SPM Syllabus
Excellence Level
Product Planning
V.1.1 Student Edition

© ISPMA 2016

This syllabus was written by the following members of the International Software Product Management Association (ISPMA): Samuel Fricker, Gerald Heller, Hans-Bernd Kittlaus, Guenther Ruhe, Krzysztof Wnuk.

We thank all honorary authors and contributors.

Copyright © 2016 for this syllabus is with the authors mentioned above. The rights have been transferred to the ISPMA International Software Product Management Association. Licensing by other standardization bodies is possible upon written acceptance.

This document is provided for educational purposes. ISPMA does not warrant that it is suitable for any other purpose and makes no expressed or implied warranty of any kind and assumes no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information contained herein.

Terms of Use:

1. Individuals and training providers may use this syllabus as a basis for seminars, provided that the copyright of the authors and ISPMA is acknowledged and included in the seminar materials.
2. Any individual or group of individuals may use this syllabus as a basis for articles, books, or other derived publications, provided that the copyright of the authors and ISPMA as the source and owners of this document is acknowledged in such publications.

Any inquiries regarding this publication, requests for usage rights for the material included herein, or corrections should be sent by email to info@ispma.org.

Preface

The goal of the International Software Product Management Association (ISPMA) syllabus for "ISPMA Software Product Manager Excellence in Product Strategy" is to promote in-depth understanding of the *discipline of product management for software products* including the management of software parts of software-intensive products, i.e. systems or services, in the area of product strategy.

The Excellence Level syllabus "SPM: Product Planning" covers the full spectrum of elements of software product management related to product strategy that are well supported by literature and industrial practice. The syllabus corresponds to a 3-day industry course.

The syllabus addresses the needs of people involved in software product management, and helps them to address the needs of people they interface with, e.g. general management, marketing and sales, research and development, service and support, and controlling.

The syllabus is the basis for examination to certify that the examinee has achieved the degree of knowledge described in this syllabus. The terms used in this syllabus are consistent with the glossary of the ISPMA.

Purpose and structure of the Syllabus:

The syllabus is the basis for consistent training, learning, and examination of software product management. It provides:

- Explicitly phrased educational objectives for each chapter, and
- Informal explanations to detail the educational objectives.
- Informal references to literature (without limiting the interpretation of the syllabus to this literature only).

This syllabus consists of six chapters. Each chapter covers one major educational unit (EU). Each chapter also includes the duration suggested to teach it. Each educational unit has educational objectives (EO) that are enumerated following the chapter headers (EO1.1.1, EO1.2.1 …). An educational objective has a defined cognitive level of knowledge that the course participant is expected to achieve. The numbering scheme for these objectives is aligned with the chapter numbering.

The educational objectives are expressed in terms of three cognitive levels of knowledge phrased using the verbs "knowing" for level 1, "understanding" for level 2, and "applying" for level 3. These three verbs are placeholders for the following:

- L1 (know): enumerate, characterize, recognize, and name.
- L2 (understand): reflect, analyze, justify, describe, judge, display, complete, explain, elucidate, elicit, formulate, identify, interpret, reason, translate, distinguish, compare, understand, suggest, and summarize.
- L3 (apply): perform, execute, develop, and adapt.

Each EO in the syllabus has one of the three cognitive levels assigned to it.

In order to address L3 objectives, ISPMA's Excellence Level syllabi are designed to put special focus on exercises. It is the trainer's responsibility to select exercises and to define concrete realistic scenarios in which all the selected exercises can be performed by the participants. ISPMA recommends to spend about 50% of the available time on exercises. In trainers' material, exercises are described in abstract terms.

Included and excluded key areas:

The syllabus covers knowledge applicable for any kind of software systems and organizational contexts. A training course may cover more domain-specific details if needed by the course participants. This syllabus, however, does not provide guidance for such specialization, rather describes the base knowledge necessary, which can be complemented with domain specific items.

The syllabus is independent of any specific process model, and thus defines knowledge of a software product manager without prescribing exact interfaces to other roles in a product organization.

Training Courses:

The syllabus corresponds to a three-day industry course. The syllabus does not prescribe the specific form and approach of learning, however. It can also be administered with other forms of teaching or learning, such as self-learning supplemented by coaching or courses at universities or universities of applied sciences.

Training providers are encouraged to tailor training courses to the participants, and to add examples and an appropriate realistic scenario for the exercises described in this syllabus so that participants get an opportunity to apply the training contents. A participant should carefully choose the training provider. A list of training providers can be found on the ISPMA web site www.ispma.org.

Examination:

The syllabus is the basis for the examination for the ISPMA certificate "ISPMA Software Product Manager Excellence in Product Planning". All chapters are relevant for the exam. The exam takes the following form:

- Demonstration of knowledge with a multiple-choice test

Multiple-choice tests can be held immediately after a training course, but also independently from courses (e.g. publicly announced exams of the examination authorities). A list of accredited examination authorities can be found on the ISPMA web site www.ispma.org.

Course participant prerequisites:

The training and learning of the syllabus assumes general knowledge of, and some experience in, the management or development of software products or software in software-intensive systems.
The formal background of the course participant is not crucial (whether it be engineering or management), rather the level of experience is predominantly the factor associated with the prerequisites. A course participant should have the ISPMA Foundation Level Certificate "ISPMA Certified Software Product Manager" or at least three years of experience in software product management. However, this is a generic recommendation and might not be applicable for all situations or course participants.

Table of Contents

EU1 Product Planning in Context

Duration: 3:00h

Educational Objectives:

EO1.1 Understand the role of product planning in operationalizing product vision and strategy

EO1.2 Be able to drive the implementation of product strategy and vision in the product planning process

EO1.3 Understand the relationship between product planning and orchestration

EO1.4 Understand the dependency of the product planning approach on the product scenario

Product Planning encompasses Life Cycle Management, Roadmapping, Release Planning, and Requirements Engineering (see ISPMA's SPM Framework in Fig. 1).

