific differences of developing software for a specific customer compared with developing software for a market?

An overview of 20 differences is presented within section 3.4. The identified differences are the outcome of an extensive literature study. While carrying out the literature study we recognized that several different terminologies which are used. Based on these terminologies we identified a large amount of differences (Appendix A: Differences). Additionally, we also looked deeper into the

differences per product management function. As a result of that, a number of additional differences were identified. Finally, we summarized all the retrieved results into an overview of twenty clear differences between customized software development and standardized software development (Table 4).

3: What are major guidelines that are important for the implementation of the SPM reference framework during the transformation?

This sub-question is answered by the presented list with implementation guidelines, which can be found in section 6. The guidelines are the outcome of an extensive literature study on product management in general and per product management function. Also a conceptual implementation hierarchy is provided within the overview of guidelines. We validated the guidelines by interviewing three SPM experts. However, more validation is required to validate the actual added value of each guideline.

4: How can an iterative and incremental implementation and the maturity matrix support the adoption of the SPM reference framework?

The SPM Maturity Matrix is a powerful method to determine the maturity of specific product management processes within an organization. By identifying which capabilities are in place, small and medium sized companies can improve their processes in a more optimal manner. The added value of the maturity matrix is validated during the case study. There the matrix is used in the productization approach in order to determine the actual gap for the productization process. The results of the gap analysis are recommendations which should be applied by an iterative and incremental implementation. This means that when an organization is willing to improve specific product management areas, they should focus on one function at the time.

5: To what extent is the implementation of SPM useful and interesting for a small sized organization?

The answer to this question is answered by the executed case study (section 7). In that section we applied the productization process and we created several recommendations for MP Objects in order to continue their path to create a standard product. Based on the situational factors we were able to determine the real need to have specific processes implemented and to what extent they should be implemented. Small sized companies regularly develop only one product and as a result of that, it seems sometimes too much to manage a portfolio and create a product roadmap. Therefore, it is not required for small sized organizations to have all the product management functions implemented to their full extent. However, supported by the literature, we recommend for MP Objects to definitely reconsider the portfolio management function. Consequently, the implementation of the SPM reference framework is definitely useful for small organization but in another degree.

Finally, the adoption of the reference framework for SPM is also another acknowledgement of the validity of this framework. Additionally, this research also contributes on the validity and applicability of the SPM Maturity Matrix. Both techniques are used within the productization process or productization approach and significantly support this entire process.

8.2 Discussion & future research

The result of this research project is a productization process which describes the transformation from customized software development to a standard software product. This process is supported by the implementation of the reference framework for Software Product Management. However, there is still room for further research in this area. We elaborate more on the available future research triggers within this section.

The scope of this research is limited to customizable software product (software-based service) and standard software product (stage 6a and 6b). Research carried out by Xu & Brinkkemper also identified the overlap of Open Source software (illustrated in Figure 25). Open source software is software for which the underlying programming code is available for inspection and modification by any interested person. An advantage of this type of software is that other developers are able to test, fix bugs, add new features, and make other changes on the programming code easier. Further research should look deeper into the integration of Open Source software in the productization process.

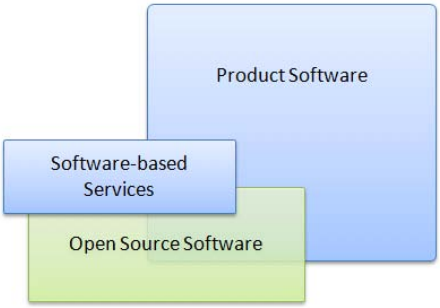


Figure 25: Categories of product software (adopted from Xu & Brinkkemper, 2005).

Secondly, the transformation process we identified in this research is limited to the process of becoming a product software company (market-driven). As a result of the validation of the transformation models and descriptions, the possibility to change back is also suggested. Eventually, there is a possibility that organizations are able to identify a need to change back for example due to a saturated market. However, the scope of this research is limited to the transformation to a software product and we therefore did not consider this process within this research project. Future research must be carried out to determine the exact reasons and consequences of changing back.

A shortcoming of the productization approach to put the transformation process into practice is that reliability of this approach should be considered. The identification of the best suitable maturity levels is currently performed by evaluating on a number of situational factors. Another approach to determine the suitable maturity levels is also defined by Bekkers et al. (2008a). Based on all the situational factors a company should able to determine how mature specific product management areas need to be in their situation. The result of the master thesis of Bekkers is a conceptual linkage of only a few of the situational factor questions and their maturity levels. The reason why we did not use this approach is because Bekkers is currently performing his PhD on this field of research. When his PhD research is finished, the usability and possibility to integrate his approach can be considered. Presumably, this will result in other maturity levels which are better applicable to a company's situations. Alternatively, future case studies should be carried out to further validate the productization approach.

Additional validation and the actual adoption of the guidelines need to be executed in order to prove the added value of using the guidelines. The usability of the guidelines should be validated by applying them in several case studies at different organization. Preferably these organizations are also different in business size, so that they are validated at small but also large companies. Finally,

further validation of the productization process is needed in order to determine the validity and applicability of the identified stage.

Finally, the suggestion of merging stages is also interesting to study in a future research project. This research project especially focuses on the customer satisfaction while changing to become a product software business. This trigger particularly affects stage 5 of our presented productization process. Future research and more expert interviews or case studies are required in order to be able to provide a valid answer.

Abbreviations

The abbreviations we used:

AML	(Product management) Area Maturity Level
BESMART	Transformation framework to change from BESpoke software development to MARkeT software development
CEO	Chief Executive Officer
CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integration
EDO	Export Document Online
FTE	Full Time Equivalent
GML	General Maturity Level
Gx	Guideline x, where x is a number
KLOC	1000 Lines of Code, K stands for thousand
PDD	Process Deliverable Diagram
PhD	Doctor of Philosophy
PL	Product Line
PLM	Product Lifecycle Management
PMngr	Product Manager
PM	Portfolio Management
PM_Fx	Portfolio Management Factor x, where x is a number
PML	Process Maturity Level
PR	Product Roadmapping
PR_Fx	Product Roadmapping Factor x, where x is a number
PSKI	Product Software Knowledge Infrastructure
R&D	Research and Development
RE	Requirements Engineering
RM	Requirements Management
RM_Fx	Requirements Management Factor x, where x is a number
ROI	Return on Investment
RP	Release Planning
RP_Fx	Release Planning Factor x, where x is a number
SCS	Supply Chain Suite
SMEs	Small and Medium Enterprises
SPM	Software Product Management
SWOT	Strengths, Weaknesses, Opportunities, and Threats
UML	Unified Modeling Language

References

Abramovici, M., & Sieg O. C. (2002). Status and Development Trends of Product Lifecycle Management Systems. *Proceeding of International Conference on Integrated Product and Process Development*, pp. 21-22.

