# ISPMA

**INTERNATIONAL
SOFTWARE PRODUCT MANAGEMENT
ASSOCIATION**

# Software Product Management Syllabus

# The Foundation

## Student Edition

V.2.1

This syllabus was written by the following members of the International Software Product Management Association (ISPMA®): Gerald Heller (editor), Jonas Als, Magnus Billgren, Erik Bjernulf, Sjaak Brinkkemper, Christof Ebert, Samuel Fricker, Tony Gorschek, Rainer Grau, Marc Hilber, Barbara Hoisl, Slinger Jansen, Mahvish Khurum, Hans-Bernd Kittlaus, Daniel Lucas-Hirtz, Andrey Maglyas, Sacha Reis, Niklas Rosvall, Kevin Vlaanderen and Inge van de Weerd.

We thank all honorary authors and contributors.

# Preface

The goal of the International Software Product Management Association (ISPMA®) syllabus for the ISPMA® Certified Software Product Manager is to promote general 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.

The syllabus "SPM - The Foundation" covers the full spectrum of elements for software product management 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, including those that the product manager interfaces with, e.g. general management, marketing and sales, research and development, production, service and support, and controlling.

The syllabus is the basis for an examination to certify that the examinee has achieved the degree of knowledge described in this syllabus.

**Purpose and structure of the syllabus:**

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

- Explicit 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 five 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 and subchapter numbering. Educational objectives that apply to the entire EU use a „0" as second digit.

The educational objectives are expressed in terms of two cognitive levels of knowledge, expressed with the verbs "know" for level 1 and "understand" for level 2. These two verbs are placeholders for the following:

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

Each EO in the syllabus has one or both of the two cognitive levels assigned to it.

**Included and excluded key areas:**

This 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.

This 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 appropriate exercises that give participants an opportunity to apply the training contents to practical cases. 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 certificate "ISPMA® Certified Software Product Manager – The Foundation". 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 at least three years of experience in the software industry in general. However, this is a generic recommendation and might not be applicable for all situations or course participants.

**Terminology**

The term SPM is used as an abbreviation for Software Product Management. It represents the function of SPM and not the individual person.

This curriculum usually uses a gender-neutral form. In cases where the masculine form is used, this is done for readability reasons and represents any other gender as well.

The terms used in this syllabus are consistent with the glossary of the ISPMA® available at ispma.org.

## Table of Contents

## EU1  Introduction and Foundations

Duration:   2:00 h

Educational Objectives:

EO1.1.1     Know the scope and effects of software product management

EO1.1.2     Know the role and objectives of software product management and relevant terms

EO1.2.1     Know the SPM Framework

EO1.2.2     Understand the multidisciplinary nature of SPM

## 1.1  Software Product Management Essentials

Duration:   1:30 h

Product management has become an established discipline in many industries since Procter & Gamble introduced it in 1931 as brand management.  During the last decades, many software product companies – such as Microsoft, IBM, and SAP – implemented Software Product Management (SPM) in their organizations, as did corporate information technology organizations in other industries, and companies that produce software embedded in software-intensive products and services. The role of software product manager has emerged during this time as being of strategic value, since it is crucial for the economic success of a product. Successful product management means delivering the right products at the right time and price for the right markets, and increasing profitability, product quality, customer satisfaction, and the success rate of releases in terms of schedule predictability and time-to-market.

Literature: *Ebert C. (2007); Fricker S. (2012); Kittlaus, H.-B. (2022, p. 7 ff)*

A product is a combination of goods and services, which a supplier/development organization combines in support of its interests and where defined rights are transferred to a customer. A software product is one whose primary component is software. It compares to non-software products in terms of high complexity, negligible manufacturing cost, great flexibility, and a high rate of change. In most cases, the interests of an organization are commercial, but there are exceptions, as is sometimes seen in open source-based products or in the public sector. Software products are typically produced for multiple (customer) organizations with many users.

In this syllabus, software product management means the management of software products and software parts of software-intensive products, i.e. systems or services. Software parts of software-intensive systems that are not marketed and priced as separate entities are called embedded software. Software-intensive systems can be products from all industries like cars, airplanes, smartphones etc. Software-intensive services, often delivered as cloud services, can also be products from all industries like financial, insurance, gaming, social software, or human services based on software support. To support ease of reading this syllabus will generally use the term "software products" instead of explicitly referring to software-intensive products as well.

Literature: *Kittlaus, H.-B. (2022 p. 10 ff)*

A product family is a set of products with common features, integration, interface similarities, and technological similarity to address a specific problem. It is defined for marketing reasons. A product line is based on a platform with defined (static or dynamic) variability for developing products tailored to different markets and users, while increasing quality and decreasing cost. Its definition is based on both technical and business reasons.

Literature: *Clements, P.; & Northrop, L. (2015); Kittlaus, H.-B. (2022, p. 14 f)*

A platform in an ecosystem is a technical product that the platform owner as well as third parties use as a foundation for conducting their own respective businesses. This may include technological collaboration and integration, as well as commercial interaction and financial transactions. There are two non-disjoint types of platforms:

● Innovation platform: Technological foundation upon which the owner and other firms develop complementary innovations.

● Transaction platform: Intermediary or online marketplace that makes it possible for people and organizations to share information or to buy , sell, or access a variety of goods and services.

A software product manager is responsible for managing software with the objective to achieve sustainable success over the life cycle of a software product (family or line). This generally refers to economic success, which is ultimately reflected by the profits generated. Software product managers have the business responsibility across different versions, variants and associated services of a product. They have to manage a broad set of product-related activities as shown in the ISPMA® SPM Framework (Figure 1). They have to act proactively and be the responsible and engaged driver of their products. Due to this broad responsibility that also requires involvement of others, the role of the software product manager is sometimes referred to as mini CEO of their product. Typically they have direct responsibility for the activities marked as "Core SPM", in particular "Product Strategy" and "Product Planning". For the activities under "Strategic Management", software product managers participate by representing their products on the corporate level, e.g. in portfolio management, by providing input and making use of the results. For the activities under "Development", "Marketing", "Sales and Distribution" and "Service and Support", the direct responsibility is typically with other units in the company, but software product managers have to orchestrate these activities such that they are performed in line with product strategy and plan. Given the broad set of responsibilities, prioritization is needed on an ongoing basis and can be based on the respective estimated impact on short and long term objectives.

Literature: *Cusumano, M. (2004); Cusumano, M., et al. (2019); Ebert, C. (2007); Fricker, S. (2012); Kittlaus, H.-B. (2022, p. 42 ff); Parker, G.G., et al. (2016); Kittlaus, H.-B. (2020b); Ries, E. (2011)*

## 1.2 Software Product Management Framework

Duration:   0:30 h

The SPM framework provides a holistic view on the activities of software product management. It can be used as a model to establish and improve the discipline of software product management in an organization. It is structured in the following way:

- The horizontal structure (columns) is based on the functional areas of a software organization.

- Vertically, i.e. within the columns, the structure is based on a top-down approach, i.e. from strategic and long-term to operational and short-term. However, the interdependencies of the elements within each column (and also across columns) are more complex than can be fully expressed in a two-dimensional structure. There are a number of cases where the actual doing requires iterative processes that go back and forth between elements until everything is synchronized. A good example is the Product Strategy column where this kind of iterative approach is mandatory between most elements before a product manager gets to a consistent strategy. Also, there are elements like "Ecosystem Management" and "Customer Relationship Management" (CRM) that contain both longer-term and shorter-term aspects.

- There is an additional overlay structure with "Core SPM", "Participation" and "Orchestration". For Market Analysis and Product Analysis in the Strategic Management column, corporate functions are typically responsible in larger companies with the product manager acting as a participant. Whereas, in smaller companies, the product manager may be responsible. In any case, obtaining reliable information about the market and product on a frequent basis is part of the core SPM responsibilities. Activities under Orchestration are under the responsibility of the respective functions. However, the activity of Orchestration itself is a core responsibility of SPM.

| Strategic Management | Product Strategy | Product Planning | Development | Marketing | Sales and Fulfillment | Delivery Services and Support |
|---|---|---|---|---|---|---|
| Corporate Strategy | Positioning and Product Definition | Customer Insight | Product Architecture Management | Marketing Planning | Sales Planning | Service Planning and Preparation |
| Portfolio Management | Delivery Model and Service Strategy | Product Life Cycle Management | Development Environment Management | Value Communication | Customer Relationship Management | Service Execution |
| Innovation Management | Ecosystem Management | Roadmapping | Development Execution | Product Launches | Operational Sales | Technical Support |
| Resource Management | Sourcing | Release Planning | User Experience Design | Opportunity Management | Operational Fulfillment | Operations |
| Compliance Management | Pricing | Product Requirements Eng | Detailed Requirements Engineering | Channel Preparation | | |
| Market Analysis | Financial Management | | Quality Management | Operational Marketing | | |
| Product Analysis | Legal and IPR Management | | | | | |
| | Performance and Risk Management | | | | | |

| Participation | Core | Orchestration |
|---|---|---|

Activity under SPM responsibility    Activity under other function's responsibility

ISPMA reference architecture v 2

Fig. 1 SPM Framework V. 2.0

This syllabus is structured in correspondence with the SPM Framework with only a few deviations due to pedagogical reasons. It starts with the core SPM columns Product Strategy and Product Planning, then Strategic Management and the Orchestration of the Functional Areas are described.

