# Syllabus

# REQB®
# Certified Professional for Requirements Engineering

Advanced Level
Requirements Management

Version 2.0, 2015

Requirements Engineering Qualifications Board

## Overview of Changes

| Version | Date | Comment |
|---|---|---|
| 1.0 | 15.03.2011 | First version of the syllabus |
| 2.0-20131009 | 09.10.2013 | New draft version of the syllabus.<br>Syllabus divided into “Requirements Management” and “Requirements Development”<br>Business Outcomes and Learning Objectives |
| 2.0-20131027 | 27.10.2013 | Introduction<br>Modification of RMs taking into account remarks from the working group. |
| 2.0-20140318 | 18.03.2014 | Draft edition of 2.0 version |
| 2.0 – RC1 | 29.04.2014 | First Release Candidate for review |
| 2.0 – RC2 | 30.11.2014 | Second Release Candidate for review |
| 2.0 – RC3 | 18.01.2015 | Release Candidate |
| 2.0 | 28.02.2015 | GA Release |

## Main Idea

The central theme for this syllabus was that the complexity of systems and software and our dependency on them continues to increase. The result is a high level of dependency on the freedom from errors in products. The Requirements Engineering Qualifications Board (REQB) has therefore decided to create uniform international standards in the area of Requirements Engineering. Standards are like languages – it is only if you understand them that you can work effectively. In order to create a uniform language in this important area of requirements engineering, international experts joined together in REQB and developed this syllabus. The main focus of this Advanced Level syllabus is the Management of Requirements, whereas another syllabus deals with the Development of Requirements.

## Acknowledgements

This document was produced by a core team from the Requirements Engineering Qualifications Board - Advanced Level Working Party (Edition 2014): Alexander Alexandrov, Andrey Konushin, Folke Nilsson (REQB Chair), Ingvar Nordström, Salvatore Reale (Agile Syllabus WP Chair), Alain Ribault (Advanced Level WP Chair), Karolina Zmitrowicz.

The core team thanks all the National Boards for their suggestions and inputs.

## Table of Contents

# 0 Introduction

## 0.1 Purpose of the Syllabus

This syllabus defines the training program to become a REQB Certified Professional for Requirements Engineering (CPRE) at the Advanced Level for Requirements Management. REQB developed this syllabus in cooperation with the Global Association for Software Quality (GASQ). The scope of this REQB program covers the process of requirements engineering for all types of IT-related products consisting of software, hardware, services, business processes, and documentation as well. Within this scope the Requirements Management syllabus covers the specific processes required for effective requirements management.

REQB provides this syllabus as follows:

1. To National Boards, to translate into their local language and to accredit training providers. Translation may include adapting the syllabus to the particular language needs and modifying the references (books and publications) to adapt to the local publications.
2. To Exam Boards, to create examination questions in their local language adapted to the Learning Objectives defined in this syllabus.
3. To Training Providers, seeking accreditation as REQB recognized training providers, to produce courseware. All areas of this syllabus must correspondingly be incorporated in the training documents.
4. To Certification Candidates, as preparation material for the certification exam (as part of an accredited training course or independently).
5. To the international requirements engineering community, to advance the profession of a requirements engineer.

## 0.2 Examination

The examination to become a Certified Professional for Requirements Engineering - Advanced Level Requirements Management is based on this syllabus. All sections of this syllabus can thereby be addressed by exam questions as well as the Foundation Level syllabus. The examination questions are not necessarily derived from an individual section; a question may refer to several sections.

The format of the examination questions is multiple choice.

Examinations can be taken after having attended accredited courses or in open examination sessions (without a previous course) and after having passed the Foundation Level examination (mandatory prerequisite). Detailed information regarding examination times can be found on REQB’s website (www.reqb.org), on the website of GASQ (www.gasq.org) or on the website of a local examination provider.

## 0.3 Accreditation

Providers of a REQB Certified Professional for Requirements Engineering Advanced Level course must be accredited by the Global Association for Software Quality. Their experts review the training provider's documentation for accuracy and compliance with the content and learning objectives of the syllabus. An accredited course is regarded as conforming to the syllabus. At the end of such a course, an officially Certified Professional for Requirements Management examination (CRM exam) may be conducted by an independent certification institute (according to ISO 17024 rules).

Accredited training providers can be identified by the official REQB Accredited Training Provider logo.

## 0.4 Internationality

This syllabus was developed in cooperation between international experts. The content of this syllabus can therefore be seen as an international standard. The syllabus makes it possible to train and examine internationally at the same level.

## 0.5 Business Benefits

Objectives and benefits of the REQB Advanced Level program for Requirements Management are presented in the following table:

| Objectives | Benefits |
|---|---|
| Obtain new key qualification | Any software, hardware or service solution is based on stakeholders' requirements and aims to satisfy specific business needs. To be able to deliver a solution compliant with the needs of the target group, proper requirements engineering is necessary. Managing the requirements in an efficient manner is necessary to achieve full success and deliver the best solution that meets the customer's needs while respecting costs and quality constraints. |
| Increase your customers' satisfaction | Customer satisfaction is achieved when they experience the solution that meets their expectations and needs. Improved requirements management minimizes discrepancies between the expectations and the perception of efficiency. Good requirements management provides the means to enhance the quality of the final product and to strengthen customer loyalty by providing what they want. |
| Minimize development and follow-up costs | Proper requirements engineering minimizes the project and product risks and helps avoid costs related to rework resulting from discrepancies between the customer's expectations and the delivered solution.<br><br>Proper requirements management reduces the cost for future improvements or corrective actions. |
| Competitive advantage | Requirements engineering helps deliver better products or services. This helps to meet the business needs and fulfill their expectations, gain confidence and loyalty from the target groups, and win a competitive advantage. |

This syllabus focuses on the following areas:

| Objectives | Focus |
|---|---|
| Requirements Management Process | Providing a deeper description of the requirements management sub-process, its activities, actors, deliverables, most important principles, and best practices for managing requirements of products. |
| Requirements Management Maturity | Providing an overview of the most important standards, that can be used to improve the efficiency of requirements management activities. |
| Requirements Management in Product Lifecycle Management | Explaining the principles of adjusting the generic requirements management activities to specific products lifecycle models. |
| Tools and techniques | Explaining the usage and benefits of techniques to manage requirements (baseline, traceability, impact analysis, change management, follow-up of a set of requirements). |
| Exercises | Writing a requirements management plan, setting up requirements traceability, conducting impact analysis for |

| Objectives | Focus |
|---|---|
| | requirements changes |

**Table 1 Objectives, benefits and main focus of the Advanced Level program for Requirements Management**

The Advanced Level Requirements Management certification program is suitable for all persons who are involved in product/business solution development and maintenance, including business and systems analysts, marketing teams, hardware/software developers, GUI designers, project managers, customer representatives, maintenance and technical staff, IT auditors and quality assurance representatives.

The main purpose of the Advanced Level – Requirements Management Certification program is to provide best practices, techniques and tools to manage and maintain the integrity of a set of requirements (answering to a customer's needs) throughout the lifecycle of a product.

## 0.6 Learning Objectives

The learning objectives of this syllabus have been divided into different cognitive levels of knowledge (K-levels). This makes it possible for the candidate to recognize the "knowledge level" of each point.

Each learning objective in this syllabus has a cognitive level associated with it:

- K1 Proficiency/Knowledge: Knowledge of precise details such as terms, definitions, facts, data, rules, principles, theories, characteristics, criteria, procedures. Students are able to recall and express knowledge.
- K2 Understanding: Students are able to explain or summarize facts in their own words, provide examples, understand contexts, interpret tasks.
- K3 Apply: Students are able to apply their knowledge in new specific situations for example, by applying certain rules, methods or procedures.
- K4 Analyze: Students are able to analyze new specific problems and give appropriate solutions based on their varied knowledge and skills

## 0.7 Business Outcomes

The first goal of an organization / company is to satisfy its customers.

The role of a requirements manager is to ensure the integrity of the set of requirements from the customer's needs to the solution requirements (needed for the development of the solution) and thus throughout the whole lifecycle of the solution.

After completing the REQB Advanced Level for Requirements Management certification program, a person can:

- BO01 Track and maintain the status of a set of requirements that are contracted between a customer and a supplier

- BO02 Identify requirements engineering activities and processes within an ordinary quality management system
- BO03 Understand all the activities involved in requirements management in detail and is able to implement them in a specific context, business, domain or project
- BO04 Understand the tailoring mechanism and be able to tailor the requirements engineering processes according to business needs as appropriate for the lifecycle model: V-model, iterative and Agile
- BO05 Define and implement the necessary roles in an organization working with requirements engineering activities
- BO06 Evaluate and select proper tools to meet business needs and utilize those tools to support the requirements management process
- BO07 Understand, apply and analyze requirement traceability to an acceptable depth according to needs
- BO08 Define change management procedures and handle requirements change management within different levels of the business context
- BO09 Plan and implement requirements management in an organization or project
- BO10 Identify, define and implement improvements in the requirements development activities

## 0.8 Level of Detail

This syllabus is intended to support internationally consistent training and examination. This syllabus comprises the following components to reach this goal:

- General instructional objectives describing the intention of the Advanced Level program of REQB certification
- Learning objectives for each knowledge area describing the objective cognitive learning outcome of the course and the qualifications that the participant is achieving
- A list of information to teach, including a description, and references to additional sources such as accepted technical literature, norms or standards, if required
- A list of terms that participants must be able to recall and understand. A list of individual terms is described in detail in the REQB “Standard Glossary of Terms used in Requirements Engineering” document.

Please note: In this syllabus, the term “product” is used throughout. Depending on the context, this term can mean a system (both software and hardware) or a software-only product or a combination of hardware, software, documentation and the services connected to it. A product is a composition of software, hardware and other outputs of the product development process, including such items as documentation, electrical schemas and source code.

The syllabus content is not a description of the entire requirements engineering field of knowledge. It is focused on the requirements management aspects of requirements engineering.

## 0.9 Organization of the Syllabus

There are seven major chapters and a final reference chapter.

The top-level heading for each chapter shows the topic that is covered within the chapter and specifies the minimum amount of time that an accredited course must spend on the chapter.

Learning objectives to be satisfied for each chapter are listed at the beginning of the chapter.

Within each chapter there are a number of sections. Each section has a minimum amount of time that an accredited course must spend in that section. Subsections that do not have a time associated with them are included within the overall time for the section.

# 1 Basics of Requirements Engineering (130 minutes)

## Terms

business analysis, business rule, constraint, functional requirement, mission analysis, need, non-functional requirement, problem domain, requirement, requirements development, requirements engineering, requirements management, solution domain

## Learning Objectives for Basics of Requirements Engineering

### 1.1 Requirements (50 minutes)

| | |
|---|---|
| RM-1.1.1 | Apply levels and types to classify a set of requirements (K3) |
| RM-1.1.2 | Analyze a requirement in its context and identify its type, level and related sources (K4) |

### 1.2 Requirements Engineering (80 minutes)

| | |
|---|---|
| RM-1.2.1 | Explain the scope of requirements engineering and how it is connected to business analysis (K3) |
| RM-1.2.2 | Justify, by analyzing a specific context, why requirements engineering adds value to the customer (K4) |
| RM-1.2.3 | Analyze problems in projects and justify why requirements management is important (K4) |

## 1.1 Requirements

### 1.1.1 Need, Problem and Solution

[Gilb] defines a need, as “something desired by a defined stakeholder”.

Stakeholders’ needs represent the view of end users and customers regarding what the business or enterprise operations must be to fulfill their needs.

The needs describe the domain of the problem, that is, the description of what a customer wishes to do in order to realize its business processes. The needs must be translated and formulated into business requirements (by adhering to the quality criteria of the requirements).

The domain of the solution is the answer to the needs of a customer (business requirements); it represents the set of functional and non-functional requirements that should meet and cover the business requirements.

The needs of the stakeholders are often not mature, thus the requirements engineer must be careful to express the needs as explicitly as possible.

[SEBOK] adapted a lifecycle of needs (deduced from Professor Shoji Shiba and Professor Noriaki Kano) describing the different levels of maturity:

- Real needs: needs according to the context in which the stakeholders work.
- Perceived needs: needs based on stakeholders' awareness that something needs to be improved, that some business opportunities must be reached or that some new values must be generated.
- Expressed needs: needs formalized from the perceived needs in a generic way and often prioritized.
- Retained needs: needs selected from the set of expressed needs. The selection is based on the prioritization, the feasibility and the cost to provide the solution. These retained "stakeholder intentions do not serve as stakeholder requirements, since they often lack definition, analysis, and possibly consistency and feasibility. Using the concept of operations to aid the understanding of the stakeholder intentions at the organizational level and the system operational concept from the system perspective, requirements engineering leads stakeholders from those initial intentions to structured and more formal stakeholder requirement statements" [ISO 29148]. The retained structured and more formal needs are considered as the stakeholders / business requirements.

The requirements engineer must ensure all implicit needs are specified explicitly, especially those that seem obvious because they are part of the business or regulations.

*[For trainings companies: explain, with examples, why needs non-explicitly expressed can imply problems in the provided solution (non-conformance to the stakeholders / business expectation).]*

### 1.1.2 Constraint and Requirement

A requirement [IEEE 610] is:

- A condition or capability needed by a user to solve a problem or achieve an objective.
- A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.
- A documented representation of a condition or capability as in (1) or (2).

In a set of stakeholders' requirements, some of them must be considered as non-negotiable such as [Wiegers05]:

- A business rule (considered as a business constraint): the limitations on the project's flexibility to implement the requested solution [BABOK] such as a policy, guideline, standard, or regulation that defines or constrains some aspect of the business. This is not a product requirement in itself, but the origin of several types of product requirements.
- A constraint: a statement of restriction that modifies a requirement or set of requirements by limiting the range of acceptable solutions. A constraint can be a business constraint or a technical constraint.