Akker, J. v.d., Brinkkemper, S., Diepen, G., & Versendaal, J. (2005). Determination of the next release of a software product: an approach using integer linear programming. *Proceeding of the Eleventh International Workshop on Requirements Engineering: Foundation for Software Quality REFSQ'05, 10*, pp. 247–262.

Akker, J.v.d., Diepen, G., & Hoogeveen, J.A. (2007). A column generation based destructive lower bound for resource constrained project scheduling problems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 4510 LNCS*, pp. 376-390.

Alves, C., & Castro, J. (2006). A Study in Market-Driven Requirements Engineering. *9th Workshop of Requirements Engineering.*

Bekkers, W., Weerd, I. v., Brinkkemper, S., & Mahieu, A. (2008a). The Influence of Situational Factors in Software Product Management: An Empirical Study, *Proceedings of the 2008 Second International Workshop on Software Product Management*, pp. 41-48.

Bekkers, W., Weerd, I. v., Brinkkemper, S., & Mahieu, A. (2008b). "Situational Process Improvement in Software Product Management", Thesis report: INF/SCR-08-09. The Netherlands: University Utrecht.

Bekkers, W., Weerd, I. v., Brinkkemper, S., & Mahieu, A. (2008c). "The Relevance of Situational Factors in Software Product Management", Technical report: UU-CS-2008-016. The Netherlands: University Utrecht.

Berander, P., & Andrews, A. (2005). Requirements prioritization. In A. Aurum & C. Wohlin (ed), *Engineering and Managing Software Requirements*. (pp. 69-94). Germany: Springer-Verlag

Bergström , J., & Dahlqvist A. (2007). "BESMART - a framework for shifting from BESpoke to MARkeT-driven requirements engineering", Thesis report: MSE-2007-24. Blekinge Institute of Technology, Sweden: Lund University.

Bohl, O., Frankfurth, A., Schellhase, J., & Winand, U. (2002). Guidelines - A Critical Success Factor in the Development of Web-Based Trainings, *International Conference on Computers in Education (ICCE'02)*, Auckland, New Zealand, pp.545.

Bosch, J. (2000). *Design and Use of Software Architectures: Adopting and Evolving a Product-Line Approach*. Massachusetts: Addison Wesley.

Brinkkemper, S., Weerd, I. v., Saeki, M., & Versendaal, J. (2008). Process improvement in requirements management: A method engineering approach. *Lecture Notes in Computer Science Volume 5025*, pp. 6-22.

Carlshamre, P. (2002a). *A Usability Perspective on Requirements Engineering: From Methodology to Product Development*, Department of Computer and Information Science**.** Linköping**:** Linkoping University.

Carlshamre, P. (2002b). Release Planning in Market-Driven Software Product Development Provoking and Understanding, *Requirements Engineering, 7*(3), pp. 139-151.

Cleland, D.I., & Ireland, L.R. (2002). *Project management: Strategic Design and Implementation,* fourth edition.

Clements, P. C., Jones, L. G., Northrop, L. M., & McGregor, J. D. (2005). Project management in a software product line organization. *IEEE Software, 22*(5), pp. 54-62.

CMMI Product Team. (2002). Capability Maturity Model Integration (CMMI), Version 1.1, CMU/SEI- 2002-TR-012. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University.

Codenie, W., De Hondt, K., Steyaert, P., & Vercammen, A. (1997). From Custom Applications to Domain-Specific Frameworks, *Communications of the ACM, 40*(10), pp. 71-77.

Cooper, R. G., Edgett, S. J., & Kleinschmidt, E. J. (2000). New problems, new solutions: making portfolio management more effective, *Research Technology Management, 43*(2), pp. 18-33.

Cooper, R., Edgett, S. J., & Kleinschmidt, E. J. (2001). Portfolio management for new product development: Results of an industry practices study. *R&D Management, 31*(4), pp. 361-380**.**

Cusumano, M. A. (2004). *The business of software: What every manager, programmer, and entrepreneur must know to thrive and survive in good times and bad*. New York: Free Press.

Cusumano, M. A. (2008). The changing software business: Moving from products to services, *Computer, 41*(1), pp. 20-27.

Dver, A. S. (2003). *Software Product Management Essentials*. Florida: Anclote Press.

Ebert, C. (2007). The impacts of software product management. *Journal of Systems and Software, 80*(6), pp. 850-861.

Ebert, C. (2009). Software Product Management. *CrossTalk - The Journal of Defense Software Engineering, 22*(1), pp. 15-19.

Gorschek, T. (2006). *Requirements Engineering Supporting Technical Product Management*. Karlskrona: Blekinge Institute of Technology.

Helferich, A., Schmid, K., & Herzwurm, G. (2006). Product management for software product lines: An unsolved problem? *Communications of the ACM, 49*(12), pp 66-67.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly: Management Information Systems, 28*(1), pp. 75-105.

Hietala, J., Kontio, J., Jokinen, J., & Pyysiainen, J. (2004). Challenges of Software Product Companies: Results of a National Survey in Finland. *10th International Software Metrics Symposium*, pp. 232-243.

Hoch, D., Roeding, C. R., Purkert, G., Lindner, S., & Müller, M. (1999). *Secrets of Software Success: Management Insights from100 Software Firms around the World*. Boston: Harvard Business School Press.

Holmes, C., & Ferrill, M. (2005). The application of Operation and Technology Roadmapping to aid Singaporean SMEs identify and select emerging technologies, *Technological Forecasting and Social Change 72*(3 SPEC. ISS.), pp. 349-357.

Höst, M., Regnell, B., Dag, J. N. O., Nedstam, J., Nyberg, C. (2001). Exploring bottlenecks in market-driven requirements management processes with discrete event simulation. *Journal of Systems and Software, 59*(3), pp. 323-332.

Jiao, J., & Chen, C. H. (2006). Customer requirement management in product development: A review of research issues. *Concurrent Engineering Research and Applications, 14*(3), pp. 173-185.

Karlsson, J., & Ryan, K. (1997). A cost-value approach for prioritizing requirements. *IEEE Software, 14* (5), pp. 67-74.

Keil, M. & Carmel, E. (1995). Customer-Developer Links in Software Development, *Communications of the ACM, 38*(5), pp. 33-44.

Kilpi, T. (1997). Product Management Challenge to Software Change Process: Preliminary Results from Three SMEs Experiments. *Software Process - Improvement and Practice, 3*(3), pp. 165-175.

Kilpi, T. (1998). Improving Software Product Management Process: Implementation of a Product Support System. *Proceedings of the 31st Hawaii International Conference on System Sciences, 6*, pp. 3-12.