Literature: *Ebert, C. (2007); Ebert, C., & Brinkkemper, S. (2014 pp. 10-18); Fricker, S. (2012); Kittlaus, H.-B. (2022, p. 37 ff); van de Weerd I., et al. (2006)*

## EU2   Product Strategy

Duration:  6:15 h

Educational Objectives:

EO2.1.1    Understand the role of a product strategy, its elements and their interdependencies

EO2.1.2    Know the relationship of product strategy to corporate strategy and portfolio management

EO2.2.1    Understand the elements of the product definition

EO2.2.2    Understand product positioning and its elements

EO2.3.1    Understand the software delivery models and their consequences

EO2.3.2    Understand the importance of integrating the service strategy into the product strategy

EO2.4.1    Know the different types of sourcing

EO2.5.1    Know the Strategic Pricing Pyramid

EO2.5.2    Know the different roles of the software product manager within pricing

EO2.6.1    Know elements relevant for financial evaluation and investment decisions

EO2.6.2    Understand the business model concept

EO2.6.3    Understand business cases

EO2.7.1    Know the role, significance and elements of a software ecosystem

EO2.8.1    Understand the legal differences between a software license contract and a service contract

EO2.8.2    Understand the importance of protecting intellectual property

EO2.8.3    Know intellectual property protection mechanisms

EO2.9.1    Understand evaluation metrics for the financial performance of the product

## 2.1   Product Strategy Essentials

Duration:  0:45 h

Software product managers are responsible for defining the strategy for their product (or platform or family) and for supporting and updating it over time. Normally, a strategy covers a time span of about one to five years, however this varies in relation to domain.

The product strategy describes how the product is supposed to evolve over this strategic time frame. It should address the following items:

- Positioning and Product Definition
- Delivery model and Service Strategy
- Sourcing
- Pricing
- Financial Management
- Ecosystem Management
- Legal and IPR Management
- Performance and Risk Management

The first elements, up to Sourcing, are often described in one cohesive product strategy document in order to emphasize the need for full consistency. The remaining elements, from Pricing to Performance and Risk Management, are also of strategic importance, but are usually only included in a product strategy document on an abstract level. The details, e.g. price list or complete forecasting numbers, are kept separate.

All of these items are highly interdependent. If, for example, business planning results in a budget smaller than originally assumed, it will only be possible to evolve the product to a lesser extent or at a slower pace. If new segments are to be added to the target market within the strategic time frame, the product scope may have to be expanded.

Dependency upon other products can also have considerable consequences, e.g., if certain functionalities or enabling code must be available in several products at the same time. In bigger companies that have one or several product portfolios an individual product strategy needs to be aligned with the corporate strategy and portfolio. It should be observed that interdependencies can exist on different levels of abstraction, ranging from portfolio to product, feature, function, and component but also cover management and business decisions included in the strategic concerns described above.

To engage and align all stakeholders, a strong product vision is helpful. Basically, it is a conceptual image of what the future product will be, i.e. high-level descriptions of a product concept (What is it? How can the product satisfy the needs of potential customers better than competition?) and a business model (How can the product be built, sold and supported in a profitable way?). The other elements of the product strategy provide the details that turn the vision into a manageable and executable path into the future. In bigger companies a product vision needs to be aligned with the corporate vision.

Literature: *Kittlaus, H.-B. (2022, p. 51 ff); McGrath, M. (2000); Osterwalder, A., & Pigneur, Y. (2010); Popp, K.M. (2015); Pichler, R. (2016, Part I)*

## 2.2 Positioning and Product Definition

Duration:   0:45 h

Product positioning includes

- Value Proposition: value definition from a customer perspective for the target market segments
- Focus with regard to the target market and segments, the company product portfolio, and the product life cycle phase (e.g. revitalization)
- Channel options
- Partnerships and alliances

The product definition needs to define:

- Functional scope
- Quality scope
- Intended use and users
- User experience (UX) design scope
- Offering architecture (see section 5.1)
- Business architecture (for application software) (see section 5.1)

The offering architecture defines separately priced components of the product offering, and tailorability options in line with the tailorability strategy (see section 2.3). The business architecture is relevant for application software, and is domain-specific, i.e. covers logical data model, process model, business object model etc. This is the responsibility of a business architect. For any architecture considerations, tight cooperation between product manager and architects is required (see also section 5.1).

When working on the product definition, the product manager has to take the company's compliance guidelines into account (see EU4). The product manager might need to address product-specific compliance issues that are not covered by the company's guidelines, in particular, the areas of sustainability and ethics.

It is good practice to document the complete set of components that determine the offering although that will typically be separate from the product strategy document.

For defining the target market and identifying potential segments, the following elements need to be addressed:

- Scope, size, competitive products and their market shares, including international markets and their opportunities.
  See activity "Market and Product Analysis" *in EU 4 – Strategic Management*

- Segmentation: should a standard market segmentation (e.g. from industry analysts) be used or is a custom segmentation required?

Literature: *Bech, H.P. (2015); Dunford, A. (2019); Kittlaus, H.-B. (2022, p. 71 ff); McGrath, M. (2000); Almquist, E., et.al. (2016, pp. 46-53); Osterwalder, A., & Pigneur, Y. (2014)*

## 2.3 Delivery Model and Service Strategy

Duration: 0:30 h

Based on the product definition, the delivery model needs to address the following items:

- Licensed product vs. service offering (e.g. Software-as-a-Service (SaaS))
- Degree of tailorability including tailorability strategy
- Mode of delivery (online access, online download, combination with services, etc.)

Tailorability means the enablement of the product for customer- or market-specific adaptations by providing properties that can be changed after system development. The categories of tailorability are:

a. Configuration: setting or changing parameters
b. Composition: adding or arranging components
c. Customization: adding or changing program or descriptive code

The tailorability architecture defines these options in more technical detail as part of architecture management in Development (see section 5.1).

The service strategy needs to define the services that are part of the total offering and who is supposed to provide these services. If external partners are to provide services this needs to be aligned with the tailorability strategy and ecosystem management (section 2.7).

Literature: *Kittlaus, H.-B. (2019); Kittlaus, H.-B. (2022, p. 75 ff); McGrath, M. (2000); Kowalkowski, C., & Ulaga, W. (2017, Ch.1,2 & 4)*

## 2.4 Sourcing

Duration: 0:15 h

Sourcing needs to address the following topics

- Sourcing options for human resources

  There are a number of motivations for working with external human resources like:
    - Need for specialized skills not available internally
    - Need for more capacity, e.g. developers, than available internally
    - Cost savings due to lower labour cost in nearshore or offshore locations.
- Make or buy decisions for software components

There are a number of motivations for not developing a piece of software internally, like:
- Focus on time to market
- High quality and low cost of an externally available software product
- Restrictions with regard to internal capacity, technology, and skills

The dependency on an external software provider resulting from a buy decision needs to be managed thoroughly from a business perspective.

Literature: *Borg, M. et al. (2019); Kittlaus, H.-B. (2022, p. 88 ff); McGrath, M. (2000)*

## 2.5  Pricing

Duration:  0:45 h

Pricing needs to address the following items:

- The importance of price with regard to business success and customer value
- Market- and value-based pricing
- Problems of cost-based pricing for software offerings
- Strategic Pricing Pyramid (price strategy, policy, level) (see figure 2)
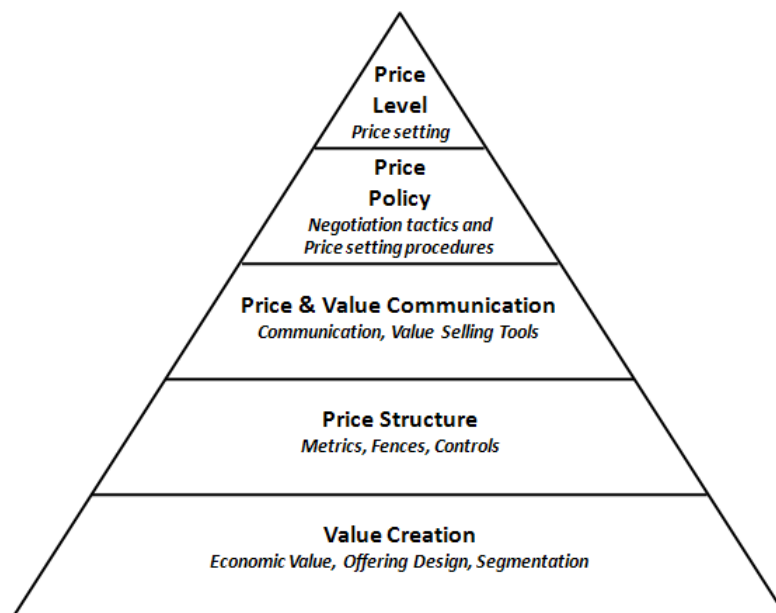- Typical pricing models for software including freemium



Fig. 2 Strategic Pricing Pyramid (© Nagle/Hogan 2005)

A product manager may have different roles with regard to pricing:

- Be the pricing manager, or

- Cooperate tightly with the pricing manager and define requirements regarding price structures and price policies

- Guide Sales regarding price structures, price policies and price levels for the product

- Support marketing regarding price and value communication for the product

Literature: *Kittlaus, H.-B. (2022, p. 90 ff); Kittlaus, H.-B., & Clough, P. (2009, p. 117 ff); Nagle T.T., et al. (2014); Nagle T.T., & Hogan J.E. (2005); Sodhi M.N., & Sodhi, N.N. (2007)*

## 2.6   Financial Management

Duration:   1:15 h

The primary objective of software product management is to achieve sustainable success over the life cycle of the product (or platform or family). This generally refers to economic success, which is ultimately reflected by the profits generated. Since profits lag behind investments, i.e., an investment phase involving losses will be followed by an extended profitable phase, a longer-term perspective is appropriate. Therefore, the product manager has to plan and track financial aspects both from a short- and long-term perspective. This is called financial management and is tightly linked to pricing. It includes the representation of the product on the corporate level.

There are a number of concepts and models relevant for financial evaluation and product investment decisions:
- Investment evaluation models – such as NPV, ROI, Pay Back Time

- The concept of Opportunity Cost and the necessity of identifying the alternative to an investment

- The concept of Cost of Delay and how time affects the business case for a proposed investment

- Different financial management objectives during different Product Life Cycle Stages (PLC)

A business model describes the rationale of how an organization creates, delivers and captures value. It is often considered at the corporate or business unit level, but its consideration can also make sense on a product level, in particular, when a product requires a business model that has previously not been used by the company. For software, the most relevant business models are the software product vendor, either with license products or with SaaS products, and the professional service provider.