Examples of constraints include:

- Interfaces with the product environment (other existing systems)
- Regulations

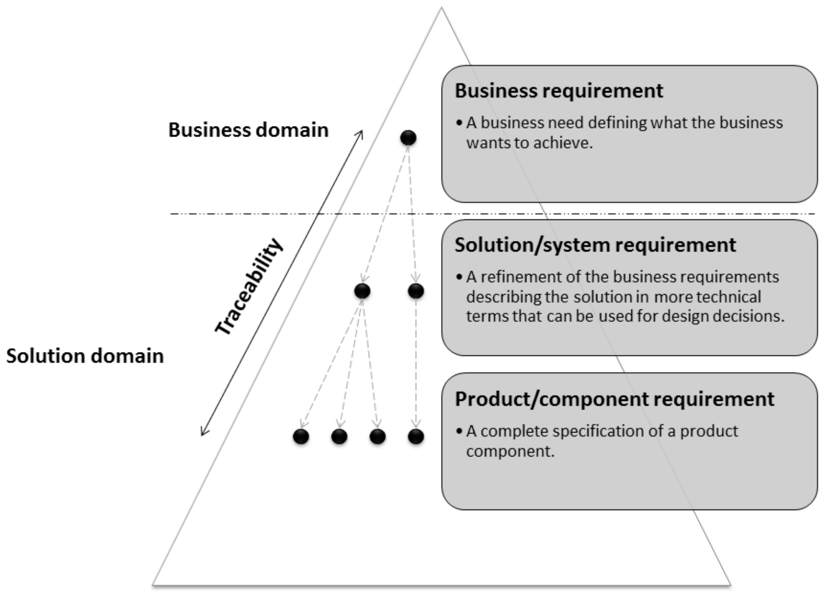
- Technologies used in the product environment (e.g., Java application server, specific database server, etc.)
- Capabilities and limitations of the end users (e.g., expectations on specific access to the product)
- Resources limitations of the product environment (e.g., memory size, bandwidth)

All the constraints must be identified as soon as possible because they restrict the definition of the solution to be implemented.

*[For trainings companies: explain, with examples, how constraints and business rules can constrain the foreseen solution.]*

### 1.1.3 Levels of Requirements

As presented in [REQB_FL_SYL], some of the common abstraction levels of requirements include the following:


Business domain
Solution domain
Traceability
Business requirement
• A business need defining what the business wants to achieve.
Solution/system requirement
• A refinement of the business requirements describing the solution in more technical terms that can be used for design decisions.
Product/component requirement
• A complete specification of a product component.

Business requirements are high-level requirements defining what the business wants to achieve but not how to implement it. These requirements express the customer's desires, needs and expectations and they are often referred to as high-level business or customer requirements. These requirements must be explicitly formulated by the customers.

Solution/system requirements are a refinement of the customer requirements describing the solution that is one of several possible ways to fulfill the customer requirements. The solution may contain non-IT related requirements for process changes or organizational/role changes. Solution/system requirements are the expression of the customer requirements in more technical terms that can be used for design decisions. These requirements are defined by the supplier of a solution and must cover all retained needs.

Product/component requirements describe the functions and characteristics of the solution. These requirements form a complete specification of a product component, including fit, form, function, performance, etc. These requirements are defined by the supplier, or one of its subcontractors, to refine the system requirements. They must be sufficiently specified to allow the developers to implement the solution without any more details.

*[For trainings companies: describe, with examples, how business requirements, solution/system requirements and product/component requirements can be classified and how this set of requirements can be structured.]*

### 1.1.4 Classification of Requirements

Stakeholders' requirements may be related not only to a software and hardware system, but to business processes or organizational structure as well. For instance, the purpose of a project may not only be implementing product components but also improving the organization's business processes.

Requirements can be divided into constraints and product requirements as follows:

- Constraints (business and technical) express expectations and needs related to costs, marketing, processing time, sales and distribution, organization, documentation, and the project. They also describe needs and limitations of the business processes. For example, business constraints may specify methodologies to be followed and the rules that an organization must respect.
- Product requirements consist of functional and non-functional requirements. Both can be regarded from the point of view of a user (external) or a customer and from the point of view of a development team (internal).

From [ISO 29148], requirements can be classified according to the following requirements types:

- Functional – describe the system or system element functions or tasks to be performed under certain conditions. Functional and non-functional requirements are discussed in the next section as well.
- Non-Functional – specify requirements under which the system is required to operate or exist, or system properties. They define how a system is supposed to behave. Quality requirements and human factors requirements are examples of non-functionality requirements.
  - Quality requirements – include a number of the 'ilities' in requirements such as transportability, survivability, flexibility, portability, reusability, reliability, and maintainability. The list of non-functional, quality requirements (e.g., "ilities") should be developed prior to initiating the requirements document. This should be tailored to the system(s) being developed. As appropriate, measures for the quality requirements should be included as well.
  - Human factors requirements – state required characteristics for the outcomes of interaction with human users (and other stakeholders affected by use) in terms of safety, security, performance, effectiveness, efficiency, reliability, maintainability, health, well-being and satisfaction. These include characteristics such as measures of usability, including effectiveness, efficiency and satisfaction; human reliability; freedom from adverse health effects.
- Interface – define how the system is required to interact with external systems (external interface), or how system elements within the system, including human elements, interact with each other (internal interface).

- Design constraints – constraints that limit the options open to a designer of a solution by imposing immovable boundaries and limits (e.g., the system shall incorporate a legacy or provided system element, or certain data shall be maintained in an online repository).
- Process requirements – these are stakeholder, usually acquirer or user, requirements imposed through the contract or statement of work. Process requirements include: compliance with national, state or local laws, including environmental laws, administrative requirements, acquirer/supplier relationship requirements, and specific work directives. Process requirements may also be imposed on a program by corporate policy or practice. Systems or system elements that are implementing process requirements, such as mandating a particular design method, are usually captured in project agreement documentation such as contracts, statements of work, or quality plans.

*[For trainings companies: give one or more examples for each type of requirement.]*

### 1.1.5 Functional and Non-Functional Requirements

**Functional Requirements**

Functional requirements specify what the system does. They specify the functions of the system that are perceived by the end user.

Functional requirements should be characterized by the following quality characteristics [ISO/IEC 25000]:

- Appropriateness
- Accuracy
- Compliance

**Non-Functional Requirements**

Non-functional requirements specify how the system delivers the functionality; they describe the quality attributes of the whole system or its specific components or functions. They may limit the solution, e.g., by requiring specific efficiency parameters.

Non-functional requirements are difficult to describe and so are often vaguely expressed or not documented at all. For example, if the requirements engineer has no knowledge and experience in describing such requirements as efficiency, he/she may not be able to document them properly or may not consider them at all. Due to the problems with expressing non-functional requirements, they may be difficult to test. Non-functional requirements must be expressed clearly and in a measurable manner.

Non-functional requirements are described by the following quality characteristics [ISO/IEC 25000]:

- Reliability
- Performance efficiency
- Operability
- Security
- Compatibility
- Maintainability
- Transferability

Non-functional requirements specify criteria that can be used to judge the operation of a system therefore they have a great impact on the customer's satisfaction in using the product. Functional requirements have to provide functions; non-functional requirements determine how easily and effectively the functions can be used.

Non-functional requirements are often called qualities of a system. Other terms for non-functional requirements include constraints, quality attributes, quality goals, quality of service requirements and non-behavioral requirements.

**Categories of Non-Functional Requirements**

There are two main categories of non-functional requirements:

- Execution qualities – can be observed at run time (e.g., security and usability)
- Evolution qualities – embedded in the static structure of the software (e.g., testability, maintainability, extensibility and scalability)

*[For trainings companies: give one or more examples for each functional and non-functional requirement category.]*

## 1.2 Requirements Engineering

### 1.2.1 Business Analysis

From [BABOK],

> Business Analysis is the set of tasks and techniques used to work as a liaison among stakeholders in order to understand the structure, policies, and operations of an organization, and to recommend solutions that enable the organization to achieve its goals.

[SEBOK] generalizes this definition as

> Mission Analysis: part of the larger set of concept definition activities – the set of systems engineering activities in which the problem space and the needs of the business or enterprise and stakeholders are closely examined; this occurs before any formal definition of the system-of-interest is developed, but may need to be revisited through the lifecycle. In fact, the activities of Concept Definition determine whether the enterprise strategic goals and business needs will be addressed by a new system, a change to an existing system, a service, an operational change or some other solution. The MA activity focuses on the identification of the primary purpose(s) of the solution (its "mission"), while Stakeholder Needs and Requirements activity explores what capabilities stakeholders desire in accomplishing the mission and may include some detail on the performance of certain aspects of the solution. Mission Analysis can be called Market Analysis or Business Analysis.

Mission Analysis must focus on the definition of operational actions (what the user wants to do with the system) and not on system functions (what the system must do to meet business requirements).

## 1.2.2 Requirements Engineering

Requirements engineering is a sub-discipline of systems engineering, focused on determining and managing the requirements of hardware and software products. [ISO 29148] completes this definition as an "interdisciplinary function that mediates between the domains of the acquirer (domain of the problem) and supplier (domain of the solution) to establish and maintain the requirements to be met by the system, software or service of interest."

Thus, the scope of requirements engineering is to define a solution to answer the needs of the stakeholders.

As stated in the document [ApproachRE], the starting point of requirements engineering is business analysis and the artifacts exchanged between requirements engineering and business analysis are business processes, business goals, business needs and business limitations.

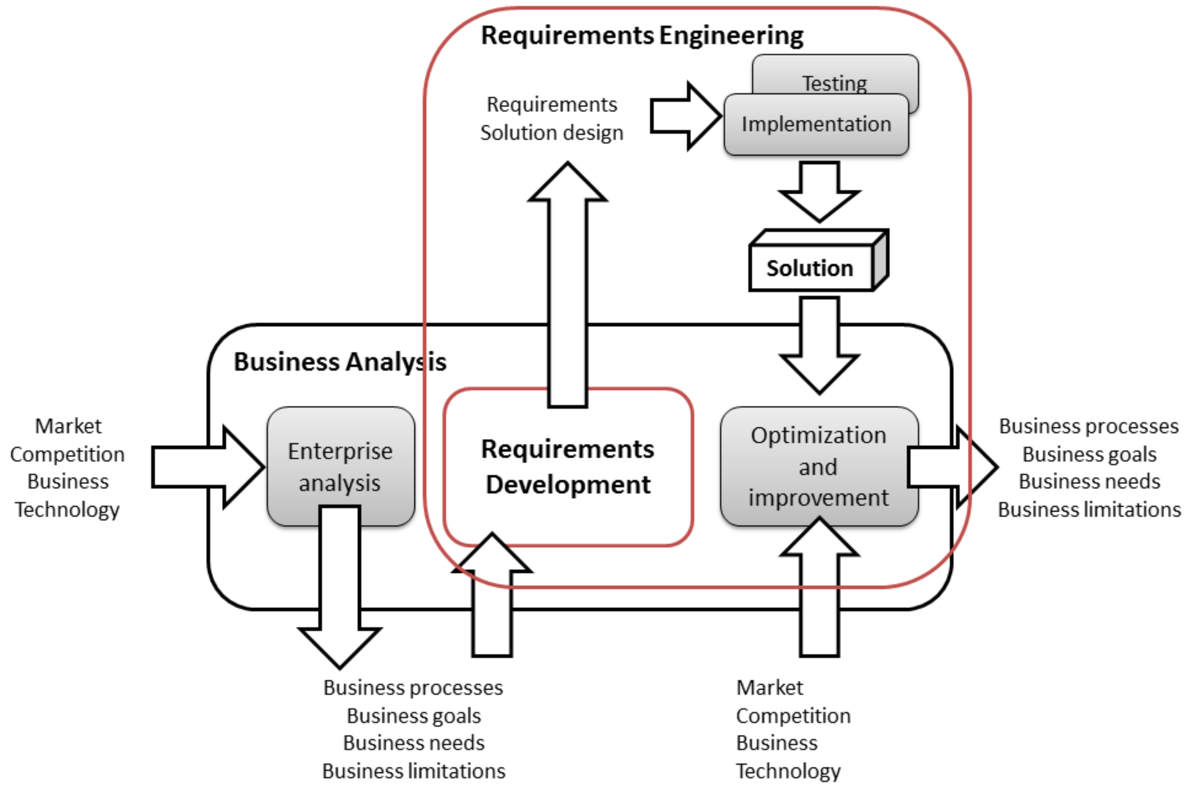


**Figure 1: The context of requirements engineering**

*[For trainings companies: give one or more examples of artifacts exchanged between business analysis and requirements engineering (such as modeled business processes)].*

Requirements engineering is a systematic requirements process (management and development) used between the domain of the customer and the domain of the supplier. It is a key activity undertaken in order to identify the needs of various stakeholders and treat those needs in a structured way. The main goal of requirements engineering is to establish and maintain a set of product requirements that meets and satisfies business requirements expressed by the customer. The result of requirements engineering is a structured hierarchy of requirements negotiated between the customer and the supplier in order to satisfy the needs of end users.

Requirements engineering includes processes needed for identifying, structuring and managing requirements. Specific activities covered by the general requirements engineering process include the following:

- Requirements elicitation
- Requirements analysis
- Requirements specification
- Requirements verification and validation
- Requirements traceability
- Configuration and change management
- Quality assurance

The requirements engineering process is a structured set of the activities listed above. The activities are categorized as part of the requirements management process or part of the requirements development process, depending on the purpose and the phase of the solution development.

The structure of the requirements engineering process depends on different factors, such as the organization culture and maturity, or the development process model used.

The generic requirements engineering process should be a starting point for each organization involved in solution development work as it provides the most crucial processes for handling requirements. This generic requirements engineering process can be adapted to the specific needs of an organization and considers the development process model to be applied. The result of this adaptation is the requirements engineering process in development process models. The planning and adoption part of requirements management is further described in Chapter 6.

The requirements development process and the requirements management process are interrelated. They can be seen as two sides of the same coin and cannot exist without each other. The requirements development process comprises all activities for the generation of the requirements and for the support of the development based on those requirements. The requirements management process provides the infrastructure and support to the requirements development process throughout the product lifecycle.

The purpose of requirements management is to manage baselines of an agreed set of requirements of the solution and to ensure alignment between those requirements and the project's plans and work products.

Requirements management constitutes both a working framework for requirements engineering as well as supporting processes for the requirements development process. The other role of requirements management is to establish and provide interfaces to the other development and managerial processes interfacing with requirements engineering such as project management, risk management, design, configuration management, change management and quality assurance.

Requirements development is a collection of activities, tasks, techniques, and tools to identify, analyze, document and validate requirements on the different abstraction levels. It includes the process of transforming needs into requirements as well as development (refinement of the high-level requirements) of a solution for those requirements. Requirements development is the main subject for the Requirements Development Advanced Level syllabus.