Strategic Management	Product Strategy	Product Planning	Development	Marketing	Sales and Distribution	Service and Support
Corporate Strategy	Positioning and Product Definition	Product Life-Cycle Management	Engineering Management	Marketing Planning	Sales Planning	Service Planning and Preparation
Portfolio Management	Delivery model and Service Strategy	Roadmapping	Project Management	Customer Analysis	Channel Preparation	Service Provisioning
Innovation Management	Sourcing	Release Planning	Project Requirements Engineering	Opportunity Management	Customer Relationship Management	Technical Support
Resource Management	Business Case and Costing	Product Requirements Engineering	User Experience Design	Marketing Mix Optimization	Operational Sales	Marketing Support
Market Analysis	Pricing		Quality Management	Product Launches	Operational Distribution	Sales Support
Product Analysis	Ecosystem Management			Operational Marketing		
	Legal and IPR Management					
	Performance and Risk Management					
Participation	**Core SPM**		**Orchestration**			

Fig. 1 SPM Framework V. 1.2

This framework reflects the artifact hierarchy of software products and their management:

1. Portfolio — Portfolio Management (in Strategic Management)
2. Product — Product Life Cycle Management + Roadmapping
3. Release — Release Planning
4. Requirement — Product Requirements Engineering

Product planning means developing a plan with associated actions for implementing the product vision and strategy and its execution. The time horizon and granularity of these activities go from strategic, coarse-grained (roadmap) to tactical, fine-grained (revitalization actions, individual product requirements). The vision describes what the future product will be, why it is needed, and why it will be successful. It helps to convince and engage the product's stakeholders to contribute to the success of the product, and it is a natural starting point for product strategy definition and execution. The product strategy documents decisions that turn the vision into a manageable and executable path into the future. Together, the product vision and strategy give direction. This syllabus assumes that a product vision and a product strategy have been created and can be used as an input to product planning (see ISPMA's Excellence Level Syllabus on Product Strategy).

The product strategy contains the following elements that are used by the product manager for product planning. Product strategy and product plan are interdependent and evolve as the product plan is elaborated and the product is developed, released, evolved, and maintained:

- Product definition: functional scope, quality scope, intended use and intended users
- Product positioning: value definition, channels
- Target market and potential market segments: scope, size and driving forces, competitive analysis, opportunities, segmentation and persona creation
- Delivery model: standard product vs. customizable product vs. licensed product vs. SaaS
- Sourcing: make or buy decisions
- Business Plan: business model, costing, pricing, forecasting

The roadmap is the bridge between product strategy and product planning. It shows the steps in which the strategy is intended to be implemented over the strategic timeframe. Product planning lays the basis for the implementation of the plan by the company. For a product manager, orchestration means ensuring that all relevant units within the company support and contribute to the product's success in the best possible way (see Excellence Level Syllabus on Orchestration). The contribution from development, marketing, sales, or services is needed for strategy, planning, and implementation (see Fig. 1).

Product planning methodology depends on the degree of vendor control, and on the product's life cycle phase.

Vendor-controlled means that the software vendor decides which changes to the software product are made when in the runtime environment, i.e. during execution. Vendor-controlled is typical for relatively unregulated environments like B2C internet platforms, and for SaaS or B2C license products and embedded software that offer automated maintenance over the internet. Customer-controlled means that customers want to supervise the runtime environment, often for quality or regulatory concerns. Customer-controlled is typical for B2B software license products, but can also be found with B2B SaaS, in particular when tied to business process outsourcing that is considered as business-critical. It also applies to software embedded products where the customer decides if and when the product is given to the vendor or his partners for maintenance.

Software Product Scenarios		Life Cycle Phase	
		New Product Development	Existing Product Evolution
Runtime Environment	Vendor-Controlled	Powerboat	Speedboat
	Customer-Controlled	Icebreaker	Cruise Ship

Fig. 2 Software Product Scenarios (*Kittlaus (2015*))

With new product development ("Conception and Creation" phase in ISPMA's Foundation Level syllabus), there is a high level of uncertainty and risk. This situation encourages a fast release of a minimum viable product. Once the product is rolled out ("Market Introduction", "Growth", and "Maturity" phases), the focus shifts to extending the product scope and target market. Compatibility and migration aspects become relevant here. While roadmapping is important in all scenarios, release planning and product requirements engineering work differently in the four resulting scenarios (see Fig. 2) which we have chosen catchy names for:

- Powerboat – New products for vendor-controlled environments (such as Amazon before its initial launch): SPM focuses on defining the minimum viable product (MVP) for the first customers by evaluating the product idea. The definition of the MVP requires a close link with development for extensive prototyping.
- Speedboat – Evolved products for vendor-controlled environments (such as Google Docs): SPM focuses on extending the product scope and thereby increasing the target market. This scaling of the product requires ongoing analysis of the actual usage of the product, of the market, and of competition, e.g. by monitoring customer feedback and product use. The release frequency may be high, and updates to the software product tend to be automated.
- Icebreaker – New products for customer-controlled environments (such as Docker before its first release): SPM focuses on defining the minimum viable product for the first customers by analyzing the customers' business processes. A major SPM task ensures that requirements are sufficiently generalized so that the first release does not become too specific for a few customers only.
- Cruise ship – Evolved products for customer-controlled environments (such as the Oracle database): SPM focuses on extending the product scope and thereby increasing the target market. This scaling of the product requires ongoing analysis of requirements, market and competition, e.g. by directly collaborating with customers. Since customers do not want to test and install new releases often, the frequency of releases is rather low, often one or two per year.

Real-world situations may be of hybrid character. An organization may decide to offer both a licensed product and a SaaS product with a largely overlapping code base. Also, a product may consist of a cloud component and an on-premise component. In these hybrid scenarios, organizations often follow the different pure-scenario approaches in parallel for the different components but need to take care of resulting conflicts.

References

C. Alvarez (2014); M. Grath (2000); H.-B. Kittlaus (2015); H.-B. Kittlaus, P. Clough (2009); G.A. Moore (2014); I. van de Weerd e.a. (2006)

EU2 Product Life Cycle Management

Duration: 3:00h

Educational Objectives:

EO2.1 Know the product life cycle

EO2.2 Know the category life cycle

EO2.3 Understand product planning options in the context of the product/category life cycle constellations

Product Life Cycle

A software product evolves over a series of phases. Each life cycle phase offers a distinct set of options for evolving the product. For example, common options are extensions of the product with new features, integration of the product with partner products, or additional product services. A product manager needs to select the right options to achieve the right business results. Also, each life cycle phase is associated with a set of leading stakeholders that influence the product and need to be considered for product planning.

Overall, a product moves through six life cycle phases. Typically, the first three phases are investment phases. Products in later phases serve as cash cows for a company. However, agile approaches combined with a minimal viable product strategy may yield a faster return on investment by generating cash flow much earlier.

Business outcome	**Phase**	**Focus Areas**
Investment	Conception and creation	Innovation, positioning
	Market Introduction	Launch, market share
	Growth	Market share, functionality
Cash Cow	Maturity	Revitalization, service
	Decline	Customer retention
	Withdrawal	Customer retention, cost reduction

Product management must have a solid understanding of the various life cycle phases to develop strategies and activities that optimally support a product in a specific phase. Data from product analysis and market analysis will be used to determine the current product life cycle phase. Challenging decisions are required when moving from one phase to another. For example, within the decline phase, a product manager might decide to move support services to a low-cost country. Methods that worked well have to be changed to prepare for the next phase. This change often creates irritation and resistance within an organization.

Category Life Cycle

According to Moore, a category is a way that customers classify what they are buying and distinguish it from other purchasing choices. A category has its life cycle, which is independent of the life cycle of an individual product. Figure 3 provides an overview (*G. A. Moore (2005)*). The category life cycle phase affects the individual product life cycle.