The Business Model Canvas is one of the tools that have been used to review and challenge existing business models and systematically invent new ones that change the way a product competes. It defines and visualizes a business model by integrating positioning and product definition, delivery model and service strategy, pricing, sourcing, business case and costing, and ecosystem which are the product strategy cells of the ISPMA® SPM Framework.

Literature: *Osterwalder, A., & Pigneur, Y. (2010); Chen, S., et al. (2013); Kittlaus, H.-B. (2022, p. 24 ff); Haines, S. (2014, Ch.6)*

A business case is a decision-support and planning approach that compares likely financial results and other business consequences with the required investment for a given undertaking. In the case of

software products, the largest investments are typically related to development and marketing. To be effective, the business case should communicate the following information: The description of the undertaking including the underlying assumptions, an estimate for the required investment, the approach to generate business benefits with impact on earnings over time, a sensitivity, risk, and a contingency analysis. A business case is credible when it is complete, balanced with important scenarios elaborated, based on best practice methodology, and when the underlying assumptions are made explicit and accepted by all stakeholders.

Literature: *Schmidt, M. (2002); Kittlaus, H.-B. (2022, p. 109 f)*

The Business Plan builds on:

- Product roadmap (see activity "Roadmapping" in EU 3 – Product Planning)
- Other planned activities that result from the product strategy: For example, decisions to use a new sales channel or to shift focus to a different geography

Understanding the planned activities enables cost management. For cost management, it is important to choose cost estimation methods that fit the organization's processes (e.g. agile development vs. plan-driven development) and that deliver results that are trusted by key stakeholders.

Cost management in mature software product organizations is usually done in a two-step process:

1. The cost target is defined and is referred to as the budget. This can be defined in money, resource allocation (headcount), or a mix.
2. Execution is expected to be done within the budget. This is the responsibility of the line or project manager.

On the level of a company or business unit which is in a mature state, the cost target for a product is defined as the affordable cost which can be estimated based on revenue and profit expectations for a certain period of time. If the company or business unit is in an investment phase, e.g. a startup, the cost target is usually defined by the available funds.

Using these inputs, from product roadmap to cost estimates, the Business Plan provides:

- Forecasting for the selected relevant measures over the strategic timeframe
- Planning of resources and budgets for different functions, in particular for development, support, sales & distribution, marketing

Literature: *Kittlaus, H.-B. (2022, p. 100 ff); Schmidt, M. (2002); Sheen, R., Gallo, A. (2015); Haines, S. (2014); Osterwalder A., & Pigneur Y. (2010)*

## 2.7 Ecosystem Management

Duration:   0:30 h

A software ecosystem is defined as a set of businesses functioning as a unit and interacting with a shared market for software and services while maintaining beneficial relationships. These relationships are frequently underpinned by a common technological platform and operate through the exchange of information, resources, and artifacts.

Organizations can take on different roles in ecosystems, such as:

● Keystone: controlling the ecosystem from the strategic position of platform owner, but leaving space for other actors

● Dominator: controlling the ecosystem from the strategic position of platform owner, striving for complete control

● Niche player: not the platform owner, but benefitting from the ecosystem through specialization and avoiding conflict with keystones or dominators

Product managers are greatly influenced by the role their respective organizations wish to play in a software ecosystem and the extended network of the organization is of increasing importance for the daily work of the product manager. It is not typically part of the responsibilities of a software product manager to decide upon the strategy and role that his/her company aims to play in an ecosystem, as these decisions are usually made by executive management. However, once these decisions are finalized, they have a significant impact on the work of the product manager.

Literature: *Bosch, J. (2009); Iansiti, M., & Levien, R. (2004); S. Jansen, et al. (2013); Kittlaus, H.-B. (2022, p. 111 ff); Messerschmitt, D.G., & Szyperski, C. (2003); Moore, G.A. (2014); Popp, K.M., & Meyer, R. (2010)*

## 2.8 Legal and IPR Management

Duration: 0:45 h

Software product managers need to consider several legal aspects specifically related to software products. Details are typically handled by legal experts (e.g. counsels), but product managers need to have an overview of the legal risks they bear in their role responsible for the sustainable success of the product. First, there are contractual issues between the software vendor and the customer. Then, there is the protection of intellectual property. Third, there are specific risks such as governance, finance, supply-chain, delivery commitments, product liability, data protection (especially for SaaS, in Europe in particular the General Data Protection Regulation (GDPR)), open source license conditions, laws on general terms and conditions, blacklisting of countries for specific software components, etc. which are increasingly relevant (see for instance, the growing impact of governance rules and transparency laws).

The contract by which software is "acquired", may be individually negotiated or, particularly in the mass-market, based on so-called "terms and conditions," which are subject to regulations, in particular if the customer is a consumer. Be it in a contract or in standard terms and conditions, the required legal terms, include:

- Scope of the license or service
- Guarantee and Warranty / SLA
- Transferability
- Type of charges
- Liability
- Maintenance provisions (a separate maintenance agreement may be concluded)
- Miscellaneous legal provisions (e.g. set-off, default, dispute resolution, governing law, severability clause)

The legal terms are accompanied by additional documents which cover product-specific terms. One example is the transaction document, which may contain a definition of the customer enterprise, number of licenses acquired, prices, terms of payment, contract duration (if any) and termination (if any).

The term "license" is a non-technical legal term which is not described by statute, thus its meaning may differ. In particular, it may denote the commercial license model, i.e. the way license fees are calculated, or it may refer to the grant of rights of use under intellectual property laws, in particular under copyright laws. In the latter sense, a license describes to which extent and under which conditions the licensee can use an item, e.g. software, which is subject to intellectual property rights. These rights of use are granted by the licensor which may be the owner of all rights in such an item or may be an entity authorized to grant the license, e.g. a reseller. The term "license contract" means an agreement between the licensor and the licensee about all terms in connection with one or more licenses. With software, the term "license" usually describes the scope of rights of use which the licensor grants to the licensee. Such rights may be exclusive or non-exclusive, perpetual or for a certain time period only, or they may be limited in some way, e.g. geographically or timely. An important part of a license is the rights of use which are granted, e.g. copying (which is required to run the software as well), changing, distributing, publicly conveying, etc. Software licenses may also allow only a particular way of using the software, e.g. only on premise or also for Software-as-a-Service offerings.

In the case of SaaS or other software-intensive services, the service provider needs to make sure that it holds the required rights of use, e.g. by way of license contracts with licensors of the software that is used as part of the service offering which explicitly allow this kind of use. Customers of such a service do not need license contracts, but only service contracts with the service provider, unless a proprietary client is required to be installed on the customer's system.

Literature: *Kittlaus, H.-B. (2022, p. 117 ff)*

Since the development of software requires significant investment and software can be easily copied, it is of utmost importance to the investor that the intellectual property resulting from the investment

is protected. Software is primarily protected by copyright. The copyright protects the source code and object code in its given form. Copyright does not protect the underlying ideas, functions and algorithms and a copyright infringement thus presupposes an unchanged use of the software code (or parts thereof). Product material such as manuals, brochures and product presentations are usually also protected. There are four more fundamental legal constructs which may play a role for the protection of intellectual property:

Trademark: Protection for the names of brands, i.e. trademarks do not apply to the software itself but only to the brand under which it is marketed.

Trade Secret: Protection of company-internal knowledge (primarily against employees). This protection is exercised by restricting knowledge and access to a very small number of people and by using non-disclosure agreements. In most jurisdictions, trade secrets are only protected under unfair competition laws.

Copyright: Protection against copying of software code (as specific expressions of an idea or way of doing something) and product material such as manuals, brochures and presentations. This is the main method for software protection. The algorithm or idea behind software is not protected under copyright law.

Patent: Protection of the specific technical concept or idea. In most jurisdictions, patent protection can generally only be obtained for software which is integrated into a technical solution to solve a problem.

Literature: *Kittlaus, H.-B. (2022, p. 122 ff); Klemens, B. (2006)*

## 2.9   Performance and Risk Management

Duration:   0:45 h

Performance management involves continuous tracking and analysis of selected relevant measures and taking timely action if needed.

Relevant measures can cover different functions, such as development, support etc. Especially important from a product management point of view are business measures. Product managers need to establish clarity over which business measures are really important: Which measures are actually used by important stakeholders to determine the success of the software product? These measures are ultimately used to assess the performance of the product manager. Here are candidates:

- Financial performance:
  - Revenue metrics such as monthly recurring revenue (MRR), average revenue per user (ARPU) and customer lifetime value (CLTV)
  - Profit and loss (P&L) metrics such as gross margin and EBIT
- Market performance:
  - Market share, share of (market) growth, share of (customer) wallet
  - Customer acquisition cost (CAC), share of market spend
- Customer performance:
  - Customer satisfaction score (CSAT) and net promoter score (NPS)
  - Customer retention rate and churn rate
- Organizational performance:
  - Speed metrics such as time-to-market, time-to-revenue, customer time-to-value (TTV)
  - Cost metrics such as cost of quality (COQ) and customer onboarding cost

Risk management requires continuous tracking and analysis of risks identified in connection with development, sales, distribution, delivery, and customer use of the software product, and taking timely action if needed.

Literature: *Kittlaus, H.-B. (2022, p. 129 ff); McGrath, M. (2000); Pritchard, C.L. (2015); Schmidt, M. (2002)*

## EU3   Product Planning

Duration:   5:00 h

Educational Objectives:

EO3.1.1    Understand the three different approaches to product planning

EO3.1.2    Understand the impact of a chosen development methodology on the interface between software product management and development

EO3.2.1    Know typical activities how product managers can gather customer data

EO3.3.1    Understand the specifics of product RE

EO3.3.2    Understand the importance of understanding the customers

EO3.4.1    Know the purpose of release planning

EO3.4.2    Understand the conflicting interests in the release planning process

EO3.5.1    Know the concept, importance and elements of a roadmap

EO3.5.2    Know the major sources for inputs into a roadmap

EO3.5.3    Understand the various ways roadmaps are used and communicated

EO3.6.1    Know the phases of the product life cycle

EO3.6.2    Understand the changes in product management's focus from phase to phase

## 3.1   Product Planning Approaches

Duration:   0:45 h

Product planning is the combination of processes used for converting the product strategy and external insights into an executable plan that is the basis of the product team's work.