Koponen, J. (2008). "Agile Release Planning in a Product Backlog Tool", Thesis report: T-110. Department of Computer Science and Engineering: Helsinki University of Technology.

Lee, K., & Kang, K.C. (2004). Feature dependency analysis for product line component design. *Lecture Notes in Computer Science, 3107*, pp. 69-85.

Lehtola, L., Kauppinen, M., & Kujala, S. (2004). Requirements prioritization challenges in practice. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 3009*, pp. 497-508.

Lehtola, L., Kauppinen, M., & Kujala, S. (2005). Linking the business view to requirements engineering: Long-term product planning by roadmapping. *Proceedings of the IEEE International Conference on Requirements Engineering*, pp. 439-443.

Lehtola, L., Kauppinen, M., & Vähäniitty, J. (2007). Strengthening the link between business decisions and RE: Long-term product planning in software product companies. *Proceedings - 15th IEEE International Requirements Engineering Conference, art. no. 4384178*, pp. 153-162.

Lubars, M., Potts, C., & Richter, C. (1993). A review of the state of the practice in requirements modeling. *Proceedings of the First IEEE International Symposium on Requirements Engineering (RE'93)*, pp. 2–14.

Meyer, M. H., Seliger, R. (1998). Product platforms in software development. *Sloan Management Review, 40*(1), pp. 61-74.

Momoh, J., & Ruhe, G. (2006). Release planning process improvement - An industrial case study. *Software Process Improvement and Practice, 11*(3), pp. 295-307.

Natt och Dag, J., Gervasi, V., Brinkkemper, S., & Regnell, B. (2004). Speeding up requirements management in a product software company: Linking customer wishes to product requirements through linguistic engineering. *Proceedings of the IEEE International Conference on Requirements Engineering*, pp. 283-294.

Natt och Dag, J. (2005*). Managing Natural Language Requirements in LargeScale Software Development,* ISSN 1101-3931, ISRN LUTEDX/TETS–1070–SE+222P. Department of Communication Systems. Sweden: Lund University.

Nawrocki, J. R., Walter, B., & Wojciechowski, A. (2002). Comparison of CMM level 2 and eXtreme programming. *European Conference on Software Quality (ECSQ)*, pp. 288-297.

Niazi, M., Wilson, D., & Zowghi, D. (2005). A framework for assisting the design of effective software process improvement implementation strategies. *Journal of Systems and Software, 78*(2), pp. 204-222.

Paasivaara, M., Durasiewicz, S., & Lassenius, C. (2008). Using scrum in a globally distributed project: A case study, *Software Process Improvement and Practice, 13*(6), pp. 527-544.

Paulk, M. C., Curtis, B., Chrissis, M. B., & Weber, C. V. (1993). "Capability Maturity Model for Software". SEI/CMU-93-TR-24, ADA263403. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University.

Phaal, R., Farrukh, C., Mitchell, R., & Probert, D. (2003). Starting up roadmapping fast. *Research-Technology Management, 46* (2), pp. 27-59.

Phaal, R., Farrukh, C., Mitchel,l R., & Probert, D. (2004). Technology roadmapping - A planning framework for evolution and revolution. *Technological Forecasting and Social Change, 71*(1-2), pp. 5-26.

Pine, J. B. (1993). *Mass Customization: The New Frontier in Business Competition*. Boston: Harvard Business School Press.

Pohl, K., Böckle, G., & Linden. F. J. v. (2005). *Software Product Line Engineering: Foundations, Principles and Techniques*. New York: Springer.

Robertson, S., & Robertson, J. (1999). *Mastering the requirements process*. Harlow, UK: AddisonWesley.

Ruhe, G. (2005). Software Release Planning. In S.K. Chang (ed), *Handbook of Software Engineering and Knowledge Engineering – Vol 3.* (pp. 365-394). Singapore: World Scientific Publishing Co.

Ruhe, C., & Saliu, M.O. (2005). The art and science of software release planning. *IEEE Software 22*(6), pp. 47-53.

Sandelowski, M. (1999). The call to experts in qualitative research. *Research in Nursing & Health, 21*(5), pp. 467-471.

Sawyer, P., Sommerville, I., & Viller, S. (1999). Capturing the benefits of requirements engineering. *IEEE Software, 16*(2), pp. 78-85.

Sawyer, S. (2000). Packaged software: implications of the differences from custom approaches to software development, *European Journal of Information Systems, 9*(1), pp. 47-58.

Singla, R. K. (2006). Planning: Meaning and Importance. In R.K. Singla (Eds.), *Business Studies.* (pp. 83-100). New Delhi: V.K. (India) Enterprises.

Srinivasan, V. (in press). An integration framework for product lifecycle management. CAD Computer Aided Design.

Staples, M., Niazi, M., Jeffery, R., Abrahams, A., Byatt, P., & Murphy, R. (2007). An exploratory study of why organizations do not adopt CMMI. *Journal of Systems and Software, 80*(6), pp. 883-895.

Stark, J. (2004). *Product Lifecycle Management: 21st Century Paradigm for Product Realization*. Germany: Springer.

Stelzer, D., & Mellis, W. (1999). Success Factors of Organizational Change in Software Process Improvement. *Software Process Improvement and Practice, 4*(4), pp. 227-250.

Sutherland, J. (2005). Future of Scrum: Parallel Pipelining of Sprints in Complex Projects, *Agile 2005: IEEE CS Press*, pp. 90–102.

Sutherland, J. & Schwaber, K. (2007). "*The Scrum Papers: Nuts, Bolts, and Origins of an Agile Process*" Retrieved November 2009, from Object Technology Jeff Sutherland, http://jeffsutherland.com/scrum/ScrumPapers.pdf.

Vähäniitty, J., Lassenius, C., Rautiainen, K. (2002). An approach to product roadmapping in small software product businesses. *Quality Connection - 7th European Conference on Software Quality (ECSQ2),* pp. 12-13.

Vähäniitty, J. (2004). Product Portfolio Management in Small Software Product Businesses - a Tentative Research Agenda. *Proceedings of the 6th International Workshop on Economic-Driven Software Engineering Research (EDSER-6)*, Edinburgh, Scotland.

Vähäniitty, J., & Rautiainen, K. (2005). Towards an Approach for Development Portfolio Management in Small Product-Oriented Software Companies. *Proceedings of the 38th Hawaii International Conference on System Sciences (HICSS-38)*, pp. 25-28.

Vandermerwe, S. (1995). The Process of Market-Driven Transformation. *Long Range Planning, 28*(2), pp. 79-91.