*[For trainings companies: give one or more examples of requirements engineering's contribution to satisfying customer's expectations].*

### 1.2.3 Requirements Management

The purpose of requirements management is to manage the baselines of an agreed set of requirements of the solution and to ensure alignment between those requirements and the project's plans and work products.

Requirements management is planning and maintaining the agreement between the stakeholders, in terms of the integrity and the accuracy of the requirements, by employing the following activities:

- Developing and adopting the requirements engineering processes
- Defining the roles and responsibilities for requirements engineering
- Evaluating and implementing tools necessary for effective requirements engineering processes
- Keeping project plans up to date with the requirements
- Change control – managing changes to the requirements baseline, through reviewing proposed changes and evaluating the likely impact of each change before approving it, and incorporating approved changes into the products and project in a controlled way
- Version control – managing document versions and requirements revisions
- Requirements status tracking – defining a set of status values for a requirement, and monitoring status changes throughout the project
- Requirements tracing – managing dependency links between requirements, and tracing requirements up to their sources and down to corresponding design, source code, and test cases

Some factors may have a negative influence on requirements engineering:

- On the internal side:
    - Lack of knowledge of the user's domain
    - Ineffective requirements engineering approach/methodology
    - Requirements engineers with insufficient experience and skill
- On the external side:
    - Lack of communication
    - Unclear and/or changing business objectives
    - Lack of knowledge about the software development process
    - Lack of involvement of users

The only thing we know for sure is that things are going to change and that includes requirements. This means that freezing requirements in many cases is unwise and unrealistic because of the following:

- A problem may not be fully described
- Stakeholders' understanding of the problem evolves constantly
- Aging systems need to be updated
- Regulations change
- Large systems can not be explored at one time

- A set of requirements is usually incomplete
- During the development, the operational environment can change (new hardware, software, interface)
- After the system is deployed, new requirements can emerge (maintenance phase)

Changes are inevitable. Requirements management is the process of understanding and controlling changes to systems requirements.

*[For trainings companies: give one or more examples of requirements management's contribution to solving some of the above issues].*

# 2 Context of Requirements Management (80 minutes)

## Terms

change management, configuration management, product decommissioning, product deployment, product maintenance, product management, project management, project monitoring and control, project planning, requirements tracking, risk management, technical solution, traceability

## Learning Objectives for Context of Requirements Management

### 2.1 Requirements Management Activities (15 minutes)

| | |
|---|---|
| RM-2.1.1 | Recall the activities of the requirements management process (K2) |

### 2.2 Requirements Management and Other Processes (65 minutes)

| | |
|---|---|
| RM-2.2.1 | Apply requirements management to mitigate known project or product risks (K3) |
| RM-2.2.2 | Analyze the risks for a specific requirements management context (K4) |
| RM-2.2.3 | Explain how requirements management is related to other processes including requirements development (K2) |

### 2.1.1 Requirements Management Activities

According to [CMMi], “the purpose of Requirements Management (REQM) is to manage requirements of the project’s products and product components and to ensure alignment between those requirements and the project’s plans and work products”.

### 2.1.2 Requirements Management in a Larger Context

Requirements management is mainly a collection of managing and supporting activities, which are needed to assure that the requirements development process is executed properly under the product lifecycle.  Requirements management activities also include change management and maintenance.

The requirements management process operates in a larger context and has strong relations with other processes including product management, project management, analysis and design, configuration management, testing, release management and maintenance. The relationship between these and other processes within the business context is discussed in section 2.2.

Products are usually developed as a result of the product development activities that are grouped together into a project. Therefore, the project itself should be analyzed from the requirements engineering perspective.

According to well-recognized research (e.g., the Chaos Report), one of the main reasons why projects fail is due to issues with requirements. Neglecting requirements engineering can result in low quality requirements and – as a consequence – a low quality final product. Careful and structured requirements management is a necessary part of any project and should be considered as a part of the overall project management.

An important challenge for product development is to handle the risks. This is usually done through the risk management process. The goal of risk management is to identify the risk and then manage it according to the mitigation plan. During requirements elicitation and while developing the solution proposal, all associated risks should be defined so the risk management process can properly address these risks.

Another important area of requirements management is requirements traceability**.** This activity deals mainly with defining and maintaining traceability of requirements. Traceability is necessary as requirements are not stable – they continue to evolve during the development lifecycle. Traceability provides a method for managing evolving requirements and other artifacts related to those requirements.

Traceability also supports configuration and change management as it allows analyzing the impact of changing a specific requirement or other configuration unit. In the case of more complex solutions, or businesses that change often, it is often not possible to manage changes in an effective way without traceability.

When defining the requirements management process, it is also necessary to define necessary quality assurance activities to be applied in order to ensure that the different requirements engineering processes and their products are of good quality.

As demonstrated above, requirements engineering deals with many tasks, from analysis of business processes, through identification, analysis and modeling of requirements, quality assurance, risk analysis and change management of requirements. In order to manage this effectively, some special roles for requirements engineering and specific competencies are needed.

## 2.2 Requirements Management and Other Processes

### 2.2.1 **Background**

Requirements management is not isolated. It is linked to other processes in order to exchange different types of information.

Throughout the product lifecycle, other processes use requirements management:

- Product management
- Product development
- Product deployment
- Product maintenance
- Product decommissioning

### 2.2.2 Requirements Management and Product Management

The main activities of product management are:

- Product definition
- Product roadmap

- Product release

During the product definition activities, the product manager utilizes requirements management to store high-level business requirements and wished features.

During the product roadmap activities, the product manager uses requirements management to define the different versions of the product and to allocate high-level business requirements and features to the versions.

During the product release activities, the release manager uses requirements management to set the scope of the release. During the development phase, the scope of a version of a product often evolves due to new requirements, deletion of features, etc.

### 2.2.3 Requirements Management and Product Development

Focusing on the product development phase, [CMMi] shows that the requirements management process interacts with the following process areas:

- Project Planning
- Project Monitoring and Control
- Risk Management
- Requirements Development
- Technical Solution
- Configuration Management

#### Requirements Management and Project Planning

The main goal of project planning is to define a project plan that is agreed to by all stakeholders.

Project planning and requirements management interact in the following areas:

- Establishing estimates – Requirements management is needed to maintain information regarding the scope of the project. The project is normally defined as a subset of a product including the applicable business requirements, system requirements and constraints from all stakeholders. The size of the project is determined by the information about the selected requirements and constraints (e.g., number, priority and complexity of requirements).
- Defining project lifecycle phases – Requirements management, via a management plan, must provide the description and the workload estimation and costs of the requirements engineering phases.
- Developing a project plan – Requirements management contributes to the definition of the project plan.
- Obtaining commitment to the project plan – Requirements management aligns the requirements management plan with the project plan.

#### Requirements Management and Project Monitoring and Control

The main goal of project monitoring and control is to provide, at any time, a clear view of the progress of the project. This information can be used to plan and implement any necessary corrective actions.

Project monitoring and control consists of:

- Monitoring the project in accordance with the project plan (following milestones and metrics) **–** Requirements management supplies information regarding the progress of the requirements engineering activities. Monitoring techniques such as requirements-driven development (e.g., using executable requirements, earned value, and burndown charts) can be used (see chapter 4).
- Managing corrective actions to closure **–** Specific corrective actions related to requirements engineering can be tracked via the requirements management process.

### Requirements Management and Risk Management

The fundamentals of risk management are described in [REQB_FL_SYL].

Risks can be mitigated by requirements engineering activities, but risks can also occur during requirements engineering [Lawrence].

The most common risks related to requirements are:

- Instability of requirements including scope changes – These risks result from the changing business environment (e.g., optimization or changes in business processes, internal procedures and regulations), or difficulties in collecting the right requirements from stakeholders (e.g., stakeholders may not be able to articulate their requirements at the beginning of the project resulting in incorrect and incomplete requirements).
- Missing requirements – These risks result from an ineffective requirements elicitation process (e.g., business analysis has not collected all necessary requirements due to lack of domain knowledge, ineffective elicitation techniques, no stakeholder involvement or lack of key stakeholder involvement). Missed non-functional requirements are particularly risky because fixing the defect is generally more expensive (e.g., requiring a new design).
- Low quality requirements specification – This risk may be caused by ineffective requirements analysis and documentation processes, tight schedules or no standards related to requirements specification within an organization. A low quality requirements specification will also affect other areas of the project (e.g., implementation and testing may take more time because the team will need to clarify any unclear parts of the specification).
- Requirements that do not provide real value [Gilb] – This risk occurs when the requirements describe functionality and solutions that provide no real value to the stakeholders. This may be caused by a lack of understanding of the business goals of the project (e.g., increasing sales to a defined amount).
- Communication issues – In many cases there are serious problems with communicating the requirements to the other members of the project team and continuing that communication if any changes are made. If effective communication does not occur, some groups of stakeholders may not be aware of the current status of the requirements.
- Inadequate customer representation – If requirements elicitation and negotiation are not conducted with the correct customer, there is a major risk of building an inadequate solution that will be rejected by the end users.
- Representing requirements in the form of design – If the requirements documentation includes design aspects, the scope of possible solutions may be reduced. Concentration on the design may result in requirements that do not define what the system must do. Validation goals ("is it the right product?") will be difficult to attain.

*[For trainings companies: give one or more examples of risks related to requirements and show how to analyze risks related to requirements management].*

Requirements-related risks can be mitigated using the following means:

- Adopting effective requirements elicitation techniques (e.g., brainstorming, workshops, prototyping)
- Adopting standards and best practices related to requirements engineering and adjusting them to the needs of the organization and the specific project
- Creating or adopting requirements specification templates
- Introducing reviews and audits as a part of the quality assurance activities
- Formalizing change management as a process to manage any changes to the requirements
- Establishing a communication plan describing the process of requirements communication together with the schedule and responsibilities.
- Instructing the whole team about the rules and procedures applying to the requirements management and ensuring they all understand and will apply them.

*[For trainings companies: give one or more examples of risks mitigated by requirements management activities].*

## Requirements Management and Requirements Development

Requirements development consists of the following:

- Describing three types of requirements: customer requirements, product and component requirements, and constraints
- Developing customer requirements
- Developing product, component and interface requirements
- Analyzing and validating requirements

Requirements development interacts with requirements management when:

- Planning the requirements development activities
- Monitoring the requirements development activities and requirements definition progress
- Tracing the different levels of requirements
- Managing changes to requirements
- Evaluating the effectiveness of the requirements development activities

## Requirements Management and Technical Solution

Technical solution consists of the following:

- Selecting product component solutions
- Developing the design
- Implementing the product design

During the development of the solution, requirements management is used to link the baseline of the requirements to the solution and to trace the requirements to the solution artifacts (design components, source code and documentation).

## Requirements Management and Configuration Management

Configuration management consists of:

- Establishing baselines (of all work products)
- Tracking and controlling changes (of all work products)
- Establishing integrity of the baselines (of all work products)

Requirements management interacts with configuration management when establishing and maintaining the integrity of the baselines and artifacts of requirements engineering (requirements and corresponding documentation).

### 2.2.4 Requirements Management and Product Deployment

One of the major errors during the deployment of a product in its operational environment is to not transfer the requirements engineering work products. It results in a loss of the knowledge of the product and could cause regression when modifications are required.

Requirements management must be used during product deployment to set the deployment baseline for all necessary and reusable artifacts produced during the development phase.

### 2.2.5 Requirements Management and Product Maintenance

Implementation of a solution that meets business requirements is not limited to the development phases. Changes are inevitable after the deployment of the product (e.g., new customers needs, defect resolution, modification of the operating environment, aging of technology). The maintenance must be managed in the same manner as development, following the same processes. Change management must be used to record all corrections or evolutions and to analyze the potential impact on solution artifacts.

Requirements management must be used during product maintenance to:

- Analyze changes to requirements
- Use traceability to identify the effect of the modifications
- Manage and maintain the integrity of the requirements baselines

### 2.2.6 Requirements Management and Product Decommissioning

Product decommissioning is the last stage of the product lifecycle before withdrawing the product from its operational environment. It is important to analyze all effects of decommissioning (e.g., modification of interactions between the product and other systems such as the suppression of a provided web service). Requirements management can be used to identify the possible effects of decommissioning by analyzing traceability of the external interface requirements.

*[For trainings companies: show in details all interactions (inputs/outputs) between requirements management and other processes].*

# 3 Roles and Responsibilities (50 minutes)

## Terms

change control board (CCB), contract, customer, hardware and software engineer, market analyst, project manager, quality manager, regulator, requirements developer, requirements manager, stakeholder, test manager, user

## Learning Objectives for Roles and Responsibilities

### 3.1 Roles in Requirements Management (10 minutes)

| | |
|---|---|
| RM-3.1.1 | Recall the different roles involved in requirements management (K1) |

### 3.2 Responsibilities Related to Requirements Management Activities (10 minutes)

| | |
|---|---|
| RM-3.2.1 | Recall the responsibilities of each stakeholder involved in requirements management (K1) |

### 3.3 Skills and Competencies of a Requirements Manager (30 minutes)

| | |
|---|---|
| RM-3.3.1 | Analyze a specific context and identify the special skills and competencies needed to complete requirements management activities (K4) |

## 3.1.1 Roles in Requirements ManagementBackground

"A stakeholder is any person or organization who is actively involved in a project, or whose interests may be affected as a result of project execution or project completion. Stakeholders can exercise control over both the immediate system operational characteristics, as well as over long-term system lifecycle considerations (such as portability, lifecycle costs, environmental considerations, and decommissioning of the system)" [Gilb].

## 3.1.2 Roles

Many roles are involved in requirements management because they have direct interests in the work products managed and/or they implement processes that interact with requirements management. Each of the roles should be listed in the requirements management plan. The requirements process is fundamentally interdisciplinary, and the requirements manager needs to mediate between the domain of the stakeholders and that of systems engineering. There are often many people involved besides the requirements manager, each of whom has a stake in the product. The stakeholders will vary across projects but will always include users and customers (who need not be the same). The typical roles are discussed below.

**Requirements Manager**

The requirements manager is the main role that implements the requirements management process. He is responsible for planning requirements engineering activities and for maintaining the integrity of the whole set of requirements and related work products.