For a long time, the standard approach for software product planning and development used to be **requirements-driven**. This approach continues to be broadly used, for example for legal or regulatory requirements, commodity functionality requirements, and technology requirements. It is also applied in methods where hypotheses on customer needs, often referred to as "problems", and potential solutions are evaluated through qualitative and quantitative validation with customers, users and stakeholders. There is a second approach called **data-analysis-driven** where the product team experiments with different implementations of design and product concepts, and makes decisions based on the analysis of performance statistics or usage data. Data can also come from the (potential) user side covering how they behave in certain situations, or what their Jobs To Be Done (JTBD) are, or how much time they spend on what tasks. Such experimentation is a good approach when the focus is on innovation and optimization under uncertainty.  A third approach is called **data-input-driven**. It applies to artificial intelligence/machine learning (ML) where data is used as input for an ML engine.

These three approaches are not mutually exclusive. In many real-world situations, it is recommended that they are used in parallel for different parts of a product and/or different types of requirements in

order to achieve optimal results. The applicability of the different approaches depends on the type of product, the target market, and the product scenario.

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

Fig. 3 Software Product Scenarios

Figure 3 introduces a classification by using two types of runtime environments and two life cycle phases. "Vendor-controlled" means that the software vendor decides which changes are made and when, in the runtime environment. This is typical for rather unregulated environments such as B2C online platforms and SaaS or B2C license products that offer automated online maintenance. If customers want to manage the runtime environment themselves, we use the term "customer-controlled". This is typical for many B2B software license products as well as for software provided by corporate IT organizations.

A requirements-driven approach works in all scenarios. A data-analysis-driven approach works best in vendor-controlled scenarios. A data-input-driven approach is tied to machine learning components in a product and works best for existing products.

In the following sections, Customer Insight, Product Requirements Engineering, Release Planning, Roadmapping, and Product Life Cycle Management are considered primarily under a requirements-driven approach. All three approaches are covered in more detail in the ISPMA® syllabus "SPM - Excellence in Product Planning".

Most of the tasks of product planning require the product manager to cooperate tightly with Development. Development organizations use a variety of methodologies. The chosen methodology on the development side has an impact on the work of the software product manager and the interface between SPM and Development – in particular, how requirements are handed over for implementation and how acceptance of deliverables are managed. In reality, most companies use a mix of different methodologies, be it agile, iterative, or stage-gate, often called "waterfall". Popular agile and lean methodologies include Scrum, Kanban, and XP (eXtreme Programming).

On a high level, agile methods, as well as iterative development, are driven with small, controllable steps or iterations. With most agile methodologies, every iteration may consist of analysis, design, coding and testing. With iterative, development and testing are usually done in iterations while analysis and design are done upfront. With stage-gate, there are no iterations, but one stage or phase is done after the other. So with stage-gate and iterative, requirements are handled early whereas with agile, requirements usually need to be handled in each iteration. There has to be a role in agile

approaches that deals with continuous requirements engineering. In Scrum that role is the Product Owner. In smaller agile development setups, the software product manager may assume the Product Owner role, but that does not scale up. So in larger development environments, the Product Owner role needs to be filled with additional team members who cooperate tightly with the software product manager. With agile methods, compared to traditional methods, typically a larger part of the requirements analysis is performed in the context of development activities.

Literature: *Kittlaus, H.-B. (2020a); Kittlaus, H.-B. (2022, pp. 38 ff. + 140 ff.); Blank, S. (2013, pp. 63-72); Bosch, J. (2009, 2019); Croll, A.,& Yoskovitz, B. (2013); Fotrousi, F., & Fricker, S. (2016); Kittlaus, H.-B. (2012); Leffingwell, D. (2011); Mironov, R. (2008); Maalej, W. et al. (2015, p.33, p.48-56); Doorley, S et al. (2018); Stompff, G., & Smulders, F. (2015); Zacarias, D. (2019)*

## 3.2  Customer Insight

Duration:  0:30 h

Creating and evolving products that meet the ever changing needs of customers requires an excellent understanding of the problems and the environment in which customers operate. Product managers must therefore work towards such an understanding. In this context, the term "customer" is used for all types of customer-side stakeholder, like user, buyer, IT manager, owner, operator, etc.

Two complementary approaches for gathering accurate customer data are available: Direct contact with customers and indirectly through the use of data analytics methods.

Typical direct contact activities are:

- Customer visits
- Meeting customers at conferences, workshops, and events
- Organizing customer round tables (focus groups)
- Design sprints with customer participation
- Supporting selected pre-sales activities
- Participating in support escalations
- Participating in online forums

Data analytics methods are:

- Monitoring online reports of market research agencies, blogs and trade press for customer information
- Monitoring, measuring, and analyzing of user behaviour while they use the product
- Using data analytics software that retrieves information about customer behaviour throughout the internet

Data gathering methods that may include both direct contact and analytics include:

- Customer observation
- A/B testing
- Launching a beta version or minimum viable product (MVP) for evaluation

While data analytics methods are increasingly used by software product managers, experience shows that it is important to spend a significant amount of time in direct contact with customers to gain a deep understanding of the context in which the product is being used and of the customer needs.

Regular participation and engagement help to keep abreast of issues and trends facing existing customers. Direct contact with customers not only supports an analytical understanding of their problems but may also create empathy.

Collecting relevant customer data is only the first step to create insights. Software product managers need to feed this data into discussions with stakeholders and use it for requirements analysis, as well as business modeling.

Literature: *Alvarez, C. (2014); Knapp, J., et al. (2016); Ries, E. (2011, Ch.1-4 ); Kittlaus, H.-B. (2022, p. 137 ff.)*


## 3.3   Product Requirements Engineering


Duration:  1:15 h

Requirements engineering (RE) in a software product management context covers typical requirements engineering activities such as elicitation, analysis, specification, validation, and management, adapted to a market-driven situation with many customers, competitors, and suppliers.

Three requirement types can be distinguished and must be managed separately: Functional requirements, quality requirements, and constraints.

Literature: *Pohl, K., & Rupp, C. (2011); Regnell, B., & Brinkkemper, S. (2005); ISO/IEC 9126-1:2001 (2001); ISO/IEC 25010:2011 (2017); ISO/IEC 40500:2012 (2019)*

Requirements can be raised from different sources:

- Stakeholders
  e.g. customers, users, user groups, business experts, executive management, partners
- Data sources
  e.g. literature, social media, market analysis, product strategy, company guidelines, analytics, standards and regulations
- Systems in operation
  e.g. other existing software products, competitive analysis

The goal of product RE is to continuously identify and manage requirements needed to implement the product strategy and address stakeholder needs. These requirements are called "product requirements" and cover requirements for the whole product (such as requirements for the business model, pricing, or marketing aspects). In contrast to product requirements, the requirements of the stakeholders are called stakeholder requirements.

**Stakeholder requirements** express individual stakeholder's needs. One very important stakeholder is the customer. Their requirements are typically under Sales' or Technical Support's responsibility as part of their Customer Relationship Management mandate.

**Product requirements** address no individual customer, but rather one or many markets which consist of any number of customers that may have similar needs, but with high variability. This market focus influences and impacts all other aspects of product RE, especially elicitation of requirements, innovation candidates, decisions, and management. Product RE is part of the software product manager's responsibilities, and tightly linked to specific SPM activities needed in a market-driven context, such as creating a product vision, strategy definition, roadmapping, innovation, and release planning. Product requirements often result in requirements for development, but there are also product requirements that address other areas, e.g. bundling, new pricing models, new delivery processes, or improved support.

**Detailed requirements** which further refine product requirements that are selected for a particular release and require implementation in the software code. Detailed requirements can also address internal development needs that are raised during development activities. Detailed Requirements are under Development's responsibility.

Literature: *Kittlaus, H.-B. (2022, p. 141 ff); Lucassen, G., et al. (2016, pp. 383-403); Regnell, B., & Brinkkemper, S. (2005)*

A product organization needs a dedicated process to deal with requirements in a systematic way.

Traceability is used to understand the origins and implications of requirements. It allows to follow the path from the original requirement to the software implementation and back. Traceability is sometimes required for legal reasons.

Literature: *Kittlaus, H.-B. (2022, p. 147)*

Needs and expectations arrive in different shapes and forms to product organizations and are named differently, depending on their abstraction level and development methodology. Different styles are used for documentation of requirements before development starts. Well known especially in agile development approaches are epics and user stories. A user story is a natural language description of one or more features of a software system from the perspective of the stakeholder. Epics often represent collections of user stories, sometimes referred to as "themes". In agile development, requirements are documented with an approach to minimize waste and with a focus on the user experience.

Literature: *Gorschek, T., & Wohlin, C. (2006); Pohl, K., & Rupp, C. (2011); Regnell, B., & Brinkkemper, S. (2005); Agile Alliance (2020)*

As requirements evolve throughout their life cycle, they are often specified in standardized documents or in the form of sorted lists called "backlog". For each requirement, standard attributes are used. To

minimize misunderstandings, language templates can be used to specify functional requirements in natural language. To further increase the precision of a specification, semi-formal graphical specification can be employed, e.g. use case diagrams to summarize the services of a system provided to its direct context. Quality requirements can be specified qualitatively, by example, operationally, or quantitatively.

Literature: *Pohl, K., & Rupp, C. (2011); Regnell, B., & Brinkkemper, S. (2005); Weber, L. (2020, p44ff)*


## 3.4   Release Planning


Duration:   0:45 h

In the Icebreaker and Cruise Ship product scenarios, due to marketing and management reasons, a software product evolves and exists in different releases. The long-term sequence and rough contents of releases and versions is planned and documented in the roadmap. Depending upon the type and amount of change, releases are called "Major", "Minor", "Update", or "Service" (patch) release. Major releases are often - for marketing reasons - called "versions". Release planning is concerned with defining the detailed contents of a forthcoming product release in order to maximize the value of the release in relation to the product's success over its life cycle. It needs to be tightly linked to the product requirements engineering process.