Vaishnavi, V., & Kuechler, B. (2007). Design Research in Information Systems. Retrieved May 7, 2009, from: AISWorld Net, http://home.aisnet.org/displaycommon.cfm?an=1&subarticlenbr=279

Weerd, I. v., Brinkkemper, S., Nieuwenhuis, R., Versendaal, J., & Bijlsma, L. (2006a). On the Creation of a Reference Framework for Software Product Management: Validation and Tool Support. *Proceedings of the 1st International Workshop on Product Management*, Minneapolis/St. Paul, Minnesota, USA, pp. 312-315

Weerd, I. v., Versendaal, J., & Brinkkemper, S. (2006b). A product software knowledge infrastructure for situational capability maturation: Vision and case studies in product management. *Twelfth Working Conference on Requirements Engineering: Foundation for Software Quality*, Luxembourg, pp. 97-112.

Weerd, I. van de, Brinkkemper, S. (2007). Meta-modeling for situational analysis and design methods. *To appear in the Handbook of Research on Modern Systems Analysis and Design Technologies and Applications*, Idea Group Publishing, USA: Hershey.

Weerd, I. v., Bekkers, W., & Brinkkemper, S. (2009). "Developing a Maturity Matrix for Software Product Management", Technical report: UU-CS-2009-15. The Netherlands: University Utrecht.

Wiegers, K. (1999). First things first: Prioritizing requirements. *Software Development, 7*(9), pp. 48-53.

Xu, L., & Brinkkemper, S. (2005). Concepts of product software: Paving the road for urgently needed research, *Proceedings of the 1st International Workshop on Philosophical Foundations of Information Systems Engineering (LNCS)*, Springer: Berlin, pp. 523-528.

Yin, R. (2003). *Case study research: Design and methods (Vol. 3rd)*. Beverly Hills, California, United States of America: Sage Publishing.

Young, R. R. (2001). *Effective Requirements Practices*. Boston: Addison-Wesley.

Zelkowitz, M. & Wallace, D. (1998). Experimental Models for Validating Technology, *IEEE Computer, 31*(5), pp. 23-31.

Appendix A: Differences

	Custom Information systems	Packaged Software
Industry	- Cost pressures - Success measures: satisfaction, user acceptance, ROI	- Time to market pressures - Success measure: profit, market, share, mind share
Software development	- Staff positions - User is close and more involved - Process is more mature - Separated design and development - Design control via consensus building	- Line positions - User is distant and less involved - Process is immature - Somewhat integrated design and development - Design control via coordination
Cultural milieu	- Bureaucratic - Less individualistic	- Entrepreneurial - Individualistic
Teams	- Matrix managed and project focused - People assigned to multiple projects - Work together as needed - Salary-based - Grow larger over time and tend to disperse - Rely on formal specifications/ documents	- Less likely to have matrix/ project structure. More likely to be self managed - Involved in entire development cycle - More cohesive, motivated, jelled - Opportunities for large financial rewards - Likelier to be small, collocated - Share a vision of their product(s)

Table 15: Adopted from Sawyer (2000).

Development dimension	Custom development	Packaged development
Goal	Software development for internal use	Software development for external use
Typical point at which most customers are identified	Before development begins	After development ends and the product goes to market
Number of customer organizations	Usually one	Many
Physical distance between customer and developer	Usually small	Usually large
Common types of products	New system project; maintenance enhancements	New products; new versions
Terms for software consumer	User; end user	Customer
Common measures of success	Satisfaction; acceptance	Sales; market share; good product reviews

Table 16: Adopted from Keil & Carmel (1995).

Facet	Bespoke development	Market driven development
Primary goal	Compliance to specification	Time-to-market
Measure of success	Satisfaction, acceptance	Sales, market share
Lifecycle	One release, then maintenance	Several releases as long as there is a market demand
Requirements conceptions	Elicited, analyzed, validated	Invented (by market pull or technology push
Requirements specification	Used as a contract between customer and supplier.	Rarely exists in orthodox RE terms, if so, they are much less formal. Requirements are communicated verbally.
Users	Known or identifiable	Unknown, may not exist until product is on market
Distance to users	Usually small	Usually large
Main stakeholders	Customer organization	Developing organization
Specific RE issues	Elicitation, modeling, validation, conflict resolution	Steady stream of requirements, prioritization, cost estimating, release planning
Developer's association with the software	Short-term (until end of project).	Long-term, promoting e.g. investment in maintainability.
Validation	Ongoing process.	Very late, e.g. at trade Affairs.
Use of RE standards and explicit methods	More common.	Rare.
Use of iterative Development	Less common.	More common.
Domain expertise available on the development team.	More common	Less common (product development often breaks new ground).

Table 17: Defined differences from Carlshamre (2002a) with additions of Lubars et al. (1993) and Robertson & Robertson (1999) adopted from Natt och Dag (2005).

RE Phase	Customer specific RE	Market driven RE
Elicitation phase	Acquired from stakeholders using traditional elicitation techniques	Invented by developing team for the First release of the product
Specification phase	Requirements document acts as contract between customer and supplier	Less formal specification, requirements are verbally communicated
Prioritization and negotiation phases	Engage stakeholders to prioritize requirements and negotiate conflicting requirements	Support requirements selection for release planning
Validation phase	Performed together with customer before product is delivered	Product acceptance occurs after product is released in the market

Table 18: Difference between market-driven and customer specific RE, adopted from (Alves et al., 2006)

Areas	Bespoke RE	Market driven RE
Stakeholding	The number of internal stakeholders is low and the external stakeholding (buyer) is limited to one organization.	The number of internal stakeholders is typically large. The number of external stakeholders is also typically large.
Schedule constraints	Time of delivery is agreed upon with the customer.	Time of delivery is set to when the requirements best fit a market-window.
Initiation	RE process commence when customer and developing organization come to an agreement.	RE process is continuous and release projects are initiated based on the requirements or fixed release dates.
Product Lifecycle	Development followed by maintenance.	Product is delivered in increments through successive releases.
Objective	Contractual fulfillment.	Deliver a product that generates the most revenue, by satisfying a larger market share.
Specification	Typically document-based and used as a contractual document.	Requirements are managed individually, typically in a database.

Table 19: Major differences of market-driven and bespoke RE, adopted from Bergström & Dahlqvist (2007).