The requirements manager:

- Identifies and schedules requirements engineering activities in a requirements engineering plan
- Monitors requirements engineering activities by tracking the status of the requirements to ensure alignment with the requirements engineering plan
- Defines requirements baselines and keeps their integrity by managing configuration and changes, tracing requirements and analyzing the impact of modifications

**Requirements Developer**

A requirements developer is a technically-oriented person mainly involved in the elicitation, analysis, documentation, and prioritization of requirements. He is involved in requirements management as he develops requirements, which are managed (tracked, traced and versioned) by the requirements manager.

**Customer**

The customer is responsible for defining and approving all requirements as well as approving all modifications to requirements. During the requirements identification activities, the customer is usually supported by business and systems analysts and requirements engineers. It is also important to look at customers in a wider scope such as stakeholders and reference groups. These groups comprise those who have commissioned the software or who represent the target market.

**Users**

This group comprises those who will operate the product. It is often a heterogeneous group comprised of people with different roles and requirements. Users are involved in requirements management as they participate in the description, evaluation and validation of requirements changes.

**Stakeholder**

Stakeholders can be individuals or organizations having a right, share, claim, or interest in a system or in its possession of characteristics that meet their needs and expectations [ISO/IEC/IEEE 29148:2011].

**Business Analyst**

After [BABOK], a business analyst is “a practitioner of business analysis”. Business analysis is the set of tasks and techniques used to work as a liaison among stakeholders in order to understand the structure, policies and operations of an organization, and recommend solutions that enable the organization to achieve its goals.

A business analyst is someone who analyzes the structure of an organization, designs its processes, and defines business models, business goals and business constraints.

**Market Analyst**

A mass market product will not have a commissioning customer, so market analysts are often needed to establish what the market needs and to act as a proxy for customers.

**Regulator**

Many application domains, such as banking and public transport, are regulated. Products in these domains must comply with the requirements of the regulations authorities.

**Project Manager**

A project manager is concerned with planning, budgeting, resourcing and scheduling the project. Knowledge of the requirements is essential for detailed planning and estimation purposes.

**Risk Manager**

The risk manager assesses the risk for an organization and for a product, and supplies needed information for risk mitigation decisions.

**Product Manager**

The product manager researches the market and defines the high-level features of a product to meet the market needs. He drives the development of products from the feasibility study through to marketing the product.

**Change Control Manager (Change Control Board (CCB)/Product Control Boards)**

The CCB is responsible for analyzing, evaluating and making decisions regarding requirements change requests based on the feasibility and impact to the project or product.

**Configuration Manager (CM)**

The CM is responsible for:

- Maintaining a matrix of all customer-approved requirements
- Overseeing the requirements change control process
- Applying changes to the requirements matrix
- Maintaining the modification history of the requirements

**Release Manager**

The release manager defines the different baselines and releases of a product, according to the product vision and market needs (e.g., which features must be the first on the market?). He is the guarantor of the integrity and the consistency of a product release and/or the integration of an application into an entire information system (applicative stage).

**QA/Test Manager**

The QA/test manager is responsible for:

- Verifying that the delivered product satisfies the customer's approved requirements
- Documenting the results of the requirements verification in a Test Summary Report
- Verifying the quality of the requirements specification

**Quality Manager**

The quality manager is responsible for verifying that the organization has the necessary processes, procedures and tools for requirements engineering and is proper trained for using them. He is responsible for audits and measurement of the requirements engineering processes/procedures. He is responsible of the development of a suitable management system included a requirements engineering process.

**Designer**

The designer defines a product architecture that meets the functional and non-functional requirements.

**Hardware and Software Engineers**

These individuals have a legitimate interest in profiting from developing the software by, for example, reusing components in other products. If, in this scenario, a customer of a particular product has specific requirements that compromise the potential for component reuse, the hardware and software engineers must carefully weigh their own stake against those of the customer. [SEBOK]

## 3.2 Responsibilities Related to Requirements Management Activities

### 3.2.1 Background

Customer involvement is necessary in the following cases:

- When the software is designed and developed from scratch
- When the purchased software product (COTS – commercial off-the-shelf) is customized according to the needs of the specific project (e.g., by adding new functions, modifying default functions)
- When the software is completed by another company and only deployed into the customer's environment

Customers should provide initial business needs and expectations together with the offer/service request. It is the supplier's responsibility to explore these needs and extract requirements on that basis. Such initial requirements are the basis for estimating the project costs and schedules, and should be documented in the contract with the customer. For each of these requirements, there should be acceptance criteria defined and approved by both sides. All stakeholders involved in the development of the product must provide their cost and schedule estimates and must participate in the review of the contract that is signed between the customer and the supplier.

From the requirements engineering point of view, there are three primary components:

- Request for proposal
    - The customer is responsible to define clear business needs and business requirements that are uniquely identified and documented in the "request for proposal". Each need/requirement should be accompanied by acceptance criteria.
    - The requirements developer must read this request for proposal and ask all questions necessary to clearly understand the customer needs.

- Offer
  - The requirements developer and other stakeholders are responsible for the development of the requirements that are uniquely identified and documented in the system specification.
  - The customer must review and approve the system specification to acknowledge that the supplier has understood the business needs and requirements.
- Solution
  - The requirements developer refines the system requirements into component requirements.
  - The requirements manager is responsible for the management of the whole set of requirements, the implementation of the traceability between the different levels of requirements, and the integrity and consistency of the requirements repository.

### 3.2.2 Contract

Contracts usually include, together with deadlines for the development and delivery of the product:

- The list of prioritized requirements
- A short description of the planned solution
- The acceptance criteria for each requirement
- The list of deliverables (documentation, code, working software)
- Other needs and expectations such as preferred technology to be used, resource requirements, etc.

**List of Prioritized Requirements**

Having a list of prioritized requirements in the contract has pros and cons for both the customer and the supplier.

In a sequential development project, a lawyer will enforce (via the contract language) the customer's wishes to articulate every possible case and the supplier will be expected to meet the stated requirement. This is driven by the assumption of a long lead-time before first deliveries.

When an iterative, incremental and/or Agile approach is contemplated in the contract, the requirements will be articulated in an iterative and evolutionary manner to minimize the risk of spending time and money in developing software for requirements that are not ultimately needed. It also recognizes that money may be better spent for requirements that were not known at the beginning rather than exhaustively describing those that are known. Some of the requirements identified and developed in a sequential development project may never be used, due to changing customer needs and priorities during the contract duration (e.g., change of market conditions, new competitors). In this case, after delivery of a system that "conforms to the contract," new requirements will be needed to address the true and current needs of the customer.

Different contract models also have pros and cons.

Fixed-price and fixed-scope contracts, that often come together with fixed duration, almost always have drawbacks for both the customer and the supplier because customers often do not get what they need and suppliers can easily lose money. With this least-favorite-but-common model, having a list of prioritized requirements is almost unavoidable regardless of the lifecycle model. In both cases the requirements manager should apply the best possible due diligence in terms of large and detailed upfront requirements analysis, acceptance test definition, and skillful effort estimation for all requirements. In fixed-price and fixed-scope contracts, changes to requirements are allowed only to replace existing requirements that are of equal effort.

Having a list of prioritized requirements becomes less important (and less recommended) with more flexible contract models and with the iterative and incremental approaches.  This provides advantages for both the customer and the supplier.

**Short Description of the Planned Solution**

Any description of the planned solution should be avoided in the contracts. It is better to include a summary of the scope, vision, and business motivation of the project or product in the contract preamble, together with the price, and payment model. The supplier can then refer to the scope and vision with fewer constraints on the solution, allowing more options to be explored with the customer.

**Acceptance Criteria for Each Requirement**

Acceptance criteria in the contracts can answer crucial questions in outsourced project work. Ambiguity around requirements is a possible source of conflict—and of litigation. On the other hand if the acceptance criteria for each requirement or the planned solution are too detailed it is a potential problem for both the customer and the supplier; the customer will give up the flexibility of changing the requirements to a certain extent, and the supplier may be inhibited by constraints that could decrease software quality.  Highly detailed acceptance criteria may also result in reduced communication with the customer.

It is better to define a framework for acceptance in the contract. For example, if using an iterative approach it might make sense to have acceptance be based on conformance to a prior agreed upon acceptance test set for an iteration.  In the case of a Scrum project, conformance to the "definition of done" [Schwaber][Larman] may comprise acceptance.

**List of Deliverables**

Contracted outsourced work is often assumed to involve low levels of transparency and trust, and a long time until working software is delivered. A classic contract response to this is to mandate a conventional quality management plan or deliverables list that defines a long checklist of documentation to provide.

Again, to have this list defined in a contract can have good and bad ramifications for the customer. In particular, if this list includes items that are not used by the customer (such as lengthy design specifications), this can require time from the supplier to create these items rather than allowing them to focus on delivering real value: working software. The list of deliverables should always be limited to only items that will provide real value to the customer.

**Other Needs and Expectations**

Most of these needs and expectations are usually non-functional requirements. They should be treated, from a contract point of view, as any other requirement as discussed above.

The requirements manager should play an active part, as far as possible, in any negotiation regarding the contract that can lead to an improvement in the contract model. Everything about the requirements in the contract should be focused to bring benefit to the customer first, through a greater overall flexibility, while allowing the supplier to focus on the value of the delivered software. This is not always possible because the discussion on the contract is often limited, but where there is room to negotiate, the role of the requirements manager becomes important to support the business functions that deal with the customer.

## 3.3 Skills and Competencies of a Requirements Manager

A requirements manager should possess the following skills:

- Methodological competence (i.e., practical knowledge of the requirements engineering process, methods, techniques and tools), in order to help define the requirements engineering process
- Knowledge about sources of best practices and the most important standards related to requirements engineering (e.g., a requirements manager should know which standards to use when defining a requirements management plan and metrics)
- Ability to enforce structure (e.g., implementing traceability between levels of requirements and conducting impact analysis for proposed changes)
- Moderation and negotiation skills, in order to participate in the evaluation of change requests and to estimate the correct costs and delays that may be incurred by a change
- Self-confident manner, necessary to maintain the integrity and the consistency of the requirements repository
- Ability to argue and convince (e.g., to persuade others that it is risky to accept a change request without renegotiation of the contract)
- Stress resistance, for example, before the release of a product to ensure that the right product has been developed correctly
- Rigor in the application of requirements management activities, as they are the basis to ensure the conformance of the solution to its business, system and component requirements

*[For trainings companies: analyze the specific context of an organization and identify the roles and competencies needed to manage the requirements set]*

# 4 Requirements Management Activities in Operation (455 minutes)

## Terms

Attribute, baseline, capitalization, change analysis, change consolidation, change control board, change initiation review, change management, change request, configuration management, impact analysis, release management, requirements communication, requirements management plan, reuse, traceability, volatility

## Learning Objectives for Requirements Management Activities in Operation

### 4.1 Requirements Engineering and Planning (65 mins)

| | |
|---|---|
| RM-4.1.1 | Explain why requirements engineering planning is important in requirements engineering (K2) |
| RM-4.1.2 | Apply the requirements engineering planning activity to a specific context (K3) |
| RM-4.1.3 | Analyze the context of a project and write the corresponding requirements engineering plan (K4) |

### 4.2 Requirements Tracking (80 mins)

| | |
|---|---|
| RM-4.2.1 | Explain, with examples, why requirements tracking is important in requirements engineering (K2) |
| RM-4.2.2 | Apply the requirements tracking activity to follow a set of requirements (K3) |
| RM-4.2.3 | Define the requirement attributes which need to be managed (K2) |
| RM-4.2.4 | Analyze a set of requirements and identify volatile requirements (K4) |

### 4.3 Change Management (65 mins)

| | |
|---|---|
| RM-4.3.1 | Explain, with examples, why change management of requirements is important when tracing the change requests of a set of requirements (K2) |
| RM-4.3.2 | For a given context, analyze the change management needs and define the appropriate activities (K4) |
| RM-4.3.3 | Write a change request for a requirement (K3) |

### 4.4 Configuration and Release Management (45 mins)

| | |
|---|---|
| RM-4.4.1 | Explain why configuration and release management of requirements is important to maintain the integrity of a set of requirements (K2) |

| | |
|---|---|
| RM-4.4.2 | Describe the configuration management activity for a set of requirements (K2) |
| RM-4.4.3 | Describe the release management activity for a set of requirements (K2) |

**4.5 Traceability and Impact Analysis (75 mins)**

| | |
|---|---|
| RM-4.5.1 | Explain, with examples, why traceability and impact analysis are important in the scope of requirements engineering (K2) |
| RM-4.5.2 | Analyze a set of requirements and define the traceability rules between those requirements (K4) |
| RM-4.5.3 | Analyze a set of requirements and identify the impact of a change to a requirement (K4) |

**4.6 Communication in Requirements Engineering (50 mins)**

| | |
|---|---|
| RM-4.6.1 | Analyze a communication plan in order to identify gaps and corrective actions (K4) |
| RM-4.6.2 | For a given project, plan the necessary communication activities and responsibilities (K3) |

**4.7 Process Capitalization and Reuse (75 mins)**

| | |
|---|---|
| RM-4.7.1 | Explain, with examples, why capitalization is important in requirements engineering (K2) |
| RM-4.7.2 | Recall what types of requirements can be capitalized (K2) |
| RM-4.7.3 | Explain, with examples, how requirements can be reused (K2) |
| RM-4.7.4 | Analyze a context and a set of requirements and identify the reusable requirements (K4) |

## 4.1 Requirements Engineering Planning

### 4.1.1 Background

Problems with requirements are one of the main reasons why projects fail. Neglecting requirements management can lead to the following issues:

- Requirements that change during the product development (does not apply to Agile approaches)
- Requirements do not fulfill the criteria and do not satisfy stakeholders needs
- Missing requirements
- Requirements scope creep
- Change in requirements without the necessary approvals
- Regression after requirements were modified without doing an impact analysis

Requirements management is necessary in IT projects and should be included in the project lifecycle and carefully planned.

### 4.1.2 Planning Requirements Engineering Activities

The purpose of a requirements management plan is to describe requirements engineering activities, roles and responsibilities mandatory to develop and manage requirements throughout the lifecycle of a product.