The release planning decisions balance opposing forces. On the one hand, the selected product requirements need to satisfy business objectives and real customer needs, while leading to a recognizable advantage over competition. There needs to be a balance between technology push and market pull, i.e. between innovative features and customer requirements. On the other hand, the selected requirements need to account for the capabilities and capacity of the product organization, while being compliant with time and budget constraints and architectural considerations. Other influencing factors can be customer commitments and sales and marketing activities such as fairs. Release planning involves negotiations and setting priorities to resolve conflicts between stakeholders about release contents and interests that are pursued with the evolving product. The release planning decisions ought to be based on strategic guidelines, e.g. to what degree the company is reactive to its markets by giving priority to customer needs or pursues a proactive innovation strategy by pushing new technologies to the markets. Results are documented in a release plan that all stakeholders finally have to agree to. Typically, this is an iterative decision-making process involving the stakeholders in requirements engineering and release planning. The software development approach can also have an impact since agile development requires the software product manager's ongoing involvement in decisions about release contents and synchronization with release planning.

With agile methodology and continuous integration (and even continuous deployment) new functionality can ideally be delivered on a daily basis if customers are able to deal with such a high release frequency. This applies in particular in the Speedboat product scenario (see Figure 3). In such environments release planning does not focus on each individual release, but rather on the requirements that are released within a certain time frame. One SPM challenge is to focus on the

strategic and important items in such a high-pace environment. Another challenge is to communicate the added product value over time.

Literature: *Berander, B., & Andrews, A. (2005); Carlshamre, P., et al. (2001); Cohn, M. (2006); Fricker, S., & Schumacher, S. (2012); Lehtola, L., et al. (2007); Ruhe, G. (2010); Wohlin, C., & Aurum, A. (2005)*

## 3.5   Roadmapping

Duration:   1:00 h

Product roadmapping translates the long-term product strategy into a series of releases that satisfy the business goals of the company and cover the strategic time frame, i.e. between one and five years. A product roadmap usually contains the following basic elements:

- Timescale
- Releases and versions
- Release themes and main features
- Target markets
- Product dependencies
- Technology impacts

The timing of releases is often governed by the "release heartbeat", i.e. a standard frequency the software product manager defines for major and minor releases. Inputs for the roadmap come from release planning, requirements engineering, roadmaps of other products either from the company itself or from partner companies, and from product strategy. There may be separate technology roadmaps that a product roadmap needs to be synchronized with.

Product roadmaps are constructed for internal and external audiences. Internal roadmaps set the scope for specific product releases. They provide the basis for forecasting, budgeting, and the instantiation of projects for the development of specific product releases. They also help with the alignment of product strategies within a company's portfolio. External roadmaps are used for communication with customers, market research analysts, or investors. They are a means to communicate strategy, receive feedback on it, and build trust in the commitment of the company to long-term continuous investment in the product. High level release themes from the roadmap are key elements in guiding the release planning process.

The relationship between roadmapping and release planning is neither top-down nor bottom-up, but the two are combined through an iterative process.

Literature: *Kostoff, R., & Schaller, R. (2001); Phaal, R., et al. (2004, 2007); Pichler, R. (2016, Part II)*

## 3.6   Product Life Cycle Management

Duration:  0:45 h

Product management is responsible for a product along the entire life cycle. Each phase of the life cycle has its individual characteristics and focus areas (Table 1). In the first three phases, investments are necessary to develop the product. During the maturity and decline phases, the product serves as a cash cow, i.e. it generates significant revenue with rather little investment. The resulting profit can be used to finance other products in the portfolio.

| Phase | Focus Areas |
|---|---|
| Conception and creation | Innovation, positioning, investment |
| Market Introduction | Launch, market share, investment |
| Growth | Market share, functionality, investment |
| Maturity | Revitalization, service |
| Decline | Customer retention |
| Withdrawal | Customer retention, cost reduction |

Table 1: Software Product Life Cycle

Product management must have a solid understanding of the various phases in order to develop strategies and activities that optimally support a product in a specific phase. This requires tight cooperation with the involved functional units within the company.

The life cycle responsibility encompasses product-related knowledge management, i.e. the product manager needs to ensure that the knowledge required for the viability of the product continues to be accessible and available to the company during the product life cycle.

Continuous measurement of a product's performance is a prerequisite to drive corrective actions if needed. Therefore, a product manager must monitor and analyze how well the product is performing. This includes product profitability, actual versus planned revenue, customer satisfaction, and market share.

Literature: *Anderson, C., Zeithaml, C. (1984); Rajlich, V. & Bennett, K. (2000); Moore, G.A (2014, Ch. 1-2)*

## EU4   Strategic Management

Duration:   1:30 h

Educational Objectives[1]:

EO4.0.1    Know the activities that belong to Strategic Management
EO4.0.2    Understand the role of a product manager in relation to these activities

## 4.1   Strategic Management Essentials

Strategic Management is an activity within an organization with the content to define, plan, agree, implement and evaluate the organization's strategy. It is part of the responsibility of executive management who can delegate preparatory work to other functions. Strategic Management includes a number of elements related to software product management which are listed below (see also ISPMA® SPM Framework in Fig. 1). Software product managers are typically not responsible for any of these activities, but they either participate in them, e.g. portfolio management, provide inputs, or make use of their outputs, e.g. product analysis.

Literature: *Kim, W.C., & Mauborgne, R. (2015); Mintzberg, H., et al. (2008); Porter, M.E. (1998)*

## 4.2   Corporate Strategy

The strategy on the corporate level typically considers a time frame of up to five years, or longer depending upon the domain. It consists of vision, mission, values and goals, corporate positioning, business model and financial plan, product portfolio and its evolution, resource and competency evolution, technology trends and innovation strategy, market trends and competitive strategy, policies and governance. Software product managers will have to provide input whenever the corporate strategy is revisited and ensure that product strategies stay consistent with the corporate strategy.

Literature: *Kim, W.C., & Mauborgne, R. (2015); Kittlaus, H.-B. (2022, p. 212 ff); Porter, M.E. (1998)*

## 4.3   Portfolio Management

Portfolio Management is an approach to define the investment strategy with regard to the products the company intends to offer to relevant market and customer segments in the strategic time frame. Software product managers are typically asked to participate in the update cycles of the product

---

[1] Educational objectives that apply to the entire EU use a „0" as second digit

portfolio by representing their products and providing input such as roadmaps, forecasts, and investment requirements. The results of such an update cycle can have significant consequences on the product strategy of individual products.

Literature: *Cooper, R.G., et al. (2001); Kittlaus, H.-B. (2022, p. 217 ff)*

## 4.4   Innovation Management

Innovation management is an activity on the corporate level that is intended to create a flow of innovations into the company's product portfolio. This includes cooperation with internal and external research. Innovation management needs to be aligned with the corresponding elements of the corporate strategy on a continuous basis. Alignment can be required in both directions. When innovation management leads to significant results, they need to be incorporated into the corporate strategy and impacted product strategies in order to transform them into competitive advantage. On the other hand, changes of the corporate strategy need to be reflected in innovation management in order to shift resources according to the strategic directions. Software product managers need to monitor and - if feasible - engage with, innovation initiatives to help build sustainable differentiation of their product.

Literature: *Chesbrough, H.W. (2005); Gorschek, T., et al. (2010); Ries, E. (2011, Ch.1-4 )*

## 4.5   Resource Management

On the corporate level, resource management needs to ensure that resources are available in the required quantities and qualities and at the required points in time so that the company is enabled to implement the corporate strategy and the aligned product strategies. This applies in particular to human resources, both in terms of numbers and skills. A software product manager needs to ensure that the resource requirements that result from the product strategy and plan can be fulfilled, i.e. are aligned with corporate resource management.

## 4.6   Compliance Management

Compliance means the act of obeying an order, rule, or request (Oxford Dictionary), in more detail:

**On the legal side:**      implementing any relevant legal or regulatory requirements
**On the non-legal side:**  acting in accordance with any relevant external or internal standards and guidelines, e.g. in the areas of sustainability or ethics

Compliance management means the management of the decision process, including which legal and regulatory requirements are relevant, and which non-legal standards and guidelines the organization wants to comply with. It also includes a governance approach that ensures that the defined compliance

requirements are consistently implemented and audited in the organization. It may include participation in and/or influencing of defining external and internal rules, standards and guidelines.

For sustainability, guidelines such as the United Nations 17 Goals and/or the Sustainability Awareness Framework may be considered.

Company guidelines from compliance management have to be considered in software product definition and may lead to additional product requirements. The alignment of company guidelines and the software product is in the responsibility of the software product manager.

Literature: *United Nations, (2015); Duboc L., et al. (2019); Kittlaus, H.-B. (2022, p. 230 ff)*


## 4.7   Market Analysis


It is of utmost importance for a corporation to have deep insight into trends and developments in the markets it wants to play in, and into the competitive landscape including competitors' strategies. The same holds true on the product level where the product manager needs reliable information. Larger corporations often have specialized market research departments that act as internal service units for the product managers. They conduct their own research and/or collect and evaluate information from market research agencies. In smaller companies, the product manager may be required to do this.

Literature: *Kittlaus, H.-B. (2022, p. 232 ff)*


## 4.8   Product Analysis


A product manager needs, at any time, to be able to know where his/her product stands in order to take action if needed. This is usually done based on measures that are defined as relevant indicators of the business performance of the product  (see section 2.9). In some companies, these numbers are provided by central corporate units, e.g. in Finance; in other companies, this is the product manager's task. In any case, the product manager needs to ensure that reliable numbers are available on a frequent basis. Some companies make a set of well-defined measures mandatory for all products in the portfolio in order to achieve comparability.