Appendix B: Situational factors

Situational factor	PM	PR	RP	RM	PM'	PR'	RP'	RM'	Difference
Business unit characteristics									
Development philosophy	1.45	1.64	2.55	2.27	1.10	0.91	0.00	0.28	0.28
Size of business unit team (FTE)	2.18	2.55	3.73	3.27	1.55	1.18	0.00	0.46	0.46
Customer characteristics									
Customer loyalty	4.09	3.91	2.64	2.82	0.00	0.18	1.45	1.27	0.18
Customer satisfaction (1-10)	3.91	4.00	3.27	3.73	0.09	0.00	0.73	0.27	0.09
Customer variability	3.18	3.45	3.55	3.82	0.64	0.37	0.27	0.00	0.27
Number of customers	3.27	3.09	2.18	2.45	0.00	0.18	1.09	0.82	0.18
Type of customers	3.27	3.55	2.45	3.45	0.28	0.00	1.10	0.10	0.10
Market characteristics									
Localization demand	2.00	2.91	2.91	2.27	0.91	0.00	0.00	0.64	0.64
Market growth	3.36	3.09	1.82	1.45	0.00	0.27	1.54	1.91	0.27
Market size	3.18	2.55	1.27	1.27	0.00	0.63	1.91	1.91	0.63
Release frequency (days)	1.64	2.27	3.36	2.82	1.72	1.09	0.00	0.54	0.54
Sector	3.09	3.36	3.27	3.64	0.55	0.28	0.37	0.00	0.28
Standard dominance	1.73	2.27	2.09	2.09	0.54	0.00	0.18	0.18	0.18
Variability of feature requests	2.18	3.09	3.36	3.00	1.18	0.27	0.00	0.36	0.27
Product characteristics									
Defects per year	1.73	1.36	3.00	2.73	1.27	1.64	0.00	0.27	0.27
Development platform maturity	2.64	3.36	3.45	3.55	0.91	0.19	0.10	0.00	0.10
New req. rate per year	2.91	3.00	3.18	3.55	0.64	0.55	0.37	0.00	0.37
Number of products	2.73	2.55	2.27	1.73	0.00	0.18	0.46	1.00	0.18
Product age (years)	4.20	3.90	3.50	3.40	0.00	0.30	0.70	0.80	0.30
Product lifetime (years)	4.55	4.36	3.45	3.36	0.00	0.19	1.10	1.19	0.19
Product size (KLOC)	2.73	2.91	2.55	2.82	0.18	0.00	0.36	0.09	0.18
Product tolerance	1.45	2.18	2.91	2.73	1.46	0.73	0.00	0.18	0.18
Stakeholder involvement									
Company policy	3.45	3.18	2.55	2.55	0.00	0.27	0.90	0.90	0.27
Customer involvement	3.36	3.64	3.18	3.91	0.55	0.27	0.73	0.00	0.27
Legislation	2.36	2.45	3.55	2.82	1.19	1.10	0.00	0.73	0.73
Partner involvement	3.18	3.09	2.27	2.18	0.00	0.09	0.91	1.00	0.09

Table 20: Differences in situational factor weights adopted from Bekkers et al. (2008b).

PM_F1. Product lifetime
The situational factor product lifetime is relevant for the determination of the implementation of the portfolio management processes because the longer the lifetime of a product is the more difficult it gets to manage the lifecycle. However, also the number of products has a big influence on this factor, when a company develops more products the complexity of managing the lifecycles of the products increases.

PM_F2. Market size
Market size is the second important factor for portfolio management because this is the main driver of the entire product portfolio. The bigger the market is the more difficult it is to satisfy the entire market. Also the strategic directions of the product lifecycle and the product line must be based on the market size.

PR_F1. Product lifetime
The situational factor product lifetime is also an important driver for the product roadmapping processes. The product lifecycle and product line are the main inputs for the roadmap, all future market and technology trends are planned according to the lifecycle and product line. A longer lifecycle means a bigger challenge to manage the roadmap.

PR_F2. Customer satisfaction
Customer satisfaction is a relevant factor for the production of the product roadmap because the roadmap is the evidence that specific customer requests are developed and implemented in following releases. By communicating the roadmap with the involved external stakeholders, the customers are acquainted with the features of upcoming releases.

RP_F1. Legislation
The level of influence of legislation is very an important factor for the release planning function. Based on explicit government rules, specific product requirements can have a bigger influence on the prioritization

process. This level is imposed upon the software product by government bodies, which can be strict or loose to none existing. An example of this is tax laws for an administrative product.

RP_F2. Release frequency
The release frequency is based on several processes of the release planning area. When the release frequency has a large amount of days (less product releases a year), the difficulty of managing the releases increases. A longer time span for example can influence the allocation of the available resources, it becomes extra complicated to be feasible with the resources. As a result of that, the possibility of scope change increases.

RM_F1. New requirements rate
The more requirements that emerge, the more difficult it is to manage and control them. This factor is also related to the number of customers, because a greater amount of customers will produce more new potential requirements. By using a database or requirements management tool this process can be simplified.

RM_F2. Customer involvement
This factor is an addition to the previous factor because when customers are more involved in the development of a product the more requirements will emerge during the resuming part of the lifecycle of a product. A high customer involvement requires a proper requirements management and especially an accurate requirements gathering process.

Appendix C: Guidelines from literature

I. General

(Dver, 2003)

1a. The more a PMngr knows about the target market, the applicability of the product versus other solutions, and specifically how the product is used, the more effective a PMngr is (G1);
1b. The PMngr is the first to know when there is a problem and is accountable for overall product decisions (G1);
1c. Manage the entire product line life cycle from strategic planning to tactical activities;
1d. Specify marketing requirements for current and future product (G1);
1e. Bring product to the market with a company-wide go-to-market plan, working with all departments to execute (Not used);
1f. Analyzing potential partner relationships for the product along with business development (Not used).

(Ebert, 2007)

2a. The PMngr is responsible for product requirements, release definition, product release lifecycles, creating an effective multifunctional product introduction team and preparing and implementing the business case (G1);
2b. The PMngr is a "mini CEO" representing the enterprise or business unit in strategy definition and operational execution (G1).

(Niazi et al., 2005)

Niazi et al. defined a list of critical success factors, which are linked with different implementation phases of the SPI.

Phase	Critical Success Factor
Awareness	Senior management commitment, staff involvement, SPI awareness.
Learning	Senior management commitment, training and mentoring, and SPI awareness.
Pilot implementation	Senior management commitment, creating process action teams, experienced staff, and defined SPI implementation methodology.
SPI implementation action plan	Senior management commitment, experienced staff, defined SPI implementation methodology, and reviews
SPI implementation across the organization	Senior management commitment, staff time and resources, staff involvement, experienced staff, SPI awareness, defined SPI, and implementation methodology,
Maintenance	Senior management commitment, reviews, and training and mentoring

Table 21: List of phases with critical success factors for SPI (Niazi et al., 2005).

(Stelzer & Mellis, 1999)

Stelzer & Mellis identified a list of critical success factors.