A template for the requirements management plan should include the following information:

1. Overview
2. Objectives
3. Identification of Requirements Scope
4. Roles and Responsibilities
    - 4.1. Customer
    - 4.2. Requirements Manager and Developer
    - 4.3. Project Manager
    - 4.4. Configuration and Release Manager
    - 4.5. Configuration (Change) Control Board (CCB)
    - 4.6. Developers
    - 4.7. QA/Test Manager
    - 4.8. Stakeholders
5. Processes and Procedures
    - 5.1. Requirements Identification
        - Pattern to follow to formalize requirements (and level of formalization)
        - Rule to identify requirements uniquely
        - List of identification techniques to be used
        - Organization of requirements repository
    - 5.2. Requirements Analysis
        - Analysis techniques to be used (including modeling)
    - 5.3. Requirements documentation
        - Templates to be used
    - 5.4. Verification and Validation
    - 5.5. Requirements Tracking
        - Indicators to follow
    - 5.5. Change Management
        - CCB organization
    - 5.6. Configuration Management
        - Rules on versioning and baseline management
    - 5.7. Traceability and Impact Analysis
        - Traceability policies
    - 5.8 Release Management
6. Risks for Requirements Engineering applied to the project
7. Requirements Management Tools
    - 6.1. Requirements tracking matrix
    - 6.2. Tools to use for requirements development and management
8. Resources and Schedule

### 4.1.3 Requirements Management Plan and Other Project Plans

Just as requirements management interacts with other processes, the requirements management plan has relationships with other project plans as follows:

- Project management plan: defines the whole scope of the project and its context
- Risk management plan: identifies risks, which must be taken into account in the requirements management plan. Moreover, modifications to requirements may be considered as new risks and treated as such
- Monitoring management plan: defines all project metrics comprising requirements metrics
- Configuration management plan: defines versioning and configuration rules to follow
- Change request management plan: defines change rules to follow and applicability to requirements
- Quality management plan: describes processes, procedures and roles to follow in order to apply an efficient requirements engineering process

*[For trainings companies: show how the requirements management plan interacts with other plans].*

*[For trainings companies: give an example of a written requirements management plan and train people to write a requirements management plan]*

## 4.2 Requirements Tracking

### 4.2.1 Background

Requirements are often not stable because of factors such as new customer expectations, new emerging technologies, mistakes, etc. One part of the requirements management process is dedicated to monitoring the status of the requirements.

Tracking the status of each requirement throughout the lifecycle of a product is an important aspect of requirements management.

### 4.2.2 Unique Identification of Requirements

One of the most important attributes of a requirement is an identifier that allows the requirement to be uniquely and unambiguously identified. This seems simple but without this unique identification, effective requirements management is not possible for the following reasons:

- It is difficult to manage a large set of requirements
- It may be impossible to trace requirements efficiently to other artifacts

The rules surrounding identifying requirements must be clearly defined and shared with all stakeholders.

### 4.2.3 Storing Requirements

Requirements have to be stored in such a way that they can be accessed easily and related to other system requirements. Possible storage techniques include:

- In a document or set of documents (e.g., requirements specification)
- In a specially designed requirements database

### 4.2.4 Attributes of a Requirement

In addition to specifying what is required, requirements also contain ancillary information, which helps manage and interpret the requirements. This should include the various classification dimensions of the requirement and the verification method or relevant acceptance test plan section. It may also include additional information, such as a summary rationale for each requirement, and a change history.

A requirement should contain the following attributes:

- Identification: unique means (e.g., number, name) to identify the requirement
- Statement: description of the requirement in natural language or using graphics / diagrams
- Creation date: date of the creation of the requirement
- Version number: version of the requirement
- Owner: person who is responsible for the requirement, including verifying and approving that the requirement has been correctly implemented and tested
- Status: current state of a requirement
- Status date: date of the assigned status
- Rationale: purpose of the requirement, reason why the requirement is needed
- Subsystem: component to which the requirement has been allocated
- Product release: planned release of the implemented requirement
- Acceptance criteria: criteria used to approve that the implemented solution conforms to the requirement
- Priority: relative importance of the requirement (e.g., must, should, could)
- Complexity: degree (e.g., high, medium, low) of complexity of the requirement
- Criticality: degree (e.g. high, medium, low) of criticality of the requirement
- Volatility: degree (e.g. high, medium, low) of change of the requirement
- Traceability: link (backward, forward, horizontal, vertical) between the requirement and other artifacts

### 4.2.5 Requirements Monitoring

A requirement's status changes throughout the lifecycle of a product. Different requirements may have different statuses as they are being reviewed, implemented, tested, approved, etc. The following is a list of sample status values:

- Proposed
- Approved
- Implemented
- Verified
- Deleted
- Rejected

It is useful to track the status of requirements and be able to determine the progress of the solution as the requirement statuses change. Using requirement status information to determine how well the solution is conforming to the requirements is called "requirements-driven development".

Other metrics can be used to manage a group of requirements:

- The number of requirements by priority

- The number of requirements by complexity
- The number of dependent, linked or related requirements
- The number of use cases per module
- The number of interfaces to external systems
- The number of requirements changes
- The number of change requests from different sources

### 4.2.6 Volatility / Stability of Requirements

Some requirements will change during the lifecycle of the software and even during the development process itself. It is useful to have some estimate of the likelihood that a requirement will change. For example, in a banking application, requirements for functions to calculate and credit interest to customers' accounts are likely to be more stable than a requirement to support a particular kind of tax-free account. The former reflects a fundamental feature of the banking domain (that accounts can earn interest), while the latter may be rendered obsolete by a change to government legislation. Flagging potentially volatile requirements can help software engineering establish a design that is more tolerant of change.

Requirements changes occur while the requirements are being elicited, analyzed and validated as well as after the system has gone into service.

In general:

- Stable requirements are concerned with the essence of a system and its application domain. They change slowly.
- Volatile requirements are specific to the instantiation of the system in a particular environment and for a particular customer.

[Kotonya] defines different types of volatile requirements:

- Mutable requirements: these requirements are subject to change due to modifications of the environment in which the system is operating (e.g., new regulations).
- Emergent requirements: these requirements are not completely defined because of a lack of information. New requirements emerge during the design and the implementation of the system (e.g., detailed requirements of human interfaces that could be modified after a first presentation of some screens).
- Consequential requirements: these requirements are modified because they were based on assumptions about how the system will be operated, but some of these assumptions could have been wrong. The end users discover the system and formulate new modifications.
- Compatibility requirements: these requirements change because they depend on requirements from other systems. As those systems change, these compatibility requirements must evolve (e.g., when a new type of smartcard is used, the banking system must be upgraded).

These volatile requirements always will occur, but good practices in requirements management can help to anticipate and master the changes (e.g., by using tracking and traceability of requirements).

## 4.3 Change Management

### 4.3.1 Background

Change management is central to the management of requirements. This section describes the role of change management, the necessary procedures, and the analysis that should be applied to proposed changes. Change management has strong links to the configuration management process (see section 4.4).

The change management process is the process of requesting, determining attainability, planning, implementing, and evaluating changes to a system, documents or other products of the project. The purpose of change management is to support the processing of changes and ensure traceability of changes.

Changes to the requirements may be requested at any time during the realization of the project as well as after the final product is released in the operational environment. Changes will always happen and it is important to plan changes in terms of process and time. An "average" project may have 15% of the total effort allocated to managing and implementing changes. In an Agile context, changes are expected to be much more frequent.

There are many sources for change. These include the following:

- Extension of existing functionality
- Defects found in the product/documentation
- Requested new functionality
- Change of existing functionality
- Changes resulting from external factors (organizational changes, regulatory changes, aging technology)

### 4.3.2 Change Management Process

The change management process includes the following activities:

- Identifying the potential change: to trace it to the artifacts that would be modified by the change
- Analyzing the change request: to check if the change is valid, to identify which requirements are affected by the change and to propose requirements modifications
- Evaluating the change: to estimate the cost of the change (including impact analysis) and negotiate with the customer
- Planning the change: in term of timeline and resource usage
- Implementing the change: including updating the traceability of the modified requirements and tracking any new requirements
- Reviewing and closing the change request: including verification and validation of the modifications

**Change Initiation**

After a change request has been raised, the change is categorized and prioritized based on the information available (usually the information obtained as a result of the impact analysis of the change). The change is then discussed according to the change initiation review to establish the next steps (accept or reject the change request).

The change initiation review is a set of guidelines and templates providing the change control board with an initial checkpoint before approving a change to the production environment.

The purpose of a change initiation review is to:

- Assess the key attributes of a change (priority, risk, effort) against standards, policies, and quality metrics
- Evaluate the completeness of preliminary rollout, training, support and cost/benefit plans
- Consider whether to accept the change for implementation and deployment
- Consider whether to approve plans for operating and supporting the change (such as training)
- Consider whether to approve the plans required for the readiness of the target production environment to operate and support the deployed change

The change initiation review provides results used by the Change Control Board to make a Go/No-go decision to approve the change request and initiate the development of the solution.

If the decision is "Go" the change continues into development. In case of a "No-go" decision, the change is returned (with justification of the rejection) to the change initiator for verification, more information, rework or closure.

**Change Analysis**

Change requests must be analyzed to identify and estimate (cost, duration) the impact of the change. In order to facilitate the impact analysis, traceability between all the artifacts (work products like component designs, source code, tests, etc.) must be managed.

**Change Prioritization**

The change requests must be prioritized. Prioritization supports the CCB in the process of reviewing changes and deciding on implementation. Prioritization should be aligned with the roadmap, if one exists.

Change requests may be prioritized based on the Kano model for quality factors. The Kano model of customer satisfaction divides product attributes into three categories: threshold, performance and excitement (sometimes called obvious, required and surprise). A competitive product supplies basic attributes, maximizes performances attributes, and includes as many "excitement" attributes as possible taking into account the available budget. The Kano model may be used in the change prioritization process to divide change requests into three categories, known as: "must have", "should have" and "nice to have".

The process of change prioritization includes the following steps:

- Examining the requirements list – All requirements affected by the change request should be examined. It must be stated if the requirements affected by the change are in the threshold, performance or excitement categories of the Kano model. If the change request is adding a new requirement, the Kano model category should be determined for the new requirement.
- Establishing the priority – The priority should be determined for each new change request. Usually the priority is defined as:
  - High (for the threshold requirements created or affected by the change request)
  - Medium (for the performance requirements created or affected by the change request)

  - Low (for the excitement requirements created or affected by the change request)
- Expediting the change request (optional step) – This may be done if the change is critical for the users or corrects a serious defect(s) in production. In that case the priority should be set to "Expedite" and the change request should be reviewed immediately by the CCB.
  It is important to remember that the expedite priority should be used rarely and only if really needed as expedited change requests can decrease the overall productivity of the supplier.
- Reassigning the change request – The prioritized change requests should be reassigned to the release manager. Then they are ready for review by the CCB.

**Change Consolidation**

Change consolidation is the consolidation of changes from multiple sources into one change**.**

The goal of consolidating changes is to:

- Manage changes that are requesting similar features but are coming from different stakeholders
- Manage changes related to the same area of the functionality/module in order to avoid conflicts (as one change requested for a specific area may conflict with other changes submitted for the same area)
- Combine a number of minor changes requesting the same modification across the whole application (such as changing the page header's colors, logos, etc.)

### 4.3.3 Change Request

A change should be submitted as a formal change request document (often referred to as a Request For Change (RFC)). This document should describe the reason for the change and the requested solution together with additional details such as:

- The name of the person/department or other entity requesting the change
- Submission date
- Planned date of implementation of the change
- Urgency of the implementation

Changes can be a request for a new feature or a defect fix, but they are normally handled in the same way.

The originator of a change may be any stakeholder on the customer or supplier side: users, customers, project managers, business analysts, developers, testers, architects, etc.

### 4.3.4 Change Control Board

The change request should be submitted to the members of the CCB, who will analyze the request and decide on further actions.

The role of the CCB is to analyze potential changes and make decisions regarding change requests. The CCB is a group of individuals within a project group who is responsible for deciding if and when changes are to be made in regards to work products or scheduled events.

The CCB for a project may have members from any of the following groups:

- Project management
- Business and systems analysis

- Development
- Quality assurance
- Business management, if applicable
- Customer representative, if applicable

In the CCB for a product, the following roles may be involved in addition to the above:

- Product manager
- Market representatives
- Product engineers or requirements engineers
- Product maintenance manager

It is important to ensure that members of the CCB represent both the supplier and customer side of the project, and individuals have knowledge in different domains (technical, business, etc.)

Large changes of the requirements can represent a contractual change; therefore, careful analysis of the potential impact of the change may be crucial.

The CCB decides about changes in two steps. The first step is to analyze the impact of the proposed change. After completing this evaluation, the CCB either approves or rejects the change. In some cases the CCB may request more information before making a final decision or may defer the decision until some other occurrence that would affect the choice. Significant and complex changes that will affect baselines should be always be reviewed by the CCB.

## 4.4 Configuration and Release Management

### 4.4.1 Background

The requirements management process identifies requirements as configuration items and manages them using the same configuration management practices as other work products from the lifecycle.

The aim of the configuration management process is to establish and maintain the integrity of all the artifacts that comprise the requirements, and make them available to all the stakeholders.

### 4.4.2 The Requirements Baseline

During the requirements development activities, business, system and component requirements, both functional and non-functional, as well as related documents are produced. After they are reviewed and approved, those work products constitute a requirements baseline. The baseline is a subset of requirements and associated documentation to which the stakeholders have agreed.

### 4.4.3 The Product Release

The baseline is often linked to the definition of the contents of a specific planned release or a development iteration. The release management activity is used to define the different versions of the product that will be deployed to the operational environment.

### 4.4.4 Configuration Management

Configuration management is the versioning, baselining, labeling and tracking of the artifacts produced during the whole product lifecycle.

Version control is an essential aspect of requirements management. All requirements documents should include a revision history and a version indicator. Specific tools are sometimes used to assist with version control activities. These help track the history of changes, the details of the changes and the rationale behind the change requests.

A version control process should adhere to the following guidelines:

- Each revision of the requirements document must be uniquely identified
- The version number must change whenever a change is made, even a minor one
- A revision history is maintained in every document
- Only designated people are allowed to make and approve changes in the requirements documents
- There must be a simple way for someone to check that a version of a document is the latest one

*[For trainings companies: give examples of requirements repositories and associated work products and describe how configuration and release management should be utilized for this set of artifacts].*

## 4.5 Traceability and Impact Analysis

### 4.5.1 Background

Traceability is a technique that supports the quality of requirements. Traceability allows verification that each requirement is associated with the following:

- Components during the definition of the product design
- Source code during the development phase
- Test cases during test execution

If a requirement is not traceable to the above work items, it may mean that the requirement was not implemented (perhaps because it was not feasible to do so) or not tested. Any gaps in traceability must be researched and rectified to avoid a gap in the implementation.