Literature: *Kittlaus, H.-B. (2022, p. 239 ff)*

# EU5   Orchestration of the Organization's Units

Duration:   3:30 h

Educational Objectives:

EO5.1.1     Know the core tasks of the functional area Development
EO5.1.2     Understand the product manager's role and responsibilities in relation to Development
EO5.1.3     Understand the different kinds of architecture and their importance to SPM
EO5.2.1     Know the core tasks of the functional area Marketing
EO5.2.2     Understand the product manager's role and responsibilities in relation to Marketing.
EO5.3.1     Know the core tasks of the functional area Sales and Fulfillment
EO5.3.2     Understand the product manager's role and responsibilities in relation to Sales and Fulfillment
EO5.4.1     Know the core tasks of the functional area Delivery Services and Support
EO5.4.2     Understand the product manager's role and responsibilities in relation to Delivery Services and Support
EO5.5.1     Know the product manager's role and responsibilities with regards to orchestration

The division of work in a company that develops software products usually implies that there are separate functional areas for development, marketing, sales, consulting, services and support, each with its own responsibilities, tasks, objectives, and often its own culture. The core tasks of the functional areas are listed in the respective columns in the ISPMA® SPM Framework (Fig. 1). Neither the framework nor the following descriptions give recommendations or make assumptions about the mapping of these functional areas to a company's organizational structure. For each functional area, there is however a section recommending what the product manager needs to understand about the orchestration role and responsibilities related to that functional area.

## 5.1  Development

Duration:   1:00 h

Development is responsible for all technical software aspects, including the implementation of changes and extensions to the software.

Product Architecture Management is done by the leading technical architect of a software product. Product Architecture has a significant impact on a software product with regard to evolution and flexibility. It consists of a number of dimensions that are listed in the table below and needs tight cooperation between product manager and architect. This table suggests a split of responsibilities. The SPM parts belong to the Product Strategy (see chapter 2).

| Architecture dimension | Software Product Manager | Technical architect |
|---|---|---|
| Offering architecture | Lead – define separately priced components of the product (suite, platform) offering, and tailorability options. | Ensure technical feasibility including access management, support for licensing and pricing approach, etc. |
| Business architecture (only for application software) | Lead – define domain-specific architecture, i.e. a logical data model, process model, business object model, etc. | Ensure that technical architecture supports the implementation and change management of the business architecture. |
| Technical architecture | Define the relevant strategy elements such as the delivery model, pricing approach, release approach, quality scope, and IT stack constraints. | Lead – define the technical architecture in line with the business architecture and strategic and technical requirements, e.g. IT stack, programming languages, etc. |
| Tailorability architecture | Define the tailorability strategy as part of the delivery model in line with the ecosystem strategy and sales and marketing strategies. | Lead – Define the tailorability architecture as part of the technical architecture in line with the tailorability strategy. |
| Governance | Ensure that development activities remain consistent with the planned offering, business architecture, and compliance goals. | Lead – ensure that development teams implement in line with the defined technical architecture, and that the technical architecture is only changed based on a well defined and controlled process. |

Table 2: Software Architecture Dimensions

Business Architecture needs to be defined by modeling experts in cooperation with domain experts. We assume that they are part of the SPM organization.

The product architecture can serve as an enabler for competitive edge and market differentiation over time. One method to secure this, is to have a *defining technology* in the software product. These are core technology assets that may be used in multiple components, products and offerings, must be difficult to copy and are the basis for significant customer value in a way that creates sustainable differentiation. Both the product manager and the architect must pay attention to the continuous improvement and protection of the defining technology of their software product.

Development Environment Management addresses development aspects that are relevant across and above development projects. This includes the governance of development processes and tools, IT infrastructure, configuration management, knowledge management, resource and skills management, development sourcing, and estimations.

Development Execution addresses the execution of the actual software development work. How this is done depends on the chosen development methodology, which in particular, has an impact on the way SPM and Development cooperate (see section 3.1). Development may work based on a project structure, or in a continuous mode. Development is usually responsible for writing internal documentation and contributing to the software-related external documentation.

Detailed Requirements Engineering is part of Development responsibility and follows a process similar to the product requirements engineering process. Once the contents of a release are defined, the corresponding product requirements are transferred to Development and further refined. Detailed requirements can also address internal development needs that are raised during the course of development. Synchronization and tracking with product requirements is continuously required. With agile methodologies such as Scrum, a larger part of the requirements analysis is typically done during development, sometimes autonomously by the development teams.

User Experience (UX) design can be a key factor for differentiation and competitive strength. It addresses every aspect of the users' interactions with a software product or component with the purpose of shaping the user's behaviors, attitudes, and emotions about that product or component. Emotions include delight and annoyance about the product, excitement and fear in games, and a feeling of being in control when using decision-support software. UX design must take into account the human-system interaction processes, user interfaces, device and workplace ergonomics, service and content offered by the product, the context of product use and standards such as ISO 9241. UX design is a broader term covering or interacting with disciplines such as graphic design, information architecture, Human-Computer-Interface (HCI) design, interaction design and usability engineering. Industry standards and user expectations for the user experience can vary across types of software and market segments. The UX design scope and expectation for a product is described in the Product Strategy under Product Definition.

DevOps is a development methodology aiming for a tighter cooperation between Development and Operations to achieve better quality of software products, shorter time to market, and improvements in operational efficiency. It is primarily applicable to the Speedboat scenario (see section 3.1). At the core of a DevOps setup is a collaborative culture. To strengthen this culture, product managers need to build a clear and common view for the product vision, strategy, and principles across the functional units involved. They can form the basis for daily decision making and execution. Once a DevOps approach with elements like automated testing and a seamless tool chain is established, product managers not only benefit from improved time-to-market, but also up-to-date product status insights that enable fact-based discussions with the functional units.

Quality Management addresses the technical quality of software. It includes test concepts and infrastructure, technical support concepts and structure (together with Support), a historical quality database, quality forecasting, and the execution of tests.

The product manager's orchestration responsibilities are dependent on the development methodology used, the organizational structure, the interfaces between development and product management, and the life cycle stage of the product. It typically includes agreements on the release scope, schedules, estimates and measurable product release targets such as performance or quality metrics. During development execution, it includes tracking of the execution of plans, tracking of product- vs detailed requirements, negotiations, and adjustments of plans - including scope changes (if needed) and the acceptance of release test results.

Literature: *Bass, L., et al. (2015); Forsgren, J., et al. (2018); Hassenzahl, M., & Tractinsky, N. (2006); McGrath, M. (2000, Chapter 3); ISO/AWI TR 9241-1 (1998); Kittlaus, H.-B. (2012); Kittlaus, H.-B. (2022, p. 247 ff); Lazzaro, N. (2004); P. Le Callet, et al. (2013); Shneiderman et al. (2013); Jabbari, R., et al. (2016); Luz, W.P. et al, (2018); Leite, L., et al. (2019, Surv. 52, 6, Article 127); Erich, F., et al. (2017); Skelton, M., & Pais, M. (2019, Ch.1-3)*

## 5.2 Marketing

Duration: 0:45 h

Marketing is responsible for all aspects in preparation and support of the product sales activities of a company, including the creation of product awareness and communication of the positioning of the product in the market. The actual split of responsibilities between Marketing and Sales may differ from company to company. In some companies, SPM and Marketing are organizationally combined.

Marketing planning addresses the development and negotiation of plans for all marketing-related activities during a given time frame, often a year, including respective budgets. The plans can be product-specific, or for groups of products. They need to be synchronized with corporate and product strategies and plans, and the sales plan.

Channel optimization means the selection, implementation and management of channels appropriate for a product. The management of marketing partners within the product ecosystem can be part of the marketing responsibility unless there is a dedicated partner management organization. Both require tight cooperation with SPM and alignment with the corresponding product strategy.

Value communication is the process of connecting defined customer values with identified target markets for the product. This is done via relevant communication and engagement tools conveying the value messaging in formats, content, and channels adapted to the customer's buyer journey.

Product launches mean the introduction of a new product, version or release to the market.

Marketing needs to orchestrate all activities that serve to create attention from existing and potential future customers, in the trade press, with market research agencies, and so on. Typically, SPM, Development, Sales, executives, partners, and sometimes customers are involved. SPM plays a key role upon agreeing target effects of the launch, which, in addition to the targets related to marketing, as well as sales and customer satisfaction should also include desired effects on user adoption, user behaviour, or scalability metrics.

Opportunity management means the continuous pursuit of identified business opportunities with the objective to turn those opportunities into concrete product success. This may include the formulation of product requirements, development and implementation of new product marketing and communication approaches, and tight cooperation with Sales. New business opportunities are identified through customer analysis, i.e. the frequent analysis of existing or potential customers or groups of customers with regard to new business opportunities.

Channel preparation means that the selected channels are enabled in time to sell a new product, version or release. It includes skills management and the provision of materials, web site, customer reference stories, etc.

Operational marketing means the execution of the marketing plan, tracking of the relevant measures, and correcting and optimizing the marketing plan and channel mix according to measurements tracking or when new insights or opportunities arise.

The product manager's orchestration responsibilities towards marketing are a core driver for continuous development of product value. Based on the defined value proposition, the orchestration includes the positioning of the product in the marketing strategy and marketing plan, and the cooperation around execution and optimization of the marketing activities, product launches, channel- and partner management. The orchestration of marketing is also an important source of input to product management. Opportunity management, customer analysis and direct customer marketing activities can feed into product planning. Participation in industry-, customer-, and marketing events is a source for qualitative validation of the value proposition. Marketing performance measurement also enables quantitative validation of values. This may be especially efficient for digital marketing channels, and digital tools used at events with many participants, where analysis of actions and reactions from the audience can be used to evaluate the relevance of different value proposition messages and even product concepts.