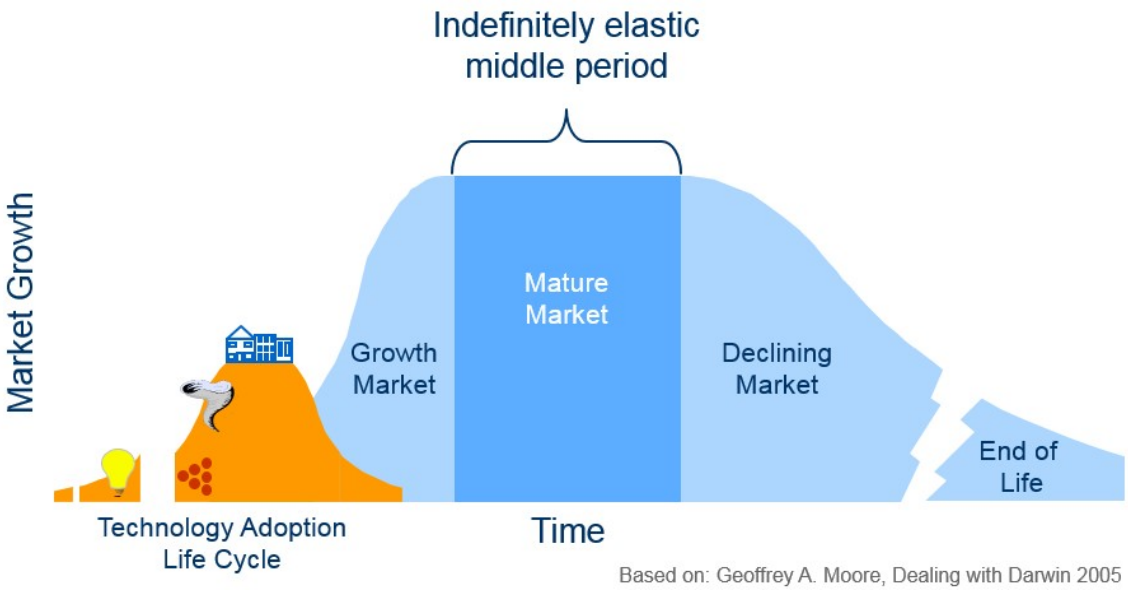

Fig 3. Category Life Cycle

The technology adoption phase describes a market that starts to exist. Disruptive technologies appear in the market and define a new category. Early adopters are willing to experiment with the new technology. If the products fail to meet the expectations, the category will cease to exist.
If the products succeed to meet expectations, the market category will grow. A category that is in its growth phase typically shows double-digit growth rates year over year. In 2015, examples of such growth rates are 3D printers and social media software products. Mature market phases may last for decades. The declining market phase is characterized by replacement technology of a new market category that takes away market share from the existing category. A product market category is at end-of-life when a new disruptive technology has replaced an existing product category.

Product Planning Options

A product manager has to understand to which category his product belongs and in which phase this category currently is to pick the appropriate options for product planning. In combination with the product life cycle phase, this information will influence product planning.

Managing products within the “technology adoption” phase involves not only intense communication with early adopters, but also development and deployment approaches that allow rapid evolution of the product. Such a product planning option is facilitated by the Powerboat scenario. A product manager in the Icebreaker scenario should also strive to find ways to evolve and

adapt the product to market needs. Alternatively, a product manager should evaluate whether and how the Icebreaker approach may be switched to a Powerboat approach.

Creating a new product for a mature market requires the selection of approaches to take away market share from existing products. The options correspond to the following four types of incremental innovation (*G.A. Moore (2005*)):

Product line extension broadens the product offering by specialization. Variants of the product are created which target specific businesses and provide a better fit for that group.
Enhancement innovation optimizes the product offering from a user experience perspective. Product features of this innovation type do not change the core product, but optimize existing features.
Marketing innovation positions the more or less unchanged product for new markets.
Experiential innovation modifies the customer's end-to-end experience from initial encounter to ultimate disposition.

Marketing and experiential innovation do not require changes to the core product. These innovation types give innovation options to the product manager that are independent of development resources.

The interplay between product life cycle and market category life cycle depends largely on the strategic planning within an organization. Based on portfolio considerations there may be situations where strategic planning decides to move a product to its end-of-life phase, even if the product is still in its growth phase.

References

C. Anderson, C. Zeithaml (1984); V. Rajlich, K. Bennett (2000); D. Kolb (2014); G. A. Moore (2014); G. A. Moore (2005)

EU3 Product Roadmapping

Duration: 3:00h

Educational Objectives:

EO3.1 Understand the role of roadmaps for operationalizing product strategy
EO3.2 Understand elements, structures, and forms of software product roadmaps
EO3.3 Understand how to develop and obtain support for a roadmap
EO3.4 Be able to perform a roadmapping workshop

A roadmap is a plan to guide the work of a team towards the fulfillment of a goal. The term has its origins in automotive travel and has found increased acceptance in areas such as science, technology, industry, and business planning. In the software industry, roadmaps are used to plan the upcoming software releases within the strategic timeframe, which can be between one and five years (*Lehtola (2009)*). When used by a company, the roadmap describes the company's intentions and assumptions of how the future will evolve (*Kostoff (2001)*). It is usually presented graphically to illustrate dependencies between its parts (*Phaal (2004)*).

In software product management, a roadmap is a plan for realizing a product vision according to the product strategy. As such, a roadmap is a tool for discussing and communicating the product vision and strategy. The roadmap is at a level of detail that the collaborating corporate functions understand how they contribute towards the implementation of the product vision and strategy. Short-term outcomes are planned in much more specific terms and with higher reliability than long-term outcomes. Product roadmapping enables top-down planning and coordination in addition to bottom-up iteration and learning.

In addition to planning, the most important uses of a roadmap within a product organization are:

- Alignment of company functions with the product vision and the strategy.
- Scheduling of multiple projects related to a portfolio of products, a series of product versions, or alternative product variants.
- Management of dependencies on technologies that are to be developed or procured, so that they can be used for development of the product.
- Transfer of product knowledge for marketing, sales, and other company functions that need to adapt their activities as product versions are released.

The roadmap operationalizes the product strategy by identifying the capabilities the product organization will deliver. A roadmap becomes an important tool when the product has been aligned with the needs of first customers and extensions and changes are planned for evolving that product. New product versions will be developed by product development. Components, platforms, services, and other technologies will be delivered by research or development and used as an input to product development. Other activities will be executed by marketing, sales, or other company functions and will use the results of product development.