Rank	Critical Success Factor
1	Management commitment and support
2	Staff involvement
3	Providing enhanced understanding
3	Tailoring improvement initiatives
5	Managing the improvement project
6	Change agents and opinion leaders
6	Stabilizing changed processes
8	Encouraging communication and collaboration
9	Setting relevant and realistic objectives
10	Unfreezing the organization

Table 22: List of ranked critical success factors for SPI (Stelzer & Mellis, 1999).

II. Requirements Management

(Sawyer et al., 1999)

1a. Use a database to manage requirements (RM2);
1b. Uniquely identify requirement (RM2);
1c. Define policies for requirements management (RM2);
1d. Define traceability policies, and maintain the tractability manual (RM3);
1e. Record rejected requirements (RM2).

(Natt och Dag et al., 2004)

2a. Make the distinction between market requirements and business requirements (RM3);
2b. Link market requirements with business requirements (RM3);
2c. Use standard (template) to describe requirements (RM2);
2d. Store all gathered requirements on one location and use a tool for managing the requirements (RM2);
2e. Define policies for requirements management (RM2).

(Jiao & Chen, 2007)

3a. Gather requirements from customers and a combination of stakeholders including the environment, feasibility studies, market analyses, business plans, and benchmarks of competing products (RM1);
3b. Derive explicit requirements that can be understood by marketing and engineering (RM3);
3c. Specify concrete product specifications in the functional domain (RM3);
3d. Considers efforts in capturing the genuine or "real" needs of the customers, rather than too much focus on the technological specification (RM1);
3e. Clearly describe requirements, often they are poorly understood and expressed in abstract, fuzzy or conceptual terms (RM2);
3f. Prioritize customer preferences with respect to a set of customer requirements (Not used);
3g. Classify requirements to help guide the designer in compiling, organizing, and analyzing product design issues (Not used).

III. Release Planning

(Berander & Andrews, 2005)

1a. Establish relative importance of each requirement to provide the greatest value at the lowest cost, balance the business benefit of each requirement against its cost (RP1);
1b. Plan and select an ordered, optimal set of software requirements for implementation in successive releases (RP2);
1c. Estimate expected customer satisfaction (Not used);
1d. Minimize rework and schedule slippage (plan stability) (RP2);
1e. Select only a subset of the requirements and still produce a system that will satisfy the customers (RP2);
1f. Trade off desired project scope against sometimes conflicting constraints such as schedule, budget, resources, time to market, and quality (RP2).

(Ruhe, 2005)

2a. Consider two dimensions of priority (RP1):
- Value addresses the assumed impact on the value of the final product;
- Urgency addresses the time-to-market aspect, to reflect market needs and competitor analysis information.

2b. Release plans have to be updated frequently due to changing environment (RP3);
2c. The involvement of the stakeholders is important during the planning process (RP3).

(Ruhe & Saliu, 2005)

3a. Provide maximum business value by offering the best possible blend of features in the right sequence of releases (RP3);
3b. Satisfy the most important stakeholders involved (Not used);
3c. Be feasible with available resources (RP2);
3d. Reflect existing dependencies between features (optimal development of release) (RP1).

(Sawyer et al., 1999)

4a. Identify global system requirements (RP2);
4b. Define change management policies by providing a framework for systematically assessing change (RP4);
4c. Identify volatile requirements, to simplify scope change (RP2).

(Momoh & Ruhe, 2006)

5a. The roadmap was used as a strategy and planning document for the organization (RP3);
5b. Derive a prioritized list of requirements to enable the design effort to be focused on areas that will make the maximum contribution to satisfying the stakeholders (5b);
5c. Prepare the initial estimation of the resources, time, effort, and cost required to completely deliver each of the requirements (RP2);
5d. The estimations and the assumptions must be documented, reviewed, and validated by the stakeholders (RP3).
5e. Changes to a release plan's content or schedule are reported to the stakeholders (RP3).
5f. Set up and evaluate the business case and the major business drivers (RP3).

IV. Product Roadmapping

(Lehtola et al., 2007)

1a. Focus is usually on features of one product (PR2);
1b. Link from business decision to requirement engineering decisions not explicit (PR3);
1c. Typical planning horizon is a few releases ahead (PR2);
1d. Preparation of roadmaps is mostly the product managers' responsibility (PR3).

(Lehtola et al., 2005)

2a. A roadmap must strengthen the link between business decisions and requirements engineering (PR2);
2b. Use a roadmap as a tool for communicating ideas to other stakeholders (PR3);
2c. Emphasize developer viewpoint (Not used);
2d. Don't let the roadmap get outdated (PR2);
2e. Tie product development resources to roadmaps (PR3);
2f. Create product roadmaps for a shorter time period (PR2).

(Phaal et al., 2004)

3a. Find a balance between a market pull and a technology push approach (PR2);
3b. Consider the following factors while creating the roadmap (PR3)
- the level of available resources including (people, time, budget);
- nature of the issue being addressed (purpose and scope);
- available information (market and technology);
- other processes and management methods that are relevant (strategy, budgeting, new product development, project management and market research);

3c. Perform product and market analysis (Not used);
3d. Evaluate product and technology options (PR3);
3e. Identification of technology: available / feasible / possible (Not used).

(Lee & Kang, 2004)

4a. Understanding commonality, variability and dependency from a domain perspective is essential for developing reusable assets (PR1).
4b. Assets must be designed so that inclusion or exclusion of variable features causes little changes to components implementing other features (Not used).

(Ebert, 2007)

5a. Describe and maintain a more detailed technology roadmap (Not used);
5b. Roadmap shows which functions arrive and what their dependencies are (PR2);
5c. Decide and communicate within the entire company, which products, platforms, features or even markets are active (PR3).

V. Portfolio Management

(Cooper et al., 2001)

1a. Define arenas for focus such as the key markets, technologies and product types that the development effort will focus on (PM2).
1b. Make strategic choices on the type of products, markets and technology (PM3);
1c. Allocate resources for the R&D, engineering, and marketing operations (PM3);
1d. Make sure that there is no pipeline gridlock in the portfolio, so undertake projects on time and in a time efficient manner (PM3);
1e. Create a balanced portfolio of high value projects. High risk versus low risk and across markets and technologies (PM3).

(Stark, 2004)

2a. Determine what kind of approach will be used (PM1);
- Strategic enterprise-wide initiative which targeting new market-leading products and full control across the product lifecycle;
- Cross-functional projects to achieve tactical benefits (by implementing new lifecycle processes across several functions;
- Targeting some very precisely defined improvements to achieve benefits in specific operational areas;

2b. Communicate PLM both internally and externally with stakeholders (PM3);
2c. Understand the lifecycle of a product (PM3);
2d. Make a clear link between PLM and the business strategy (PM3);
2e. Involve the customer and listen to product feedback (PM3).