Traceability is an important administrative activity as well as a prerequisite for verification and validation. It is also necessary for the change management process when the potential effects of a change are analyzed.

### 4.5.2 Backward and Forward Traceability

[Jarke] defines four types of traceability links:

- Forward to requirements – from business requirements to system requirements: to know the coverage of customer needs by the solution and which system requirements are affected by a modification in a business requirement
- Backward from requirements – from system requirements to business requirements: to know the origin and the justification of each system requirement
- Forward from requirements – from system requirements to other work products: to know how system requirements are satisfied by the designed and implemented solution
- Backward to requirements – from other work products to system requirements: to know and justify the creation of each designed component and each piece of source code

*[For trainings companies: give examples of traceability links]*

### 4.5.3 Traceability Policies

In the requirements management policy, described in the requirements management plan, the rules to be followed for traceability information need to be defined. This traceability policy should describe the links to maintain between artifacts and how information traceability must be maintained and presented to the different stakeholders.

*[For trainings companies: give examples of traceability policies]*

*[For trainings companies: give an example of a requirements repository and identify the traceability rules between the requirements]*

### 4.5.4 Requirement Traceability Matrix (RTM)

A useful matrix used in quality control, is the requirements verification matrix or requirement traceability matrix.

The RTM is a tool for ensuring that the scope of the project, its requirements and outcomes, remains unchanged when compared to the baseline. RTM monitors (traces) the outcomes by establishing a thread for each requirement from the project's initiation to the final implementation.

### 4.5.5 Impact Analysis

Impact analysis is performed to evaluate the possible effects of a proposed change in order to make a reasonable decision whether or not to implement the change.  Change impact analysis can be defined as:

- Identifying the potential consequences of a change, or estimating what needs to be modified to accomplish a change [Bohner]
- The evaluation of the many risks associated with the change, including estimates of the effects on resources, effort, and schedule [Pfleeger06]

Knowledge of the design details and risks associated with the proposed modifications are critical to performing accurate impact analysis within the change management processes.

The purpose of impact analysis is to identify the consequences of implementing a requested change in the context of the whole software system or business process. These consequences can be expressed as a list of risks related to the change and usually include estimations on cost, effort and schedule for the change implementation.

Impact analysis techniques can be classified into three types [Kilpinen]:

- Traceability: analyzing links between requirements, specifications, design elements and tests cases to determine the scope of a change
- Dependency: analyzing links between parts, variables, logic, and modules to determine the consequences of a change
- Experiential: determining the impact of changes based on expert knowledge

Supporting techniques are:

- Review meetings
- Informal team discussions
- Individual engineering judgment

The result of impact analysis should be communicated to the stakeholders involved in change processing. The CCB then uses this information to help form their decision to approve, reject or defer the change.

*[For trainings companies: give an example of a requirements repository and a change request and identify the impact of the requested modification]*

## 4.6 Communication in Requirements Engineering

### 4.6.1 Background

Requirements communication includes activities for expressing the output of the requirements development and management to the stakeholders. Communication of requirements is an on-going and iterative activity, including presenting, communicating, verifying, and obtaining approval of the requirements from the different stakeholders.

Communicating requirements is one of the major tasks of requirements engineering. The aim is to not only identify and document the customer's requirements, but also to bring the stakeholders to a common understanding of the requirements and the resulting solution.

Clear and effective communication is essential, as the stakeholders may have different knowledge and represent different domains. The role of a requirements engineer is to communicate requirements in a way that allows all stakeholders to gain the same understanding of a particular requirement. To ensure this, the requirements engineer must consider what communication approach is appropriate in a given situation.

Requirements engineering deliverables are inputs to many other project phases and processes, such as establishing the system architecture that will allow meeting the business goals, creating detailed functional and non-functional system specifications, and planning and executing QA activities.

Requirements engineering provides information to the following processes:

- Management decisions
- Project management (scope planning, scheduling, and estimating development and testing)
- Systems analysis
- Design (system specification and architecture)
- Implementation
- Testing
- Maintenance

The following roles are affected by the results of requirements engineering activities:

- Stakeholders
- Project manager (controlling the project schedule and scope)

- Systems analysts and developers (planning and designing the implementation)
- Architects (planning the system architecture, integration, etc.)
- QA staff
- Testers

Requirements engineering activities and deliverables can be communicated in both formal and informal ways. Common methods of communication include:

- Plans
- Specifications
- Workshops
- Presentations
- Reviews
- Change requests

Any communication activity should take into consideration the focus of the communication (e.g., needs, information, and consequences). Having this information, the requirements engineer can decide what the appropriate delivery method is, the appropriate audience, and how to present the information. For each communication, the requirements engineer must decide the most effective form of communication for both the topic and the stakeholder.

There are many different factors that should be considered when planning requirements engineering communication. These factors include:

- Type of project
- Communication formality
- Communication frequency
- Geographical location
- Culture

Different types of projects require varying amounts of documentation, and often have diverse processes and different deliverables.

Communication formality varies between projects, project phases and stakeholders. Communication tends to be more formal when the project is large, or its domain is complex, or the project itself is considered to be critical or strategic, or dependent on legislation, sector standards, or agreements. Some stakeholders may require formal communication regardless of the project type.

Communication frequency may vary among stakeholders for every form of communication.

Geographic disparity can also be a factor that limits communication possibilities, especially when stakeholders live in different time zones.

## 4.7 Process Capitalization and Reuse

### 4.7.1 Background

Capitalization and reuse is always a goal for those wishing to improve their productivity. The first goal is to capitalize and reuse source code but many of the other artifacts produced during the lifecycle of a product have potential reuse. The capitalization and reuse of requirements can improve the productivity of the organization and the quality of products (same requirements used in different products).

Some contexts and opportunities are more auspicious to bring reuse into operation:

- Software product lines
- Replacement of systems with other systems providing the same functions
- Common non-functional requirements in a organization (e.g., operational requirements, performance requirements)
- Common features among different products (e.g., same authentication and role-based feature)
- Same regulations and business rules across products
- Project requirements coming from a well-defined quality management system (e.g., milestones, document template conformance)

Capitalization and reuse means saving costs and decreasing defects by taking feedback and work products from work previously done within a project or between projects.

There are different ways to capitalize and reuse requirements:

- Copy and past requirements from a previous project into a new project
- Reuse an entire part of a product from the requirements through design, implementation and testing

Reuse is not easy to achieve. It takes more time to produce good reusable requirements with efficient practices for the structure and the formalization. Some rules must be followed in order to take advantage of reuse [Wiegers06]:

- Use a requirements repository with functionality for requirements information storage, search and filtering
- Define and respect quality rules to formalize requirements and quality characteristics of requirements
- Define and maintain traceability between requirements to justify each requirement and to promote the reuse of requirements (e.g., to conform to the same regulations)
- Define and maintain a glossary to reuse the same vocabulary (e.g., definition of a domain specific language)
- Spread the culture of reuse in the organization to facilitate and encourage stakeholders to formulate and structure reusable requirements

### 4.7.2 Capitalization of Requirements

Capitalization is the action of keeping some best practices and artifacts in order to reuse them afterwards. Non-functional requirements are better candidates for capitalization because they can be specified in such a manner that they can be applied to different projects. For example, performance requirements to be applied to all website products developed for different customers, can be defined for an organization.

### 4.7.3 Reuse of Requirements

*"Why re-invent the wheel?"*

Reusing requirements has many benefits. Among these are:

- Mutual or shared development costs

- Better estimation of workload
- Better mastery of milestones
- Shorter product development phases due to less defects, less rework, increased delivery frequency
- Better satisfaction for the customer and end users because of the ability to demonstrate similar functionality prior to developing the final product

### 4.7.4 Types of Requirements and Information to Reuse

Depending of the context, different types of requirements assets / work products could be reused [Wiegers05]:

- Within a project or application: business rules (constraints) and business requirements, specific non-functional requirements
- Across a product line: vision (business objectives), business processes, requirements for environments (development, testing, operational, decommissioning)
- Across an organization: business rules, stakeholder's profiles, glossary, non-functional requirements
- Across a business domain: requirements from regulations (constraints), business processes, glossary (domain specific language), requirements on assets (objects and data modeling), specific non-functional requirements (e.g., security requirements in a military domain)
- Within an operational environment: requirements on interfaces (e.g., with a monitoring tool), requirements on technical infrastructures (e.g., requirements on web applications and database servers)

Capitalization and reuse must be taken into account early and carefully because requirements can be created to be product specific or too detailed where they could have been better created as reusable.

*[For trainings companies: give context examples and demonstrate how requirements (which ones) can be reused].*

# 5 Quality Assurance and Process Improvement (135 mins)

## Terms

CMMi, maturity models, REQM, quality assurance

## Learning Objectives for Quality Assurance and Process Improvement

### 5.1 Quality Assurance (70 mins)

| | |
|---|---|
| RM-5.1.1 | Define the specific quality assurance activities necessary to ensure the quality of the product meets the expectations of the customer (K3) |
| RM-5.1.2 | Analyze a non-compliance issue concerning requirements management and identify corrective actions (K4) |
| RM-5.1.3 | Define the specific quality assurance activities related to the requirements engineering process that are necessary to ensure the quality of the product meets the expectations of the customer (K3) |

### 5.2 Maturity Models and Requirements Management (65 mins)

| | |
|---|---|
| RM-5.2.1 | Understand and apply the best practices described in the CMMi REQM key area to define a requirements management process for a specific context (K3) |
| RM-5.2.2 | Explain how to improve requirements management by using examples of requirements engineering process improvement models (K2) |
| RM-5.2.3 | Analyze a requirements management process and identify ideas for improvement (K4) |

## 5.1 Quality Assurance

### 5.1.1 Background

Software quality assurance provides assurance that the software products and processes in the project lifecycle conform to their specified requirements. This is done by planning and performing a set of activities together with using selected tools and techniques to deliver confidence that quality is being built into the products of the software development process.

One of the main concepts of quality assurance is to ensure the best possible quality by employing specified activities at each stage of the project to identify potential problems as early as possible. The role of quality assurance is to ensure that processes are appropriate and correct (resulting in providing desired outputs and performed according to the process requirements) and implemented according to plan. The level of quality is measured by relevant measurement processes as defined in the quality assurance plan.

**Quality Assurance**

Quality assurance is defined as:

”All the planned and systematic activities implemented within the quality system, and demonstrated as needed, to provide adequate confidence that an entity will fulfill requirements for quality.” [ISO 8402]

The key words in this definition are that the actions taken are "planned and systematic" and that these actions “provide adequate confidence” that the desired level of quality will be achieved. These actions include those operational techniques and activities used to fulfill requirements for quality. Quality control forms an essential part of the measures taken to achieve quality assurance.

ISO 9000:2000 defines quality assurance as:

“Part of quality management focused on providing confidence that quality requirements will be fulfilled.”

The “confidence” comes from knowing that all necessary preventative measures have been taken before the process is started.

The major activities of quality assurance are:

- Establishing a quality management system and processes for requirements engineering and providing appropriate training
- Performing audits of requirements engineering and following up with corrective actions
- Performing measurements and analysis of requirements engineering processes
- Establishing and running process improvement programs to improve requirements engineering processes

**Quality Management System**

Quality management systems, based on the concept of prevention, provide confidence to both suppliers and customers that defined requirements will be met. This implies that having a management system, processes, procedures and tools together with skilled and properly trained personnel is a very important part of assuring quality in requirements engineering.

A quality management system is the organizational structure, procedures, processes and resources needed to implement quality management.

The following are elements of a quality management system:

- Organizational structure
- Responsibilities
- Methods
- Data management
- Processes
- Resources
- Customer satisfaction
- Continuous improvement
- Product quality

Requirements engineering contributes to most of the elements of a quality management system as follows:

- Responsibilities: listing the role and responsibilities of requirements engineers, systems analysts, business analysts

- Methods: methods and techniques of requirements analysis, design, modeling and documentation
- Data management: requirements documentation management
- Processes: processes of requirements analysis, design, modeling and documentation, requirements quality assurance
- Resources: tools and human resources related to requirements engineering
- Customer satisfaction: quality of requirements engineering as a key factor for ensuring customer satisfaction by meeting the needs and expectations regarding the software
- Product quality: a direct result of the fact that the quality of a product is determined by the quality of requirements

**ISO 9000**

The ISO 9000 family of standards relates to quality management systems and is designed to help organizations ensure they meet the needs of customers and other stakeholders.

ISO 9000 deals with the fundamentals of quality management, including the eight management principles on which the family of standards is based.

ISO 9001 deals with the requirements with which organizations wishing to meet the standard must comply.

**ISO 9001:2008 Quality Management Systems — Requirements**

The standard specifies requirements for a quality management system where an organization:

- needs to demonstrate its ability to consistently provide products that meet customer requirements, and
- aims to enhance customer satisfaction through the effective application of the quality management system.

These requirements are structured into the following five areas:

- Quality management systems: including general requirements and requirements for documentation
- Management responsibilities: defining requirements for management commitment, review, customer focus, policy, planning, responsibilities, authorities, and information
- Resource management**:** including requirements for provision of resources, human resources, infrastructure and work environment
- Product realization: including requirements for processes for planning, customer relations, design and development, purchasing, product and service provisioning and control of monitoring and measuring devices
- Measurement, analysis and improvement: including requirements for monitoring and measurement, control of non-conforming products, analysis of data and improvement

The key documents required by ISO 9001:2008 are a quality policy and quality manual.

Requirements engineering usually contributes to:

- Quality policy (statements related to satisfying the customer's needs and expectations)
- Section 7: Product Realization (requirements engineering's practices used in the development process, quality attributes, etc.)

- Section 8: Measurement, analysis and improvement (measuring requirements engineering processes and products)

Specifically, the following sections are closely related to requirements engineering:

7.2.1 Determination of requirement related to the product
7.2.2 Review of requirement related to product
7.3.2 Design and development inputs

**TickIT and TickITplus**

TickIT is a certification program for quality management in software development. The TickIT guide provides information to help understand and apply ISO 9001 in the IT industry. It also includes detailed information on how to implement a quality management system and its expected structure and content relevant to software activities. The TickIT guide also helps to define appropriate measures and metrics.

The original TickIT scheme has been updated to become TickITplus.