A product roadmap defines the scope of planned releases and sets expectations for release dates. Drivers for software releases include:

- Market opportunities: customer needs that the product company can convert into substantial revenue and can defend against competitors. External stakeholders and partners will need to be able to consume the product changes.
- Threats: competitor moves, technology changes, and new alternatives for satisfying customer needs that threaten a product company's market share or revenue.
- Improvements: user experience and other quality improvements, reliability and performance improvements, security enhancements, and bug fixes. Bug fixes may have been explicitly requested by customers or may have been discovered by the product company's staff.
- Cost reduction: changes to platforms, components, and services that enable future development projects.
- Revenue generation: for example, in the case of a one-time charge pricing model, customers pay upon product upgrades.
- Change of regulations: changes of regulations must be implemented for becoming or remaining authorized to offer the concerned products and services.

The project schedules will be aligned with events expected to happen in the product organization's environment. Such events may be fairs or conferences where the product is presented or launches of competitors' products. Also, the resources available to the product organization, e.g. the developers that can be assigned for product development, will constrain the development velocity and the features that can be implemented.

A variety of graphical representations for product roadmaps has been developed. The variants account for the many ways product roadmaps are used. Examples include (*Phaal (2004)*):

- Bar roadmaps are used to specify expected technology evolution. Each row indicates states how technologies will evolve for a given technology domain.
- Trees are used to specify alternative options for product development. The branches and leafs state possible product variants or features.
- S-curves are used to specify series of product successions. For a given technology or platform, each s-curve states the relative timing and performance of product releases.
- Layers are used to specify series of interdependent projects. For a given organizational unit, each row indicates the scope and timing of projects. Dependencies among the projects indicate the use of projects results.

A layered product roadmap that is used for product planning consists of two axes (*Phaal (2004)*). The horizontal axis indicates the time and covers the strategic timeframe. For software products with frequent releases, this time frame is up to one year (*Lehtola (2009)*). Products with hardware elements often use longer time frames (*Groenveld (1997)*). The vertical axis refers to the organizational units that collaborate for achieving the product vision. Additional layers that may be used for product planning are scenarios for product use, development capacity, and availability of expertise (*Fricker (2012*)).

With the Agile and DevOps movements, the cycle times from customer requirements to deployed application feature is shortened. In such an environment, the product manager is managing a portfolio of potential product features that are prototyped, exposed to selected customers, and evaluated to decide about maturation and inclusion of the feature into the product for all customers, respectively removal of the feature candidate. Here, the roadmap consists of feature candidates and planned development and validation of these features.

Roadmaps need the support of the organization to be effective. That support can be gained by co-creation of the roadmap in a workshop with the stakeholders that are responsible for the projects and events stated in the roadmap. Such a workshop includes the following steps (*Phaal (2007)*):

1) Share strategic landscape: each stakeholder presents its perspective, captures these and other relevant information on a roadmap wall chart, and sets priorities.
2) Identify innovation opportunities: together, the stakeholders identify and prioritize opportunities for creating efficient approaches to satisfy existing and future needs and for exploiting capabilities that are developed over time.
3) Explore priorities: the meeting participants explore ways forward to implement vision and strategy of the various concerned functional units of the organization. The timing of actions will be aligned with customer events, fairs, dependencies among product releases, the corporate heartbeat used for synchronizing product and system releases, and the availability of skills and resources.
4) Review and agree on the plan: the meeting participants review the roadmap and prepare the detailed planning and launch of imminent projects.

A roadmapping workshop will lead to an understanding of context, objectives, projects, and dependencies that is shared among the concerned company functions. For the product manager, this shared understanding is critical to know that the organization supports the product development and evolution plans. When evolving a product, the roadmap is a useful tool for coordinating the projects that are launched. To account for the many events and surprises that emerge as the roadmap is implemented, the roadmapping process is regularly repeated.

In addition to the internal use for planning and coordination, roadmaps are used for communicating to stakeholders outside the product organization and company. Such sharing of a roadmap may be integrated into marketing communications. The sharing allows external stakeholders to understand the product vision and strategy and the approach initiated to implement the vision and strategy. The stakeholders' reactions are an input for validating the contents and messaging of the roadmap. Hence, an external roadmap is a tool to elicit feedback for the planned product development and evolution.

External roadmaps also play an important role in demonstrating the viability of a product. They are used to build trust in the commitment of the company to continuous long-term investment in the product. Influential customers or partners may want to see a product roadmap, against signing non-disclosure agreements, before they make a significant investment decision or decide about continued cooperation. Similarly, press and market analysts base their judgment of where the company is heading on a convincing story about a product's future, which is expressed in the roadmap.

References

S. Fricker, S. Schumacher (2012); P. Groenveld (1997); R. Kostoff, R. Schaller (2001); L. Loehtola, M. Kauppinen, J. Vähäniitty, M. Komssi (2009); R. Phaal, C. Farrukh, D. Probert (2004); R. Phaal, C. Farrukh, D. Probert (2007)

EU4 Product Release Planning

Duration: 3:00h

Educational Objectives:

EO4.1 Know the generic release planning process
EO4.2 Understand criteria for requirements prioritization
EO4.3 Be able to apply methods for requirements prioritization
EO4.4 Be able to apply visualization of prioritization results analysis
EO4.5 Understand common variations of the release planning process

The purpose of release planning is to define the content and schedule of one forthcoming product release so that the product value is maximized over its life cycle. The software product manager needs to make sure that the release plan is in sync with the product roadmap and matches the organization's capability for delivery. Release planning may be carried out for several versions of the product in parallel.

Usually, the development resources are not sufficient to implement all the identified requirements for the product. Therefore, product requirements need to be prioritized to determine which ones will be delivered in the forthcoming release. Requirements prioritization is the focal point of release planning decisions as it allows focusing on the most important functional or quality requirements and their timely delivery to the potential customers.

For market-driven companies, prioritizing requirements is challenging because of the number of candidate requirements suggested and a lack of direct customers to validate the importance of those requirements. Thus, balancing market pull and technology push as well as resolving conflicts between Marketing and Development becomes an integral part of requirements prioritization negotiations (*B. Regnell, S. Brinkkemper (2005)*).

The release planning process differs in scope and intensity substantially depending on the product scenario a company is in. In vendor-controlled scenarios more explicit release planning is used only for larger requirements. Product functionality is delivered often and in small increments. In these scenarios, an agile methodology is frequently applied. A small number of items from the backlog are selected by the team, implemented, and delivered to the customer. Each increment delivers some value to the customer that can be evaluated. The feedback after each increment helps to maximize the customer value and satisfaction. Vendor-controlled environments enable product managers to experiment with certain functionality. Instead of providing new functionality to all customers, it will be provided only to a small group of customers. Based on the usage feedback from this group it can be decided whether the functionality will be provided to the full customer base.

Care must be taken on requirements that span teams or delivery cycles. Techniques how to address these issues are described in the methods for large-scale agile development (*D. Leffingwell (2011), C. Larman, Bodde (2010)*).

The following text applies to customer controlled scenarios.