(Clements et al., 2005)

3a. Make sure that current and planned products are positioned to take advantage of upcoming technology trends (PM3);
3b. Decide what products / domains are in the product line (PM1);
3c. Establish and tracking product line goals / scope (PM1);
3d. Determine new strategic directions (PM3);
3e. Establish organizational readiness (Not used)

(Srinivasan, in press)

4a. Collaborate with partners which share and exchanging product information, and integrate their engineering and business decision support systems (PM3);
4b. Consider a make-or-buy decision before collaborating with partners (PM3).

Appendix D: Assessment

I. General questions

The general questions section consisted of a small discussion about the current SPM practice of MP Objects. The main challenges which were mentioned are:

- Requirements are not registered properly, at this moment MP Objects is not using a requirements management tool or database to manage requirements.
- Not enough product and technical documentation.
- Difficulties in linking requirements to software artifacts.
- Difficulties in requirements traceability.
- The company used to develop customized software. The change process from customized software to product software needs more structure.
- Not clear which deliverables have to be produced in which part of the SPM process.

The participants have 1 to 2 years experience with SPM.

II. Situational factors

Characteristic	Situational factor	Value
Business unit	Development philosophy	Iterative
	Size of business unit team (FTE)	13
Customer	Customer loyalty	High
	Customer satisfaction (1-10)	8
	Customer variability	10%
	Number of customers	1252 users, 59 companies, 5 contracting parties
	Type of customers	Small companies, Medium companies, Large companies
Market	Localization demand	Netherlands, England, Italy
	Market growth	Growing
	Market size	1500–3500 customers
	Release frequency (days)	Per quarter, 90 days
	Sector	Logistic
	Standard dominance	Medium
	Variability of feature requests	High
Product	Defects per year	40 per year
	Development platform maturity	Ever changing
	New requirements rate per year	250 per year
	Number of products	2 products: EDO, Supply Chain Suite
	Product age (years)	4 years
	Product lifetime (years)	>15 years
	Product size (KLOC)	320.000 executable statements
	Product tolerance	Medium
Involvement of the stakeholders	Company policy	High
	Customer involvement	High
	Legislation	Loose
	Partner involvement7	Low

Table 23: Answers situational factor questions results MP Objects.

III. Maturity questions

	0	1	2	3	4	5	6	7	8	9	10	11	12
RM1. Requirements gathering		A		B				C		D			
RM2. Requirements identification			A			B			C				
RM3. Requirements organizing					A		B		C				
RP1. Requirements prioritization		A				B		C				D	
RP2. Requirements selection			A			B			C		D		
RP3. Release definition				A			B			C			
RP4. Release validation				A			B		C				D
RP5. Launch preparation					A			B			C		
RP6. Scope change management					A				B		C		
PR1. Theme identification				A	B					C			
PR2. Core asset identification				A					B				C
PR3. Roadmap construction			A			B				C			
PM1. Market trend identification			A		B				C				
PM2. Partnering & contracting			A		B					C			
PM3. Product lifecycle management			A			B		C					
PM4. Product line identification			A			B		C					

Table 24: Answers maturity questions MP Objects.

Note: The red line is the mode score of the results from an earlier internet survey among product managers in January / February 2009.

IV. SPM Productization

Area	Step 1	Step 2	Step 3	Step 4	Step 5a	Step 5b
Software	Customized system; reuse of features across projects	Customized; large amount of feature reuse, recognizing product	Standardized, product platform; large customized layer per customer	Standardized, emerging product; customized layer per customer	Standardized product; customizable for customers by adding small customized layer	Standardized product; product is fully configurable
Sequel	Maintenance per customer	Maintenance per customer	Maintenance per customer (including product platform)	Event based (customer specific) releases; designed per customer	Releases based on fixed release dates; equal for all customers; not customizable	Releases based on fixed release dates; for entire market
Requirements origin	Elicited for each customer	Gathered from all customers	Gathered from all customers and start involving internal stakeholders	Gathered from all customers, internal stakeholders, and start looking at market trends	Gathered from entire market; all internal stakeholders and all customers	Gathered from entire market; all internal and all external stakeholders
Requirements selection	Select all customer requirements per project (more or less fixed list)	Select all customer requirements per project (more or less fixed list)	Select requirements for product platform and select (large quantity of) customer requirements	Focus on selecting requirements for product (roadmap based) and subset of customer requirements	Optimal selected subset of market requirements (roadmap based) and requirements for customization	Optimal selected subset of market requirements (roadmap based)
Main stakeholder involvement on development	High external; barely any internal	High external; low internal	High external; medium internal	Medium external; medium internal	Medium external; high internal	Low external; high internal
Objective	Satisfy customer and focus on return on investment	Satisfy customer but start looking at market needs and identify a certain product	Satisfy customer but start creating standardized platform for market (focus on long term)	Focus at bringing product to the market and still try to satisfy customer	Bring product to the market and provide support services (customizable layer)	Bring product to the market and increase market share by selling licenses

Table 25: Initial position for the productization for MP Objects.

Note: This part is carried out later on during this study.

Appendix E: Process Delivery Diagram

I. Activity Descriptions

Portfolio management

Activity	Sub-activity	Explanation
Product Lifecycle Management	Weekly developers meeting	Every week all the developers discuss the status of the development of the releases and the product. During these meeting specific features or changes in the PRODUCT LIFECYCLE can emerge. Only internal stakeholders are involved during these sessions.
Product line identification	Determine influence on product line	Based on the changes in the product lifecycle, there is a possibility that also the PRODUCT LINE changes. The product line is not actively managed and monitored.
Partnering & contracting	Consider partnering and contracting	When there are changes in the product line there is the opportunity to consider products or services of other companies. This make-or-buy decision is of essence for the further development of the product.
	Plan changes	When the decision is made to 'make' a specific part, function, or technical improvement for the product, the development needs to be planned in upcoming releases. When the changes are too complex and large, there is a possibility that a specific project is created to apply it.
	Retrieve contract	On the other hand when the decision is made to 'buy' a specific part, function, product, or service an agreement with the vendor needs to be determined. This agreement is also described in a CONTRACT.