Literature: *Bech, H.P. (2015); Kittlaus, H.-B. (2022, p. 256 ff); Sandy, K. (2020, Ch.2 & 10); Anderson, J.C., et al. (2007, Ch.2-3); Osterwalder, A. (2014); Elis, S., & Brown, B. (2017, Ch.1 & 10); Weber, L. (2020, p75ff & p100ff)*

## 5.3 Sales and Fulfillment

Duration:   0:30 h

Sales is responsible for all sales activities at a company. The actual split of responsibilities between Marketing and Sales may differ from company to company. Fulfillment means making the product available to the customer for use. This can include online downloads, making Software-as-a-Service available to the customer, delivery of software on physical storage media or the distribution of packaged software products to retail outlets. Manufacturing is usually not an issue for pure software products or services except for the case of retail distribution. Fulfillment can fall under the responsibility of Sales, or a central fulfillment unit.

Sales planning addresses the development and negotiation of plans for all sales-related activities during a given time frame, often a year, including target values and incentives. The plans can be product-specific, or for groups of products. They need to be synchronized with corporate and product strategies and plans, as well as the Marketing plan.

Customer relationship management means the systematic management of a company's interactions with customers and sales prospects. This includes customer communication, knowledge management, and customer requirements engineering. CRM must not only focus on short-term sales success, but also on long-term customer relationships.

Operational sales means the execution of the sales plan, tracking of the relevant measurements, and taking corrective actions when measurements deviate from the plan. This includes offers and the negotiation of contracts, and the management of offers and contracts.

Operational fulfillment means ensuring smooth order and distribution processes, stable and easy online orders and distribution, and smooth and correct billing/payment. It does not include professional delivery services which are part of Delivery Services and Support.

The Sales orchestration includes the positioning of a product in the sales plan, in particular the incentives plan, and the cooperation around execution and optimization of the sales plan. It is also an important source of input to product management. Involvement in product requirements and customer commitments to secure the balance between short-term sales and long-term product strategy gives input to product planning. Handling deviations from standard terms and conditions, price structure or price levels, gives feedback on the perceived customer value. Participation in sales work and customer meetings is an opportunity for validation of the value proposition. Furthermore, product fulfillment and customer relationship management are key components of the total product and therefore need to be aligned with, and sometimes incorporated in, the strategy and product planning of the SW product.

Literature: *Kittlaus, H.-B. (2022, p. 265 ff); Brown, M., & Ellis, S. (2017); Sandy, K. (2020, Ch.2 & 10); Anderson, J.C., et al. (2007, Ch.2-3); Osterwalder, A. (2014); Weber, L. (2020, p106ff)*

## 5.4 Delivery Services and Support

Duration: 0:45 h

Delivery Services mean all customer-specific services provided to customers to help them become productive with the initial software product or when a new version is installed. This includes installation and tailoring services. Tailoring based on the product's tailorability options (see product architecture in section 5.1) is a customer-specific service that can mean a large project for configuration and customization.

Support refers to all product-related services provided to existing customers such as maintenance, training, operations, user help desk etc. Support provides technical support to customers, usually covered by maintenance or SaaS contracts.

Other customer-specific services like consulting are not considered product-related and therefore, are not part of a product organization's responsibility or a software product manager's responsibility.

Service planning and preparation address the development and negotiation of plans for all product service-related activities during a given time frame, often a year, including target values and incentives. The service plans need to be synchronized with product strategies, product plans, and marketing plans. Preparation includes development of a technical basis (if required), forecast of demand (with SPM), resource management, skills development, and the development of marketing material (together with Marketing).

Service Execution means the execution of the service plan, tracking of the relevant measurements, and taking corrective actions when measurements deviate from plan.

Technical support refers to the fulfillment of contractual obligations, i.e. maintenance contracts with license products or of service-related elements of SaaS contracts. The typical support structure is:
**Level 1:**  Help Desk
**Level 2:**  Technical Maintenance
**Level 3:**  Change Team (typically in or with Development)

Customer requests need to be categorized in defects, requirements, and non-technical problems, and documented in a customer issue database.

Operations is a key element in all products offered with the delivery model of SaaS (Software as a Service) or as a customer-specific managed service. In these cases, the vendor assumes the responsibility of operating the software at an internal or external data center, also known as a hosting service, and giving access to the customer. The quality requirements of this hosting service are defined in a service level agreement that is part of the contract with the customer (see section 2.4). The vendor's approach may include DevOps (see section 5.1).

Product-related documentation is required both internally and externally. Internal documentation includes documents such as specifications, technical manuals etc. This documentation is usually

developed by Development and is not intended for customers. External documentation refers to all documentation intended for use by people outside of the product-owning company such as end users, other ecosystem actors or service partners. For end users' guidance, printed and/or online manuals, help functions, or step-by-step instructions need to be specifically developed and require collaboration between UX design, software development, technical support, and marketing. This work needs to be orchestrated by the product manager.

Since many products are a combination of goods and services, the orchestration responsibility towards delivery services and support is often an important part of SPM. The product manager's orchestration responsibility includes the management of product-related services as part of the product offering, tracking of service execution, resource management and skills development. Requirements management of delivery services and support includes service-level agreements (SLA) for service provisioning, e.g response times on customer requests, and for operations, e.g. infrastructure availability. It also includes a compliance level of regulations and industry standards.

SPM also needs to provide volume forecasting for service planning, and information to support delivery services, support and operations for new product releases.

Literature: *Kittlaus, H.-B. (2022, p. 272 ff); Kowalkowski, C., & Ulaga, W. (2017, Ch.1 & 9)*

## 5.5 The Role of the Software Product Manager as Orchestrator

Duration:   0:30 h

One of the central tasks of SPM is to optimize the cooperation of all other functional areas with respect to product-related goals. The value of the total product - the combination of goods and services delivered to the customer - is provided based on deliveries and efforts from all these functional areas.

Orchestrating the work of these other functions is a two-way collaboration process in which the product manager, in most cases, has to act without formal authority. The product manager needs to develop work methods and skills for how to guide and influence these other units with common principles, engagement and agreements, and to evaluate the direction as well as the results. The goal is not the governance of these functions in itself, but a successful product.

Clear interfaces are needed in order to handle all aspects of the orchestration. Interfaces to stakeholders and other functional areas are usually an important component of a software product manager's role description.

The most important aspect of orchestration is the establishment of personal relationships with stakeholders.

The orchestration responsibility does not only apply to the functional areas discussed in 5.1 to 5.4, but also other areas. In particular, with software product managers of other products within the company, Finance, Controlling, Human Resources, and Research. Here the product manager's orchestration responsibility includes the alignment of product strategies and plans, research and innovation initiatives for both functional and technical innovations, resource management, and availability of correct and timely measurements. When expanding internationally, the product manager must ensure that resources are provided for localization activities for all target countries. Often, localization is done in the target countries. The product manager must then orchestrate the timely execution of localization.

Literature: *Fricker, S., et al. (2010); Kittlaus, H.-B. (2022, p. 245 ff); Raiffa, H., et al. (2007); Thompson, L. (2014); Lemay, M. (2017, Ch.2, 12); Sandy, K. (2020, Ch.2); Pichler, R. (2020, p.17ff, p.28ff, p.44ff, p.59ff, & p.74ff); Skelton, M., & Pais, M. (2019, Ch.7); Lemay, M. (2017)*

## Bibliography

**Recommended SPM Overviews:**

*Ebert, C. (2007): "The Impacts of Software Product Management". Journal of Systems and Software 80:850-861.*

*Fricker, S. (2012): "Software Product Management", in Mädche A., Botzenhardt, A., & Neer L. (eds.): Software for People - Fundamentals, Trends, and Best Practices. Springer.*

*Kittlaus, H.-B. (2022): Software Product Management - The ISPMA-Compliant Study Guide and Handbook (2nd ed.). Springer.*

*van de Weerd, I., Brinkkemper, S., Nieuwenhuis, R., J. Versendaal, & L. Bijlsma (2006): "On the Creation of a Reference Framework for Software Product Management: Validation and Tool Support". International Workshop on Software Product Management (IWSPM 2006).*

**Literature on specific aspects of SPM:**

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

*Almquist, E., Senior, J., Bloch, N. (2016): The Elements of Value. Harvard Business Review*

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

*Anderson, C., & Zeithaml, C. (1984): "Stage of the Product Life Cycle, Business Strategy, and Business Performance". Academy of Management Journal 27(1)*

*Anderson, J.C., Kumar, N., & Narus, J.A. (2007): Value Merchants*

*Bass, L., Weber, I., Zhu, L., (2015): DevOps: A Software Architect's Perspective, Addison-Wesley. Upper Saddle River*

*Bech, H.P. (2015): Building Successful Partner Channels. TBK Publishing*

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

*Berntsson-Svensson, R., Gorschek, T., Regnell, B., Torkar, R., Shahrokni, A., & Feldt, R. (2011): Quality Requirements in Industrial Practice – an extended interview study at eleven companies. IEEE Transactions on Software Engineering, vol. 99*

*Blank, S. (2013): Why the Lean Startup Changes Everything. Harvard Business Review, vol. 91, no. 5*

*Borg, M., Chatzipetrou, P., Wnuk, K., Alégroth, E., Gorschek, T., Papatheocharous, E., Shah, S.M.A., Axelsson, J. (2019): Selecting component sourcing options: A survey of software engineering's broader make-or-buy decisions. Information and Software Technology 112*

*Bosch J. (2019): How to develop software. LinkedIn, https://www.linkedin.com/pulse/how-develop-software-jan-bosch/*

*Bosch, J. (2009): From Software Product Lines to Software Ecosystems. 13th International Software Product Line Conference (SPLC 2009), San Francisco*

*Carlshamre, P., Sandahl, K., Lindvall, M., Regnell, B., & Natt och Dag, J. (2001): "An Industrial Survey of Requirements Interdependencies in Software Product Release Planning". 5th International Symposium on Requirements Engineering (RE'01)*

*Chen, S., Cheng, A., Mehta, K. (2013): "A Review of Telemedicine Business Models". Telemedicine and e-Health 19(4):287-297*

*Chesbrough, H.W. (2005): Open Innovation: The New Imperative for Creating And Profiting from Technology. Harvard Business Review Press*

*Clements, P.; & Northrop, L. (2015), Software Product Lines: Practices and Patterns. Addison Wesley.*