A generic release planning process includes the following steps (*G. Ruhe, M. Saliu (2005), Amandeep, G. Ruhe, M. Stanford (2004), G. Ruhe (2010), Svahnberg M, Gorschek T, Feldt R, Torkar R, Saleem S, and MU S (2010)*):

1. Understand the product and organizational environment
2. Understand the parameters of the development
3. Understand product life cycle dependencies that influence release decisions
4. Prepare release planning
5. Execute release planning by generating, evaluating, and negotiating possible release plans, and making decisions
6. Analyze and reflect on the release planning experience
7. Package the experience and the results to make the experience available to others

Regarding steps 1, 2 and 3, a solid understanding of the product and organizational environments helps the product manager in the execution of the release planning process. Knowing the parameters of development regarding stakeholders, requirements, preferences, constraints, and, in particular, cost constraints streamlines the release planning process.

Before the actual release planning process begins, product managers must make provisions: Appropriate information systems have to be selected, and stakeholder workshops need to be planned.

The execution of the release planning process is typically iterative and involves discussions and negotiations among various stakeholders, including in most cases SPMs, requirements analysts, engineering and management.

Depending on the product's life cycle phase, the release planning process varies. In the early product life cycle phases, release planning is characterized by a high degree of uncertainty. The product is not yet completely shaped, and product use cases are evolving. Release planning focuses, therefore, on the most valuable requirements that are needed next by customers to enable product success. A prominent technique to pursue this goal is minimum viable product (MVP). According to Eric Ries, an MVP "*is that version of a new product which allows a team to collect the maximum amount of validated learning about customers with the least effort.*" (*Ries (2009)*) MVP allows stakeholders to focus on a reduced market segment initially and expand later on.

Once products reach the maturity phase, release planning no longer focuses on new functional requirements. Typical requirement categories considered then are requirements dealing with changing environments, UX optimization, performance and reliability requirements. As the product architecture and constraints are well-known, requirements can be estimated more precisely than in earlier phases, leading to improved release planning outcomes.

For products approaching the end of their life cycle, release planning focuses on requirements to retain customers with minimal investment. Large and disruptive changes should be directed towards a replacement product so that user are not disturbed in their habits (*K. Hoffman, K. Wnuk and D. Callele (2014)*).

Depending on the product and organizational environments, specific criteria are used for driving release planning decisions. Business criteria are based on the product strategy and include marketing concerns, e.g. release themes and dates, competitive situation, stakeholder priorities, and requirements volatility. Management criteria include development cost/value, risk, resources and competencies, delivery date and calendar time in relation to the roadmap, and support for education and training. System criteria include system impact, complexity, requirements dependencies, evolution, and maintenance. Requirements dependencies increase the complexity of release planning and require additional analysis to be performed, e.g. if a dependent requirement is removed from the release what would be the impact on the dependent requirements. A good release plan balances the most important criteria in a way that is accepted by the product stakeholders (*C. Wohlin, A. Aurum (2005), M. Cohn (2006)*).

Several requirements prioritization techniques are available. They range from simple methods requiring almost no prerequisites to more advanced techniques, which require detailed upfront estimations about multiple prioritization criteria. Simple requirements prioritization techniques include selecting the top ten, ranking (e.g. into three groups with high medium or low priority), or distributing 100 dollars among the candidate requirements (also called cumulative voting). Selecting the top ten requirements and ranking are considered easy but have coarse granularity. The 100 dollars test is considered to provide finer granularity. These methods do not require estimating the absolute cost and value of the prioritized features. AHP is rather sophisticated and operates on a ratio scale for estimating cost and value for the prioritized requirements. Requirements can be prioritized on multiple criteria.

Requirements prioritization differs greatly between companies and their development models.

For companies utilizing agile methodologies, requirements prioritization is an iterative process that helps to know the most urgent requirements or features to be developed in the next short development cycle. As a result, Agile development uses frequent and iterative requirements prioritization as a method to adjust to the rapidly changing market situation and customer needs. Before each development iteration, the requirements backlog is re-prioritized, and new items are inserted based on their estimated priorities.

Tool-based techniques allow more fine-grained and robust prioritization than manual techniques. A popular technique called Analytical Hierarchical Process (AHP) is suitable for prioritizing strategic aspects from the product manager perspective. It is based on comparing all possible pairs of hierarchically classified requirements to identify which in the pair has a higher priority. The method works well for up to 30 requirements, but for larger requirement sets the number of comparisons increases dramatically.

Collaborative prioritization methods with active participation of key stakeholders are usually more efficient and trustworthy than analytical methods. A prominent example of the agile techniques is called “Planning game.”

The results of the prioritization can be visualized in a two-dimensional space, see Fig4. The features (dots) placed in the upper left triangle represent the best tradeoff between cost and benefit. Other visualization methods for showing the prioritization results include simple or cumulative bar diagrams of relative priorities according to given criteria, and stakeholder satisfaction or disagreement charts (*B. Regnell (2001)*).