Table 26: Activity descriptions of the PDD for portfolio management

Release planning

Activity	Sub-activity	Explanation
Requirements gathering	Call for calls	Before creating a new release, all customers are able to provide their last (prioritized) requests for the upcoming release. All the incoming requirements are gathered and identified like in the RM function. This results into a LAST REQUIREMENTS LIST. The difference here is that the 'call for calls' is a process based on a single sequence, where the RM function is continuous process.
Requirements organizing	Organize requirements per function	After that all requirements are gathered from the stakeholders, they are organized in an ORGANIZED REQUIREMENTS LIST. Within this list the requirements are organized per customer or as general improvements.
Requirements prioritization	Prioritize requirements	After organizing the requirements they are prioritized for a release into a PRIORITIZED REQUIREMENTS LIST. Mainly the prioritization is based on the results of the customers' last call for calls. Additionally, also the changes from the internal stakeholders are prioritized.
Requirements selection	Consider capacity	Before selecting the requirements for a release, first must be determined what capacity is required to develop the requirements of the prioritized requirements list. This results in a set of requirements and it is possible that specific requests cannot be integrated within a release.
	Inform involved stakeholder	When specific requirement cannot be integrated (because it requires too much effort or it is to complex) in a release, the involved stakeholder of that particular requirement is informed.
	Eliminate requirement from this release	After informing the customer, the specific requirement(s) are eliminated from the prioritized requirements list which means that they will not be developed in the release.
	Select subset of requirements	When the selection procedure is finished, the SELECTED REQUIREMENTS LIST is created. This list is a subset of the requirements prioritization list and consists of all of the requirements which will be developed in the release. Partially, this list consists of (at least) the first three requests from all customers and additional requests from the internal stakeholders.
Release definition	Determine resource	Based on the selected requirements the required resources are determined and specified; this decision is based on the total number of selected requirements and the complexity of each of the requirements. Within MP Objects there is the possibility to allocate the resources in a flexible matter.
	Determine time path	Parallel with the determination of the resources, also the entire time path of the development of the release is determined. The results of these two actions are used for the creation of the release definition.
	Generate release definition	The RELEASE DEFINITION is created by combining the SELECTED REQUIREMENTS LIST with the results of the determination of the resources and time path.
Release validation	Validate release definition	This activity consists of the validation of the release definition by the involved (internal) stakeholders. If it is not accepted then the sequence restarts with the selection of the subset of requirements. When it is approved the development of the release begins.

Scope change management	Develop release	When the release definition is accepted the development of the RELEASE is executed. This process takes four months of work, after that the development of the next release starts.
	Change scope	If during the development of the release unforeseen events occur there is a possibility that it is necessary to change the scope of the release. If so, the sequence continues at the release definition phase and an alternative subset of requirements is selected.
Launch preparation	Inform the customer	When the development of the release is finished and ready to be launched, the customers are informed via an email. This email is sent after that the release is tested and the board approved to launch the release.
	Implement release	When the release is finished, it is implemented at the current customers. After that the sequence for developing a new release starts again. During this process a template is used which consists of: specific customer information, what to prepare, what to do on the day of implementation, and what to do after the implementation (after care).
	Integrate release	When the product is sold to a new customer, the implementation of the release is performed by integrating the release in the current infrastructure. This entire integration process takes more time and therefore is this customer is not eligible for the next release.
	Acceptance	After implementing or integrating the release, there is an acceptance phase in which the customers are able to validate the end result and accept the product.

Table 27: Activity descriptions of the PDD for release planning.

Requirements management

Activity	**Sub-activity**	**Explanation**
Requirements gathering	Gather customer requirement	The customers' REQUIREMENTs are specific requests or wishes from existing customers. These wishes can consist of new functionalities, changes or other requests.
	Gather technical request	Also from the technical point of view there can emerge new REQUIRMENTs, for example bug fix requests or technical improvement.
	Gather consultant requirement	Finally the consultant can also have specific REQUIREMENTs. These are related to market requests, market demands, specific functionalities, or other wishes.
Requirement identification	Identify additional requirements	The gathered requirements are the basis for the identification of new emerging (additional) requirements, such as interface requirements, technical requirements, or other specifications. MP Objects make for the customers' changes a CHANGE QUOTATION; this determines the required development time and additional costs.
	Add requirements to the list	All the requirements need to be added to the REQUIREMENTS LIST; additionally this means that there can be dependencies between requirements in place.
	Inform stakeholder	Only the external stakeholders need to be informed about the Change Quotation information.
	Remove from list	When a requirement cannot be implemented in the product, this requirement and related requirements should be removed from the REQUIREMENTS LIST.

Table 28: Activity descriptions of the PDD for requirements management.

II. Concept Definitions

Portfolio Management

Concept	**Definition**
PRODUCT LIFECYCLE	The lifecycle consists of "the sum of all activities needed to define, develop, implement, build, operate, service, and phase out a product and its related variants" (Ebert, 2007). In terms of MP Objects, only the short-term decisions are discussed and barely documented. The long-term decisions are not actively monitored and not documented at all.
PRODUCT LINE	A product line "is a family of software intense products that share a common set of features, where each individual product is then developed from a common set of assets in a controlled manner" (Clements & Northrop, 2002).
CONTRACT	After the decision is made to buy something from another software vendor, a contract is retrieved which consists of the agreements related with services, releases, and other relevant information.

Table 29: Concept definitions of the PDD for portfolio management.

Release Planning

Concept	**Definition**
LAST REQUIREMENTS LIST	This list of requirements consists of the last (prioritized) list of requirements for the upcoming release. The customers are able to specify which request they want to have in the upcoming release.
ORGANIZED REQUIREMENTS LIST	Within the organized requirements list, the requirements are classified in an Excel document per 'customer'. When requirements are applicable for the product in general, they are

	organized as 'general improvements'.
PRIORITIZED REQUIREMENTS LIST	After the organization of the requirement they are prioritized for the release. This prioritization is supported by the customers' preferences.
SELECTED REQUIREMENTS LIST	Based on the capacity of the resources, requirements must be selected for the upcoming release. While selecting, for all the customers (at least) the three highest prioritized requests are selected.
RELEASE DEFINITION	The release definition (release calls) is a document which consists of the selected list of requirements which are within the release. Additionally, also the allocation of the necessary resources and a detailed time path are included within this document. There is a possibility that a release is customized for a customer, than this is clearly stated within the release definition.
RELEASE	A release is a new version of "software products which are often offered to a market, which consists of a significant increase in functionality" (Xu &Brinkkemper, 2005). MP Objects launches a new release every quarter.

Table 30: Concept definitions of the PDD for release planning.

Requirements Management

Concept	**Definition**
REQUIREMENT	All the requests from the customer, consultants, and developers are handled as a requirement. A requirement "is a singular documented need of what a particular product or service should be or do. It is a statement that identifies a necessary attribute, capability, characteristic, or quality of a system in order for it to have value and utility to a user" (Young, 2001);
REQUIREMENTS LIST	The requirements list consists of a bigger quantity of requirements depicted from all internal and external stakeholders. This includes additional requirements which are defined based on the gathered requirements.
CHANGE QUOTATION	The change quotation consists of the detailed information of a specific customer change; this involves an estimation of the required resources and costs to develop it.

Table 31: Concept definitions of the PDD for requirements management.