TickITplus is both an improvement tool and an ISO 9001 accredited IT certification scheme. TickITplus extends the existing TickIT scheme by combining industry best practices with international IT standards. It uses the following standards:

- ISO 9001:2008
- ISO/IEC 15504
- ISO/IEC 12207

Amongst others, TickITplus provides a defined process model covering the whole range of IT activities, not just software development. There is also a revised qualification and training structure for both assessors and practitioners.

TickITplus is built around the ISO/IEC 15504 standard – IT Process Assessment. There are five levels of the assessment:

- Foundation – This is the normal entry level and requires a process model to be defined and verified, but there is no direct process assessment.
- Bronze – This equates to level 2 (the Managed level in ISO/IEC 15504), and ensures the processes are operated with planned, monitored and adjusted management.
- Silver – This equates to level 3 (the Established level in ISO/IEC 15504), and ensures that processes are capable of achieving their outcomes in terms of definition and deployment.
- Gold – This equates to level 4 (the Predictable level in ISO/IEC 15504), ensuring that processes operate within predicted parameters.
- Platinum – This equates to level 5 (the Optimizing level in ISO/IEC 15504), and ensures that quantified measures and improvements are applied to key processes.

*[For trainings companies: give examples of non-compliance issues concerning requirements management and identify and explain the corrective actions].*

## 5.2 Maturity Models and Requirements Management

### 5.2.1 Background

Requirements engineering may not only be measured in terms of the outputs (such as requirements) but in terms of the process itself. The maturity of the process may be measured and evaluated. The maturity may be measured using the following criteria:

- The organization of the requirements engineering process and its place in the software development process
- Compliance with relevant standards
- The number and type of requirements engineering documents that have been adjusted to the organization's needs
- Usage of tools supporting the requirements engineering process
- The experience and competence of the personnel (e.g., number of certifications)
- Ability to meet deadlines and other planning-related criteria

The quality of requirements engineering can be measured using the following metrics:

- The number of issues found during specification reviews (by priority, severity)
- The number of issues found in requirements during implementation or testing (by priority, severity)
- The cost of fixing issues found in requirements in every project phase

### 5.2.2 General Maturity Model

When a general maturity model is used to describe requirements engineering, three levels are commonly used:

Level 1. Initial Level: Problems with requirements are normal

- A process for requirements engineering is missing; there are often problems with unsatisfied customers and unresolved requirements, the process does not use methods and is depending on individuals.

Level 2. Repeatable Level: Requirements specifications provide better value

- Standards for requirements documents and requirements expressions exist as well as procedures for requirements management; there are some tools and techniques in use.

Level 3. Defined Level: Requirements engineering process is established

- There is a process model that is built on experience and established methods. Any improvement work is based on independent evaluations and new methods and tools.

### 5.2.3 CMMI and Requirements Engineering

Capability Maturity Model Integration (CMMI) is a process improvement approach that provides organizations with the essential elements for effective process improvement. CMMI is a trademark owned by Software Engineering Institute of Carnegie Mellon University.

CMMI helps "integrate traditionally separate organizational functions, set process improvement goals and priorities, provide guidance for quality processes, and provide a point of reference for appraising current processes." [SEI]

CMMI best practices are published in a form of “models” each of which addresses a different area of interest. CMMI version 1.3 describes the following models:

- CMMI for Development (CMMI-DEV). It addresses product and service development processes.
- CMMI for Acquisition (CMMI-ACQ). It addresses supply chain management, acquisition, and outsourcing processes in government and industry.
- CMMI for Services (CMMI-SVC). It addresses guidance for delivering services within an organization and to external customers.

**Maturity Levels**

CMMI has five maturity levels. Maturity level ratings are awarded for levels 2 through 5.

- Maturity Level 1 - Initial
- Maturity Level 2 - Managed (REQM - Requirements Management)
- Maturity Level 3 - Defined (RD - Requirements Development)
- Maturity Level 4 - Quantitatively Managed
- Maturity Level 5 - Optimizing

The following areas of CMMI are related to requirements engineering:

- Requirements Development (RD) – the purpose of RD is to produce and analyze customer, product, and product component requirements. Specific practices by goal are:
  - SG 1 Develop Customer Requirements
    - SP 1.1 Elicit Needs
    - SP 1.2 Develop the Customer Requirements
  - SG 2 Develop Product Requirements
    - SP 2.1 Interface Requirements
  - SG 3 Analyze and Validate Requirements
    - SP 3.1 Establish Operational Concepts and Scenarios
    - SP 3.2 Establish a Definition of Required Functionality
    - SP 3.3 Analyze Requirements
    - SP 3.4 Analyze Requirements to Achieve Balance
    - SP 3.5 Validate Requirements
- Requirements Management (REQM). The purpose of REQM is to manage the requirements of the project's products and product components and to identify inconsistencies between those requirements and the project's plans and work products. Specific practices by goal are:
  - SG 1 Manage Requirements
    - SP 1.1 Obtain an Understanding of Requirements
    - SP 1.2 Obtain Commitment to Requirements
    - SP 1.3 Manage Requirements Changes
    - SP 1.4 Maintain Bidirectional Traceability of Requirements
    - SP 1.5 Identify Inconsistencies Between Project Work and Requirements

### 5.2.4 Improvement of Requirements Engineering

Increasing the quality characteristics of specified processes is the goal of process improvement. Good requirements engineering results in improving the overall production process by:

- Speeding up the implementation effort by eliminating time wasted for clarifying and explaining requirements
- Reducing the number of defects caused by errors in the requirements specification and/or misunderstanding the requirements documentation due to its low quality
- Increasing the quality of testing by providing well described, precise and testable requirements
- Making the acceptance testing easier and more reliable by using clear requirements for the basis of the testing

Therefore the role of requirements engineering in improving the development process is very important. To improve the requirements engineering itself, the following actions can be taken:

- Applying standards and good practices related to requirements engineering
- Educating personnel about the meaning and role of requirements engineering
- Designing and introducing the organization's approach to requirements engineering
- Selecting requirements engineering tools that are well matched to the organization's and project's needs
- Introducing quality control (e.g., audits, reviews) into requirements engineering activities
- Sharing knowledge between requirements engineering professionals within an organization (e.g., internal training, workshops, lessons learned and experience exchange, common knowledge base)
- Using lessons learned to improve the future projects

Improvement can apply to each of the requirements engineering activities:

- Elicitation of Requirements – This activity may be improved to collect customers' requirements in a more effective way. In order to do so, the organization may introduce some techniques to ensure the requirements are gathered more quickly, are complete and are agreed upon by most stakeholders. These techniques include interviews, brainstorming, initial prototyping, using personas, scenarios, etc.
- Requirements Analysis – Identified requirements have to be analyzed and detailed. Improving the process of requirements analysis ensures that the risk of overlooking important aspects of the requirements is lowered. Requirements analysis should involve the customer representative (to verify the work products), development team (to check the technical feasibility of detailed requirements and provide early feedback) and the quality assurance team (to ensure all requirements are testable).
- Specification of Requirements – To improve requirements specification the organization may introduce reviews, checklists and apply specific requirements engineering standard(s). A common practice is releasing the draft specification for internal technical (development) and quality assurance review. In this way most of the mistakes and gaps can be captured and corrected in the next version of the specification.
- Tracking and Controlling Requirements – To improve the quality control, it is necessary to collect measurements and note deficiencies, make corrections and train people, and also to analyze trends and other business results in order to improve the way of working.
- Quality Assurance – To implement and improve the processes of the quality management system, organizations often apply models such as TickITplus, Spice and CMMI to assist in the improvement work.

The processes in the different improvement models that are related to requirements engineering are:

ISO 15504

- ACQ1 Acquisition preparation BP1-3
- ENG1 Requirements elicitation BP1-6
- ENG2 System Requirements Analysis BP1-6
- ENG4 Software Requirements Analysis BP1-6

CMMI

- 2 Requirements Management
- 3 Requirements Development

TickITplus

- AGR 1 Acquisition and Contract Management
- TEC10 Stakeholder Requirement Definition
- TEC11 Requirements Analysis

### 5.2.5 TickITplus

Improvements are an integral part of TickITplus. The way the improvements are monitored and controlled within the scheme depends on the different maturity grades. For example, at the Foundation Level a defined improvements plan is a requirement; at higher grades the contents of this plan form a planned activity within assessments. At the Platinum Level additional maturity processes dealing with quantitative analysis and improvements are defined

ISO 15504-2 requires a process reference model to exist in order for a capability assessment to be completed. However, it would not be possible to define a single process reference model that could satisfy all potential organizational types. For this reason, TickITplus has created a Base Process Library (BPL) from which organizations select the most applicable processes to create an organizational specific process reference model suited to their specific activities. The processes in the BPL that are part of requirements engineering are TEC.10 and TEC.11.

**TEC.10 Stakeholder Requirement Definition**

To define the requirements of the products and services expected by the customer the following base practices are expected:

TEC.10.BP.1 Engage Requirements Stakeholders
TEC.10.BP.2 Develop Stakeholder Requirements
TEC.10.BP.3 Validate Stakeholder Requirements
TEC.10.BP.4 Manage Changes to Stakeholder Requirements

**TEC.11 Requirements Analysis**

The customer needs and other stakeholder requirements are analyzed and interpreted into structured system requirements. The organization considers its own product development strategy in light of stakeholder requirements. System requirements are reviewed and maintained under configuration management.

The following base practices apply:

TEC.11.BP.2 Estimate System Requirements Size
TEC.11.BP.3 Manage System Requirements
TEC.11.BP.4 Manage Changes to Requirements

### 5.2.6 Requirements Management Maturity Model

The Requirements Management Maturity Model deals with the increasing levels of formality and sophistication in eliciting and managing requirements.

The model comes in five levels [Rational Software].

1. Chaos
2. Written requirements
3. Organized
4. Structured
5. Integrated

*[For trainings companies: give an example of a requirements engineering process and explain how it can be improved by using the requirements engineering process improvement models].*

*[For trainings companies: give an example of a requirements engineering process and explain how to identify ideas for improvements].*

# 6 Requirements Management in Practice (210 mins)

## Terms

Agile model, commercial off-the-shelf, critical systems, customer-driven product, enhancement project, green-field project, in-house product, market-driven product, outsourced product, product line, replacement project, V-model

## Learning Objectives for Requirements Management in Practice

### 6.1 Product Lifecycle Management and Requirements Management (105 mins)

| | |
|---|---|
| RM-6.1.1 | Explain the differences between V-model and Agile models in terms of requirements management (K2) |
| RM-6.1.2 | Explain how to manage requirements in a V-model process (K2) |
| RM-6.1.3 | Explain how to manage requirements in an Agile model process (K2) |
| RM-6.1.4 | Explain the difference between new product, system integration, product maintenance and a commercial off-the-shelf product in terms of requirements management (K2) |
| RM-6.1.5 | Explain with examples why requirements management is important in product lifecycle management (K2) |
| RM-6.1.6 | Analyze a specific product lifecycle and determine how to apply requirements management to it (K4) |

### 6.2 Requirements Management in the Business Operational Context (105 mins)

| | |
|---|---|
| RM-6.2.1 | Explain the differences between market-driven products and custom products (K2) |
| RM-6.2.2 | Apply requirements management in a market-driven product context (K3) |
| RM-6.2.3 | Apply requirements management in a custom product context (K3) |
| RM-6.2.4 | Explain the concept of product lines (K2) |
| RM-6.2.5 | Apply requirements management in a context of product lines (K3) |
| RM-6.2.6 | Explain why configuration and release management are key issues in a product line context (K2) |

## 6.1 Product Lifecycle Management and Requirements Management

### 6.1.1 Background

A process model gives a standard format for planning, organizing and running a project.

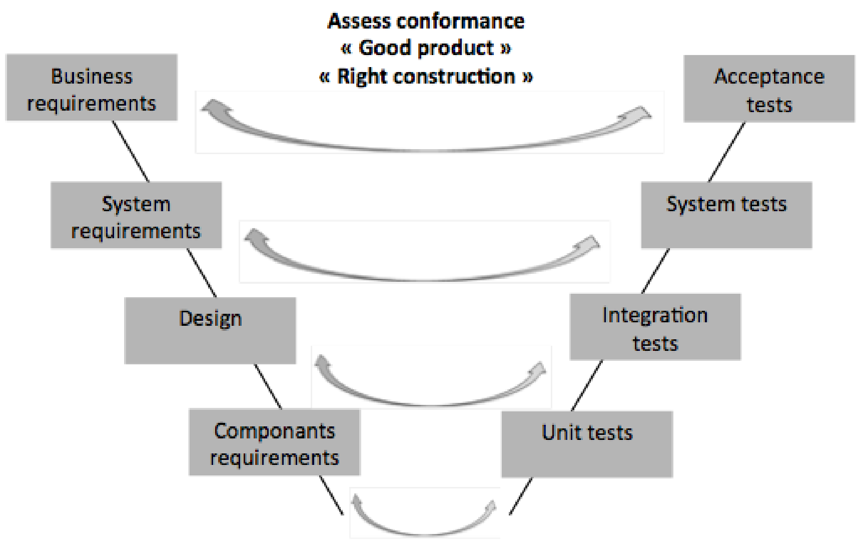
There are traditional process models proposing various approaches to developing products. These include the waterfall model, the V-model, the Rational Unified Process (RUP), etc. The main drawback is that some of these models have a more complex method for controlling requirements changes.

A requirements manager should know how to manage requirements within different process models, how to tailor the requirements management process and how to adapt.

### 6.1.2 Difference between V-Model and Agile Model

**V-Model**

The V-model was developed as an extension of the waterfall model in the late 1980s. It has a V-shape and each development phase has an associated testing phase as shown in this diagram.



Its major disadvantage is that it is difficult to change requirements after the specification phase is completed.

The V-model has advantages when used in the following situations:

- The project is long term (e.g., in the automotive or aeronautics industry)
- The requirements are clear and unlikely to change
- The software development organization has experience with similar projects
- It is important to complete each phase before starting the next one (e.g., to have clearly and precisely defined requirements before moving to the design specification step)

In the V-model, the different levels of requirements are addressed one after the other. Business requirements development must be finished and the requirements approved before beginning the work on system requirements. This is the same for the component level requirements. By the time the project reaches the implementation phases, the requirements should be stable. If the requirements are not stable and significant modifications are needed, it indicates there is a problem in the quality of the requirements.

Since there is an early involvement of testing, defects can be detected earlier and testers can develop a better understanding of the product. The early involvement of testing is a concept that is used in the iterative and Agile approaches as well.