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

*Cooper, R.G., Edgett, S.J., & Kleinschmidt, E.J. (2001): Portfolio Management for New Products (2nd ed.). Perseus Books*

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

*Cusumano, M., Gawer, A., Yoffie, D.B. (2019): The Business of Platforms. Harper Business*

*Cusumano, M. (2004): The Business of Software. Free Press*

*Doorley, S., Holcomb, S., Klebahn, P., Segovia, K., & Utley, J. (2018): Design Thinking Bootleg. Stanford University*

*Duboc L., et al. (2019), Do we really know what we are building? IEEE 27th International Requirements Engineering Conference (RE). IEEE*

*Dunford, A. (2019): Obviously Awesome – How to Nail Product Positioning so Customers Get It, Buy It, Love It. Ambient Press*

*Ebert, C., Brinkkemper, S. (2014): Software product management - An Industry Evaluation. Journal of Systems and Software, vol. 95*

*Elis, S., & Brown, B. (2017): Hacking Growth*

*Erich, F., Amrit, C., & Daneva, M. (2017): A Qualitative Study of DevOps Usage in Practice. Journal of Software Evolution Process*

*Forsgren, N., Jumble, J., Kim, J. (2018): Accelerate - The Science of Lean Software and Devops: Building and Scaling High Performing Technology Organization. IT Revolution Press*

*Fotrousi, F., Fricker, S. (2916): Software Analytics for Planning Product Evolution. International Conference on Software Business (ICSOB 2016), Ljubljana, Slovenia*

*Fricker, S., & Schumacher, S. (2012): "Release Planning with Feature Trees: Industrial Case", 18th Intl. Working Conference on Requirements Engineering. Foundation for Software Quality (RefsQ 2012), Essen, Germany*

*Fricker, S., Gorschek, T., Byman, C., & Schmidle, A. (2010): "Handshaking with Implementation Proposals: Negotiating Requirements Understanding". IEEE Software 27(2):72-80*

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

*Gorschek, T., Wohlin, C. (2006): "Requirements Abstraction Model". Requirements Engineering 11:79-101*

*Haines, S. (2014): The Product Manager's Desk Reference. McGraw-Hill Education Ltd*

*Hassenzahl, M., & Tractinsky, N. (2006): User Experience – A Research Agenda. Behavior & Information Technology 25(2):91-97*

*Iansiti, M., & Levien, R. (2004): The Keystone Advantage – What the New Dynamics of Business Ecosystems Mean for Strategy, Innovation, and Sustainability. Harvard Business School Press*

*ISO/AWI TR 9241-1 (1998): Ergonomics of Human-System Interaction. Part 1: Introduction to the ISO 9241 Series*

*Jabbari, R., bin Ali, N., Petersen, K., & Tanveer, B. (2016): What is DevOps? A Systematic Mapping Study on Definitions and Practices. Proceedings of the Scientific Workshop Proceeding of XP'2016*

*Jansen, S., Cusumano, M., & Brinkkemper, S. (2013): Software Ecosystems: Analyzing and Managing Business Networks in the Software Industry. Edward Elgar Publishers*

*Jansen, S., Peeters S., & Brinkkemper, S. (2013): Software Ecosystems: From Software Product Management to Software Platform Management – From Start-up to SaaS Conglomerate. Life Cycles of Software Products (IWLCSP'13)*

*Khurum, M., Khurum, A., & Gorschek, T. (2007): A Model for Early Requirements Triage and Selection (MERTS) Utilizing Product Line Strategies. 11th International Software Product Line Conference, Kyoto, pp. 97-104.*

*Kim, W.C., & Mauborgne, R. (2015): Blue Ocean Strategy: How to Create Uncontested Market Space and Make the Competition Irrelevant. Extended Edition. Harvard Business School Press*

*Kittlaus, H.-B. (2020b): Platform – What Do We Mean? https://www.linkedin.com/pulse/platform-what-do-we-mean-hans-bernd-kittlaus/*

*Kittlaus, H.-B. (2020a): Increasing Diversity in Software Product Planning and Development Approaches. https://www.linkedin.com/pulse/increasing-diversity-software-product-planning-hans-bernd-kittlaus/*

*Kittlaus, H.-B. (2019): Customer-Specific Tailorability of Your B2B Software Product – the Good, the Bad and the Ugly. https://www.linkedin.com/pulse/customer-specific-taylorability-your-b2b-software-product-kittlaus/*

*Kittlaus, H.-B. (2012): Software Product Management and Agile Development, in A. Mädche, A. Botzenhardt, L. Neer (eds.): Software for People - Fundamentals, Trends, and Best Practices. Springer*

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

*Klemens, B. (2006): Math You Can't Use – Patents, Copyright, and Software. Brookings*

*Knapp, J., Zeratsky, J., & Kowitz, B. (2016): Sprint: How To Solve Big Problems and Test New Ideas in Just Five Day., Transworld Digital*

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

*Kowalkowski, C., & Ulaga, W. (2017): Service Strategy in Action. Service Strategy Press*

*Lazzaro, N. (2004): Why We Play Games: Four Keys to More Emotion Without Story. Report. XEO Design*

*Le Callet, P., Böller, S., Perkis, A. et al. (2013): Qualinet White Paper on Definitions of Quality of Experience. White Paper. European Network on Quality of Experience in Multimedia Systems and Services*

*Leffingwell, D. (2011): Agile Software Requirements. Addison-Wesley*

*Lehtola, L., Kauppinen, M., & Vähäniitty, J. (2007): Strengthening the link from business decisions to requirements engineering: Long-term product planning in software product companies. Proceedings of the 15th IEEE International Requirements Engineering Conference (RE'07), New Delhi*

*Leite, L., Rocha, C., Kon, F., Milojiicic, D., & Meirelles, P. (2019): A Survey of DevOps Concepts and Challenges. ACM Comput. Surv. 52, 6, Article 127*

*Lemay, M. (2017): Product Management in Practice. O'Reilly UK Ltd*

*Lucassen, G., Dalpiaz, F., van der Werf, J., Brinkkemper, S. (2016): Improving Agile Requirements: The Quality User Story Framework and Tool. Requirements Engineering, vol. 21, no. 3*

*Luz, W.P., Pinto, G., & Bonifácio, R. (2018): Building a Collaborative Culture: A Grounded Theory of Well Succeeded DevOps Adoption in Practice. ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*

*Maalej, W., Nayebi, M., Johann, T., Ruhe, G. (2015): Toward Data-Driven Requirements Engineering. IEEE Software*

*McGrath, M. (2000): Product Strategy for High-Technology Companies – Accelerating Your Business to Web Speed (2$^{nd}$ ed.). McGraw-Hill*

*Messerschmitt, D.G., Szyperski, C. (2003): Software Ecosystem – Understanding an Indispensable Technology and Industry. MIT Press*

*Mintzberg, H., Ahlstrand, B., & Lampel, J. (2008): Strategy Safari: The Complete guide through the wilds of strategic management (2$^{nd}$ ed.). Financial Times Prentice Hall*

*Moore, G.A (2014): Crossing the Chasm. Harper Business*

*Nagle, T.T., Hogan, J.E., Zale, J. (2014): The Strategy and Tactics of Pricing – A Guide to Growing More Profitably (5$^{th}$ ed.). Pearson Prentice Hall*

*Nagle, T.T., & Hogan, J.E. (2005): What Is Strategic Pricing? Strategic Pricing Group Insights (Monitor Group)*

*Osterwalder A., & Pigneur Y. (2010): Business Model Generation. Wiley*

*Osterwalder, A. (2014): Value Proposition Design. Wiley*

*Parker, G.G., Van Alstyne, M.W., Choudary, S.P. (2016): Platform Revolution. W.W. Norton*

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

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

*Pichler, R. (2020): How to Lead in Product Management - Practices to Align Stakeholders, Guide Development Teams, and Create Value Together. Pichler Consulting*

*Pichler, R. (2016): Strategize: Product Strategy and Product Roadmap Practices for the Digital Age. Pichler Consulting*

*Pohl, K., Rupp, C. (2011): Requirements Engineering Fundamentals. Rocky Nook Computing*

*Popp, K.M. (2015): Best Practices for Commercial Use of Open Source Software. Books on Demand*

Popp, K.M., & Meyer, R. (2010): Profit from Software Ecosystems: Business Models, Ecosystems and Partnerships in the Software Industry. Books on Demand

Porter, M.E. (1998): Competitive Strategy. Free Press

Pritchard, C.L. (2013): Risk Management: Concepts and Guidance (5th ed.). CRC Press

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

Rajlich, V. & Bennett, K. (2000): "A Staged Model for the Software Life Cycle". IEEE Computer 33(7):66-71

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

Ries, E. (2011): The Lean Startup. Random House

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

Sandy, K (2020): The Influential Product Manager. Berrett-Koehler Publishers

Schmidt, M. (2002): The Business Case Guide. Solution Matrix

Schwaber, K., Sutherland, J. (2020): The Scrum Guide - The Definitive Guide to Scrum: The Rules of the Game. https://www.scrumguides.org

Shneiderman, B., Plaisant, C., Cohen, M., Jacobs, S. (2013): Designing the User Interface: Strategies for Effective Human-Computer Interaction (5th ed.). Addison-Wesley

Skelton, M., & Pais, M. (2019): Team Topologies: Organizing Business and Technology Teams for Fast Flow. It Revolution Press

Sodhi, M.N., & Sodhi, N.N. (2007): Six Sigma Pricing: Improving Pricing Operations to Increase Profit. Pearson FT Press

Stompff, G. and Smulders, F. (2015), The Right Fidelity: Representations That Speed Up Innovation Processes. 19th DMI: Academic Design Management Conference

Thompson, L. (2014): The Mind and Heart of the Negotiator (6th ed.). Pearson Prentice Hall

United Nations, (2015), "The 17 Goals", https://sdgs.un.org/#goal_section

Weber, L. (2020): Software Product Management – Tips and Templates, Amazon

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

Zacarias, D. (2019): 20 Product Prioritization Techniques. Folding Burritos

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