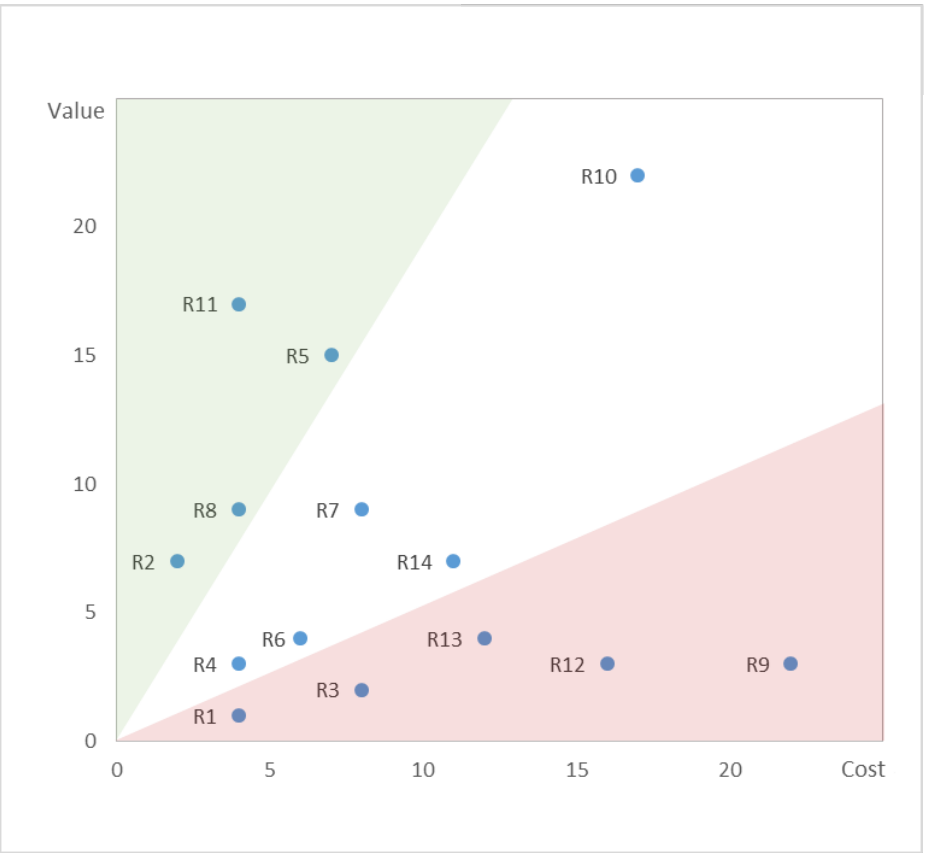

Fig. 4: Cost value prioritization results visualized on a two-dimensional space

Pareto analysis is a technique to match the most important requirements for the given delivery capacity. The technique is based on the 80:20 Pareto principle that 20% of the work will generate 80% of the results. The top 20% of the analyzed requirements are selected (*H. Raiffa, J. Richardson, D. Metcalfe (2007)*). Davis recommends starting development with a requirements baseline that represents 60% of the capacity available in development.

Results from release planning are documented in a release plan that all stakeholders have to agree to. Minimal elements of a release plan include release dates, resource assumptions, and selected requirements. Depending on the type of business, varying degrees of details regarding the requirements specification are required. The range starts at one-liners in requirements backlogs of agile teams to fully elaborated functional specifications in controlled environments.

Tool support for release planning is highly recommended, especially for large releases with hundreds of features and many decision iterations. Currently available tools offer several functionalities beyond simple data storage, e.g. visualization and what-if-analysis.

References

P. Berander, A. Andrews (2005); M. Cohn (2006); A. Davis (2005); D. Greer, G. Ruhe (2003); C. Larman, B. Vodde (2010); D. Leffingwell (2011); H. Raiffa, J. Richardson, D. Metcalfe (2007); B. Regnell, M. Höst, J. Natt och Dag, P. Beremark, T. Hjelm (2001); E. Ries (2009); G. Ruhe (2010); G. Ruhe, M. Saliu (2005); J. Shore (2007); M. Svahnberg, T. Gorschek, R. Feldt, R. Torkar, S. Saleem, and S. MU (2010); L. Thompson (2005); C. Wohlin, A. Aurum (2005)

EU5 Product Requirements Engineering

Duration: 4:00h

Educational Objectives:

EO5.1 Know the main requirements engineering activities
EO5.2 Be able to set up a process for continuous requirement engineering
EO5.3 Understand the role of requirements engineering in software product planning
EO5.4 Understand the role of innovation and the Kano model in requirements engineering
EO5.5 Be able to apply requirements engineering activities dependent on the product scenario

Requirements engineering (RE) in a software product management context covers the classical requirements engineering activities, such as elicitation, analysis, selection, and validation, adapted to the market-driven situation with many often anonymous customers, competitors, and partners. In the Powerboat and Speedboat scenarios, experimentation is added, i.e. new functionality is introduced in the product, customer acceptance and usage are monitored, and based on that data, the functionality is adapted or removed again. This is called customer discovery.

Product requirements do not only address the software itself, but all elements of the whole product offering including product-related services, terms and conditions, etc. They can come from a broad range of sources, i.e. not only customers. Nevertheless, it is more important than ever to get customer or user feedback during the elicitation and analysis phases. As described in ISPMA's Foundation Level syllabus, we distinguish functional and quality requirements and constraints, and also customer, product and project requirements.

To handle large numbers of product requirements, a process and corresponding tool support are needed. The process can be defined based on the "requirements inquiry cycle" or a state model (see ISPMA's Foundation Level syllabus). For experimentation, iterations need to be added to the standard process. For tool support, see EU6 in this syllabus.

Product requirements engineering focuses on aspects which provide additional value for a larger group of existing or potential future customers and product-related stakeholders, whereas requirements engineering on the level of a bespoke development project focuses on an individual customer and the stakeholders of that solution. The focus on value is supported by approaches like the software value map (*M. Khurum, T. Gorschek, M. Wilson (2013)*). On the product level, market-driven requirements engineering that provides customer and market needs understanding is needed. Its key characteristics are the lack of an individual customer-supplier-relationship, a large number of continuously incoming requirements, the success of the product measured by market response rather than contract fulfillment, and the presence of competitors who generate pressure on long-term planning that integrates release planning, roadmapping and product strategies into the RE process.

As stated in the Kano model, a product organization must not solely focus on customer requirements; it also needs to bring innovative ideas into the product. Innovation requires the application of appropriate processes that help to prepare, incubate, illuminate, and verify the

suggested ideas. Openness, dedicated time for idea generation, allowance to discuss or suggest risky ideas, and experimentation are among the key factors for establishing an environment that fosters innovation.

Innovation can apply to products, product features, quality, service, and process. Products have different degrees of innovation: from new-to-the-world products that are novel, useful, and surprising (radical innovations) to simple cost reductions or incremental improvements to existing features (incremental innovation). Highly innovative features can add substantial value to a product and differentiate it in competitive situations. On the other hand, radical innovations often require substantial investments and may not be warmly accepted by the customers.

Requirements management applies tracing to manage requirements across customer, product and project requirements and projects and releases by documenting their change history, source, states, and rationale. Siemens' Poirot system is an example for this (*J. Cleland-Huang, R. Settimi, E. Romanova, B. Berenbach, S. Clark (2007)*). In this way, requirements are analyzed only once, and changes are captured and correlated to the product versions, releases and variants and the relevant development projects. This enables requirements reuse and supports release planning and roadmapping as well as documentation in the form of release notes.

While techniques like workshops and interviews can be helpful for requirements engineering in all product scenarios (see Fig. 2 in EU1), there are some scenario-specific approaches:

Powerboat: Determining the minimum viable product for the first customers requires intense cooperation with these first (potential) customers and users. Hands-on cooperation based on prototypes and iteration results from development is needed. Agile development supports this nicely with experimental requirements engineering being part of the agile team's work. A/B testing is a technique that allows evaluation of alternatives by defining separate sets of users, matching alternatives to user sets, and comparing the usage.

Speedboat: Ongoing analysis of the actual usage of the product provides valuable insight into what works and what does not work, which in turn produces new requirements. Analysis needs to go hand in hand with experimental requirements engineering, i.e. customer discovery can be applied to test new ideas as long as it does not irritate users. A freemium price model can help to increase the number of users faster, thereby enabling faster feedback.

Icebreaker: Determining the minimum viable product requires extensive domain analysis with potential customers. Joint business process analysis may be applied. If regulatory requirements are relevant, early in-depth analysis is needed. The product manager needs to ensure that the resulting product requirements are not too customer-specific, but relevant to the complete initial target market. The degree of involvement of the development team in requirements engineering depends on the chosen development methodology.

Cruise ship: Since the release frequency is low, there is hardly any room for applying the customer discovery approach. Problem reports and customer requirements are valuable inputs. Depending on the type of software, prototyping may be helpful for elicitation and analysis of product requirements, in particular for user experience-related requirements. Release contents are a key to success. So

release planning is important and it needs to be supported by product requirements engineering providing analysis results as a basis for decision making.

References

C. Alvarez (2014); S. Blank (2013); J. Cleland-Huang, R. Settimi, E. Romanova, B. Berenbach, S. Clark (2007); A. Croll, B. Yoskovitz (2013); J. Fagerberg (2005); T. Gorschek, S. Fricker, K. Palm, S. Kunsman (2010); Kano (1984); M. Khurum, T. Gorschek, M. Wilson (2013); H. Kittlaus, P. Clough (2009); N. Maiden, A. Gizikis, S. Robertson (2004); K. Pohl, C. Rupp, (2011); B. Regnell, S. Brinkkemper (2005); K. Ulrich, S. Eppinger (2011); Q. Xu, R. Jiao, X. Yang, M. Helander, H. Khalid, A. Opperud (2009); D. Zowghi, C. Coulin (2005)

EU6 Process Management for Product Planning

Duration: 2:00h

Educational Objectives:

EO6.1 Understand how to improve the planning processes of a software organization

EO6.2 Understand the role of tools to support the planning processes of a software organization

EO6.3 Is able to adapt and use a tool to support the flow and life cycle of product planning decisions or product requirements

Classroom training alone is not sufficient to achieve noticeable improvements in software product management practice. The effective product manager improves with experience gained from his practice and by learning from others, i.e. cooperation and coaching. Observing and reflecting on one's practice, generalizing from the observations, and testing the learned in new situations provides relevant advancement of understanding and abilities (*Kolb (2014)*). Communities of practice allow practitioners to develop relationships with experts, peers and stakeholders, share ideas, set standards, and build tools to solve problems (*Wenger (2000)*). Such learning allows product managers to improve as individuals and as a community.

Various benchmarking frameworks have been developed for setting an agenda that guides the development and improvement of product management practice in an organization. Well-known published frameworks are the Software Product Management Framework by Kittlaus (*Kittlaus (2009)*) and by Ebert (*Ebert (2009)*). Both frameworks cover the full life cycle from the strategy-oriented market and product analysis to product planning, development, marketing, orchestration, and evolution. Similarly, the ISPMA reference model, which is derived from these frameworks, can be used to obtain comparable guidance. The frameworks have been used for guiding in process improvement (*Ebert (2007)*).

A framework that focuses on product planning is the Software Product Management Competence model (*Bekkers (2010)*). The framework recommends 68 practices for portfolio management, roadmapping, release planning, and requirements management areas that are performed in collaboration with external and internal stakeholders. Within each area, the practices have been ordered to allow an organization to increase its maturity incrementally and where improvements are most important. The framework has become an important tool for a substantial number of companies to steer their process development for product planning.

The performance of product planning can be measured in terms of process and outcomes. Common process measurements are the size of the product backlog or the number of pending product requirements, the average and peak durations for requirements to be decided upon, time and money spent on product planning, and confidence in the product planning decisions. Changes in the number of pending product requirements and duration required for requirements decisions can be used to identify bottlenecks or overcapacity in the product organization (*Petersen (2010)*). Common confidence issues concern the linkage of requirements with business strategy, seeing the big picture of the offering, understanding of the planned product's value, and knowledge of customer problems

are present (*Komssi (2015)*). Outcome measurements include those related to product development, such as planning accuracy (*Herrmann (2008*)), and those related to how well requirement cost and value were predicted during the release decisions (*Karlsson (2006*)). A root-cause analysis of the major deviations between prediction and realized results should be used for guiding the future product planning decisions. Examples of causes that lead to deviations are under-estimation of development effort, orders issued by a specific customer, actions from competitors, or the lack of availability of a good solution to a design problem.

The tool landscape for Product Planning has not stabilized yet and is rapidly evolving. Important for the product organization is that the tools support their users, integrate into the organization's toolchain, and yield productivity improvements. The following gives an overview of important tool categories:

- Office tools: documentation of ideas and decisions, visualization of concepts, and communication with stakeholders. Examples are word processors, presentation tools, e-mail, and messengers.
- Modeling tools: informal structuring and analysis of data, information, and knowledge. Examples are mind mapping and roadmapping tools.
- Tracking tools: enactment of workflows. Examples are spreadsheets and requirements or backlog management tools.
- Prototyping tools: design of wireframes and other approximations of a product. Examples are GUI design tools.

The effect of a tool on productivity differs significantly, depending on the complexity of the products and the processes (*T. Bruckhaus (1996)*). A systematic tooling process investigates the problem to be addressed, identifies available tools to address the problem, evaluates tool candidates with adequate selection criteria, checks the feasibility of implementing the target processes by prototyping the tool with representative data, and checks acceptance of the tool in daily practices with a pilot. Only then a tool should be rolled-out organization-wide (*Gorschek (2006)*).

References

W. Bekkers, I. van de Weerd (2010); T. Bruckhaus, N. Madhavji, I. Janssen, J. Henshaw (1996); C. Ebert (2007); C. Ebert (2009); T. Gorschek, C. Wohlin, P. Garre, P., S. Larsson (2006); A. Herrmann, M. Daneva (2008); L. Karlsson, B. Regnell, T. Thelin (2006); H. Kittalus (2009); D. Kolb (2014); M. Komssi, M. Kauppinen, H. Töhönen, L. Lehtola, A. Davis (2015); K. Petersen, C. Wohlin (2010); E. Wenger, W. Snyder (2000)

Bibliography

This literature has been used by ISPMA as the scientific basis for this syllabus. It is not required reading for course participants.

C. Alvarez (2014): Lean Customer Development – Build Products Your Customers Will Buy. O'Reilly.

C. Anderson, C. Zeithaml (1984); V. Rajlich, K. Bennett (2000)

W. Bekkers, I. van de Weerd (2010): SPM Maturity Matrix, Technical Report UU-CS-2010-013. University of Utrecht.

P. Berander, A. Andrews (2005): Requirements Prioritization. In A. Aurum, C. Wohlin (eds.): Engineering and Managing Software Requirements. Springer.

S. Blank (2013): The Four Steps to the Epiphany. 2nd ed. K+S Ranch.

T. Bruckhaus, N. Madhavji, I. Janssen, J. Henshaw (1996): "The Impact of Tools on Software Productivity", IEEE Software 13(5): 29-38.

J. Cleland-Huang, R. Settimi, E. Romanova, B. Berenbach, S. Clark (2007): Best Practices for Automated Traceability, IEEE Computer, June 2007: 27-35.

M. Cohn (2006): Agile Estimating and Planning. Prentice Hall PTR.

A. Croll, B. Yoskovitz (2013): Lean Analytics – Use Data To Build a Better Startup Faster. O'Reilly.

A. Davis (2005): Just enough Requirements Management, Dorset House Publishing.

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

C. Ebert (2009): "Software Product Management", CrossTalk 16.

J. Fagerberg (2005): Innovation – A Guide to the Literature. In J. Fagerberg, D. Mowery, R. Nelson (eds.): The Oxford Handbook of Innovation. Oxford University Press.

S. Fricker, S. Schumacher (2012): "Release Planning with Feature Trees: Industrial Case", 18th International Working Conference on Requirements Engineering (RefsQ 2012), Essen, Germany.

T. Gorschek, S. Fricker, K. Palm, S. Kunsman (2010): A Lightweight Innovation Process for Software-Intensive Product Development, IEEE Software 27(1): 37-45.

T. Gorschek, C. Wohlin, P. Garre, P., S. Larsson (2006): "A model for technology transfer in practice", Software, IEEE, 23(6): 88-95.

M. Grath (2000): Product Strategy for High-Technology Companies – Accelerating Your Business to Web Speed. McGraw-Hill.

D. Greer, G. Ruhe (2003) Software release planning: an evolutionary and iterative approach, Information and Software Technology, 46 (2004), 243-253.

P. Groenveld (1997): "Roadmapping integrates business and technology", Research Technology Management 40(5): 49-58.

A. Herrmann, M. Daneva (2008): "Requirements prioritization based on benefit and cost prediction: an agenda for future research", 16th IEEE International Conference on Requirements Engineering.

Kano (1984): "Attractive Quality and Must-Be Quality", Journal of Japanese Society for Quality Control 4: 39-48.

L. Karlsson, B. Regnell, T. Thelin (2006): "Case studies in process improvement through retrospective analysis of release planning decisions", International Journal of Software Engineering and Knowledge Engineering, 16(06): 885-915.

M. Khurum, T. Gorschek, M. Wilson (2013): "The software value map—an exhaustive collection of value aspects for the development of software intensive products", Journal of Software: Evolution and Process, Volume 25, Issue 7: 711-741.

H.-B. Kittlaus (2015): One Size Does Not Fit All: Software Product Management For Speedboats vs. Cruiseships, in: Fernandes, J.M., Machado, R.J., Wnuk, K. (Eds.): Software Business, Proceedings of ICSOB 2015, Braga, Portugal. Springer, pp. XII-XIV.

H.-B. Kittlaus, P. Clough (2009): Software Product Management and Pricing – Key Success Factors for Software Organizations. Springer.

D. Kolb (2014): Experiential learning: Experience as the source of learning and development. Pearson Education.

M. Komssi, M. Kauppinen, H. Töhönen, L. Lehtola, A. Davis (2015): "Roadmapping Problems in Practice: value creation from the perspective of the customers". Requirements Engineering 20(1): 45-69.

R. Kostoff, R. Schaller (2001): "Science and Technology Roadmaps", IEEE Transactions on Engineering Management 48(2): 132-143.

C. Larman, B. Vodde (2010), Practices for Scaling Lean & Agile Development, Addison Wesley.

D. Leffingwell (2011), Agile Software Requirements, Pearson Education, Inc.

L. Loehtola, M. Kauppinen, J. Vähäniitty, M. Komssi (2009): "Linking business and requirements engineering: is solution planning a missing activity in software product companies?", Requirements Engineering 14: 113-128.

N. Maiden, A. Gizikis, S. Robertson (2004): Provoking Creativity: Imagine What Your Requirements Could Be Like, IEEE Software 21(5): 68-75.

G. A. Moore (2014): Crossing the Chasm, 3rd ed., Harper Business.

G. A. Moore (2005): Dealing with Darwin: How Great Companies Innovate at Every Phase of Their Evolution, Portfolio.

K. Petersen, C. Wohlin (2010): "Measuring the Flow in Lean Software Development." Software – Practice and Experience 41(9): 975-996.

R. Phaal, C. Farrukh, D. Probert (2007): "Strategic Road-mapping: A Workshop-based Approach for Identifying and Exploring Strategic Issues and Opportunities", Engineering Management Journal 19(1): 3-12.

R. Phaal, C. Farrukh, D. Probert (2004): "Technology Roadmapping – A Planning Framework for Evolution and Revolution", Technological Forecasting and Social Change 71: 5-26.

K. Pohl, C. Rupp, (2011): Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam – Foundation Level - IREB compliant (Rocky Nook Computing).

H. Raiffa, J. Richardson, D. Metcalfe (2007): Negotiation Analysis – The Science and Art of Collaborative Decision Making. The Belknap Press of Harvard University Press.

B. Regnell, S. Brinkkemper (2005): Market-Driven Requirements Engineering for Software Products, in C. Wohlin, A. Aurum: Engineering and Managing Software Requirements, Springer.

B. Regnell, M. Höst, J. Natt och Dag, P. Beremark, T. Hjelm (2001): An Industrial Case Study on Distributed Prioritization in Market-Driven Requirements Engineering for Packaged Software, Requirements Engineering 6: 51-62.

G. Ruhe (2010): Product Release Planning: Methods, Tools and Applications, CRC Press.

G. Ruhe, M. Saliu (2005): "The Art and Science of Software Release Planning", IEEE Software 47-53.

J. Shore (2007), S. Warden, The Art of Agile Development, O'Reilly Media.

M. Svahnberg, T. Gorschek, R. Feldt, R. Torkar, S. Saleem, and S. MU (2010): A systematic review on strategic release planning models. Information and Software Technology Information and Software Technology 52, 237-48.

L. Thompson (2005): The Mind and Heart of the Negotiator. Pearson Prentice Hall.

K. Ulrich, S. Eppinger (2011): Product Design and Development. McGrawHill.

E. Wenger, W. Snyder (2000): "Communities of practice: The organizational frontier", Harvard business review, 78(1): 139-146.

C. Wohlin, A. Aurum (2005): "What is Important when Deciding to Include a Software Requirement in a Project or Release?", International Symposium on Empirical Software Engineering.

I. van de Weerd e.a. (2006): Towards a Reference Framework for Software Product Management, in Proceedings of 14th IEEE International Requirements Engineering Conference (RE'06), IEEE.

Q. Xu, R. Jiao, X. Yang, M. Helander, H. Khalid, A. Opperud (2009): An Analytical Kano Model for Customer Need Analysis, Design Studies 30: 87-110.

D. Zowghi, C. Coulin (2005): Requirements Elicitation: A Survey of Techniques,Approaches, and Tools, in: C. Wohlin A. Aurum Engineering and Managing Software Requirements, Springer.

Internet Links:

IL1: http://www.startuplessonslearned.com/2009/08/minimum-viable-product-guide.html (E. Ries (2009), Minimum Viable Product: a guide)