*[For trainings companies: describe how requirements engineering works within the V-model].*

**Agile Models**

In contrast with the traditional "heavyweight" models (e.g., waterfall and V-model), there are "lightweight" models, which support changing requirements. These include Scrum, Crystal Clear, and Extreme Programming (XP). The Agile models developed out of the Agile Manifesto which was written in 2001. There are iterative as well as incremental models (e.g., the product is designed, developed and tested in small portions at a time).

The Agile [Manifesto] promotes:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

"Welcome changing requirements, even late in development" is one of the 12 principles behind the Agile Manifesto.

**Scrum**

Scrum was defined in 1995 as an Agile methodology. It has predefined roles (Scrum Master, Product Owner, Team) and practices timeboxed periods of development called sprints. It uses a prioritized list of high level requirements, called the product backlog, and a list of tasks to address during a sprint, called the sprint backlog [Scrum].

*[For trainings companies: describe how Scrum works].*

### 6.1.3 Requirements Management in the V-Model

Depending on the product context and the involvement of the stakeholders, requirements are created by using different elicitation techniques. Requirements are recorded in a requirements document as part of the requirements development process. The correct stakeholders must approve the requirements document (e.g., the customer) before continuing to the next stage.

*[For trainings companies: give examples on how requirements are managed in a V-model, from business requirements up to the release of the product].*

### 6.1.4 Requirements Management in the Agile Model

Requirements are created by questioning customer representatives. Their expectations are captured into user stories. User stories are used with Agile software development as the basis for defining the functions a business system must provide, and to facilitate requirements management. It captures the "who", "what" and "why" of a requirement in a simple, concise way, often limited in detail by what can be handwritten on a small paper notecard.

The usual form of a user story is: ***As a <role> I want to <action> so that <benefit expected>***

User stories are written by or for the business user as that user's primary way to influence the functionality of the system being developed. Developers and other stakeholders may also write user stories to express non-functional requirements (security, performance, usability, etc.).

A typical scenario for requirements engineering in an Agile context is the following:

- The development teams meet the other stakeholders, including customer representatives.
- The customer representatives formulate the user stories.
- The development teams ask questions about the features described in the user stories (e.g., the display of products on an e-commerce website).
- The customer representatives and the other stakeholders prioritize the user stories.
- The development teams analyze in detail a subset of high priority user stories, estimate them, plan the next iteration, then start to implement those stories. They may ask the customer representative for clarification at any time during this process.

The customer and the supplier must be aware of the volatility of the requirements and must take care to manage the requirements. The scenario described above is used for one iteration. For the next iteration, new requirements are analyzed and may be changed by the customer representatives. The developers must use a repository to store all requirements and to record all requirements modifications (creation, deletion, change). In this way, the repository can also be used during the maintenance phase.

With Scrum, changing requirements are managed the following way:

- Requirements, including their changes, are managed through the product backlog. For each new iteration, a subset of requirements is selected to be implemented in the next iteration and becomes part of the iteration backlog (the sprint backlog).
- The product backlog can change across the iterations. The sprint backlog cannot change during the iteration, with the following exception: important changes can be introduced only when the pre-defined sprint goal (the objective of the iteration) does not change and still can be reached in the iteration timebox. No changes are allowed in the sprint backlog if the sprint goal would be changed as a consequence of those changes, or if the sprint duration would be extended. If strong business reasons require changing the iteration with impact on the sprint goal and/or sprint duration, the iteration must be stopped and aborted and planning must start for a new iteration.

Scrum is one of the methods for planning and controlling the Agile type of development and is an alternative to traditional project management methods.

*[For trainings companies: give examples of how requirements are managed in Scrum, iteration after iteration up to the release of the product].*

### 6.1.5 Requirements Management in the Product Lifecycle

There are different product contexts and project lifecycles:

- New development
- Maintenance
    - Evolution
    - Correction
    - Adaptation
- Replacement, including commercial off-the-shelf options

**New Product or Green-field Project**

New development of a product is sometimes called a green-field project. In a green-field project, a product is built from scratch to address new end user problems and explore new opportunities. The following requirements management activities are used:

- Planning the management of all levels of requirements
- Defining the structure of the requirements repository
- Defining the rules to set up traceability between requirements and other artifacts

**Product Maintenance**

Product maintenance, sometimes executed as an enhancement project, can comprise evolution, correction, or adaptation of the product. Change management activities must be used to analyze the requested modifications. Traceability and impact analysis activities are used to estimate the cost of the changes. When the modifications (creation, change, deletion) of the requirements are complete, a new baseline of the requirements repository must be set.

One of the major problems with product maintenance is that, very often, requirements modifications are managed by delta meaning that only the change to the original requirement is noted, the original requirement is not re-written to include the change. This is done to reduce the costs of the modifications. Unfortunately, after several releases, it is very difficult to record all the changes together into the requirements repository. Requirements management for product maintenance must be focused on tracking all changes and updating all work products impacted by the modifications.

**Product Replacement**

A replacement of a part of a product consists of changing an existing application to a new custom-built one, a commercial off-the-self application or a mix of both. At the beginning of the replacement project, specific requirements dedicated to the scope of the replacement must be written and approved. Once the scope is set, requirements engineering drives the selection of the replacement application, with respect to the needs and constraints of the customer.

A common issue in product maintenance or product replacement is that the developers who implemented the product originally may have left the project and even the company. The scope of requirements management is not isolated to the development phase; requirements continue to live and to be modified during the maintenance stage up to the decommissioning (end of life) of the product. Requirements management must be involved each time a change request is raised for the product. Requirements management is essential to maintaining the integrity and the consistency of the requirements repository throughout the lifecycle of the product. In this way, even if the development staff changes, there is a consistent view of the state of the product and new developers can easily understand the implementation.

New development and delivery concepts such as continuous delivery or devops need to have an efficient requirements management process because of the necessary coordination between the requirements repository and product being deployed into operation.

*[For trainings companies: give one or more examples about how to introduce requirements management in different product lifecycle phases].*

## 6.2 Requirements Management in the Business Operational Context

### 6.2.1 Background

There are different business contexts of products for which the management of requirements must be adapted:

- Customer-driven product
- Market-driven product
- Critical product
- Product line
- In-house product
- Outsourced product

### 6.2.2 Market-driven Product and Customer-driven Product

Market-driven products are designed for a broad audience. They usually contain functionality requested and used by most of the potential users. The scope of functionality is usually determined by market research and feedback from the users.

Sometimes such products may be parameterized to meet the needs of specific user groups. Further modifications may require special patches, applications or code changes (which are not always possible).

Customer-driven products are created for a concrete customer and are designed to satisfy that customer's specific needs, expectations and requirements. Such solutions are designed, for example, to support the customer's business processes, to work with specific IT infrastructures and/or technology. Customer-driven solutions often are perceived as more expensive than off-the-shelf software, but that may not be true in all situations. Such systems are usually designed for unique businesses, technology or when the customer's needs vary from the requirements of common users, necessitating a dedicated solution.

#### Similarities

When requirements engineering processes are established for both market-driven and customer-driven products, the entire lifecycle is considered. Processes are established for assigning and maintaining the requirements across different releases.

Selected tools and methods support the complete lifecycle, taking into account the competencies of the different organizations that will perform the maintenance and further development. These tools and methods should include the procedures for requirements management.

**Common Problems**

The common problem is that products are often released to production when the development of the solution is completed but the necessary infrastructure is not yet developed. In this case the product would need to be upgraded when there are more customers/users operating in production and that often includes managing different requirements collections for different customers and for different releases.

Another difficulty is finding a good process for handling requirements in product management. This may be caused by the fact that the customer is often represented by a marketing function and there is a need to have business case models as well as clear road maps to assist in the requirements analysis and prioritization steps.

When developing a solution there is often a need to involve a project change control board (CCB) for controlling requirements and changes. In some cases two layers of CCB are needed, one for the product (often called the product board or product CCB) and one for the different release projects.

*[For trainings companies: give examples of market-driven and customer-driven contexts and describe how to manage requirements in these contexts.]*

### 6.2.3 In-house Product and Outsourced Product

For an in-house product, the same company performs all the activities of developing the product, from the product definition through to the deployment into the operational environment. In this case, the different stakeholders within the company manage the requirements.

For an outsourced product, the development usually is performed by a subcontractor. In this case, the contractor is selected by assessing the responses to a request for proposals. The customer must take care to clearly elaborate the request for proposal. The best practice is to present clear, well-documented business requirements in the request for proposal.

### 6.2.4 Product lines

[Clements] defines a software product line as:

"A software product line is a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way".

In a product line project, a product family is developed to cover multiple variants. Each variant is a customization of a product to meet the needs of a class of end users or to manage and control a subset of features. A product line is defined as a set of commonalties and a set of variants. Requirements management activities, especially configuration management, must be used to define and manage the different product releases that are instantiated from the product line.

*[For trainings companies: give examples of products lines contexts and describe how to manage requirements in these contexts.]*

**Requirements Configuration and Release Management for Product Lines**

During the development of the requirements of a product line, commonality and variability must be addressed; this is often expressed by a features diagram showing dependencies and exclusions between the different functionalities. The definition of a product belonging to the product line consists of choosing from a list of features. This set of requirements must be stored as a "configuration" and linked to the release of the product. All the products based on the same features diagram must be managed in the same way, by defining a product configuration and setting the corresponding release.

*[For trainings companies: give an example of a features diagram and show how configurations and releases of a product are managed.]*

### 6.2.5 Critical Systems

Critical systems are systems that have strong requirements in term of reliability, availability, maintainability, safety and/or security. The impact of defects in these systems is very high and often linked to protection of a human being. Requirements engineering must ensure that all the stakeholders have enough confidence in the system (quality assurance) before it is deployed in the operational environment. The specific non-functional requirements, above mentioned, must be carefully managed to prove that they are correctly addressed by the implemented system.

In term of requirements management, these non-functional requirements (and all the requirements of the critical system) must be traced to and from the other artifacts. For example, a best practice could be to trace requirements to risks that are identified during the assessment of safety or security risks. Another best practice is to link requirements to the specific tests (robustness, load testing, penetration testing, static analysis, etc.) to verify and validate that the system behavior conforms to the defined non-functional requirements.

*[For trainings companies: give examples of critical systems contexts and describe how to manage requirements in these contexts.]*

# 7 Tools for Requirements Management (100 mins)

## Terms

requirements management tool

## Learning Objectives for Tools in Requirements Management

### 7.1 Reasons to Use a Requirements Management Tool (15 mins)

| | |
|---|---|
| RM-7.1.1 | Explain why tools are necessary for requirements management (K2) |

### 7.2 Use of a Requirements Management Tool (65 mins)

| | |
|---|---|
| RM-7.2.1 | Explain which factors are main drivers for the tool selection (K2) |
| RM-7.2.2 | Analyze an organizational context to suggest the proper tool to manage requirements (K4) |
| RM-7.2.3 | Describe goals and advantages of possible integration between requirements management tools and tools supporting other areas in a project (K3) |

### 7.3 Practical Examples for the Use of Tools (20 mins)

| | |
|---|---|
| RM-7.3.1 | Select an adequate tool for managing requirements in a given scenario (K3) |

## 7.1.1 Reasons to Use a Requirements Management ToolBackground

Requirements management activities process a large amount of information (requirements, attributes of requirements, traceability, versions and baselines). All these artifacts must be tracked and presented to the different stakeholders involved in a product lifecycle. Moreover, each change request and modification of the requirements must be kept and traced back to the origin of the request and other artifacts. Considering this large amount of data and frequency of updates, manual management is not feasible.

The different activities of requirements management can benefit from the use of efficient tools. Tools for storage and administration of requirements facilitate requirements engineering by performing prescribed activities and ensuring proper overview. With good tools, it is possible to keep complex static documents consistent and current. It is important to select the tool before the product is developed in order to maximize the usage and the benefit from the tool.

Tools may provide any of the following:

- A database to store and manage requirements, maintain traceability, track change requests and record change history
- Functionality to manage the different attributes of requirements (e.g., the status as a requirement moves through the lifecycle)

- Functionality to manage the state of a set of requirements and the associated metrics (e.g., metrics assessment and reporting)
- Functionality to set and manage traceability between requirements (at the same level and at different levels) and between requirements and other artifacts (to source code and test)
- Functionality to manage impact analysis and show the real cost of change requests and modifications
- Functionality to manage the versioning of requirements and to baseline a set of requirements
- Functionality to show the requirements and attributes to the given stakeholders via reporting

The advantages of using tools may be any of the following:

- Ensuring that all requirements are stored in one place and accessible to all involved stakeholders
- Supporting requirements traceability (of test cases, etc.) and allowing verification of the relevant requirements coverage
- Managing requirements changes in an easy way

## 7.2 Use of a Requirements Management Tool

### 7.2.1 Background

Using requirements management tools has become necessary due to the growing complexity and size of software systems, along with the increasing number of requirements and their dependencies.

Different types of tools may be used on a specific project. Selecting the right tool depends on the following aspects:

- The size of the project (the number of requirements)
- The type of the project (development of completely new software, extending existing software, providing new functionality and interfaces, etc.)
- The complexity of the requirements
- The development approach (different tools may be better suited for use with a V-model or an Agile model)
- The customer's requirements regarding the tools
- The requirements for traceability (between which artifacts)
- The goals for reuse of requirements (e.g., the use of dedicated requirements repositories)
- The product and project risks
- The level of criticality of the product
- The knowledge and experience of the vendor's team and the customer (e.g., familiarity with a technique/notation used by a tool)

### 7.2.2 Integration of a Requirements Management Tool with Other Tools

A requirements management tool is used in the following ways:

- To store requirements in a repository

- To provide different views of requirements to different stakeholders
- To track the status of the requirements
- To manage the versions of the requirements and the baselines of the requirements repository
- To manage the change requests for the requirements
- To manage the traceability between the requirements themselves and between the requirements and the other work products / artifacts of the project
- To calculate metrics for the requirements

To ensure all of these features, a requirements management tool needs to be integrated with other tools. These interfacing tools will vary depending on the environment. The following is a sample list:

- A development requirements tool to link the implemented requirements together
- A design development tool to allocate requirements to design components
- A source code development tool to allocate requirements to source code
- A test management tool to trace requirements to tests
- A configuration management tool to manage globally the version of requirements and the baselines of the requirements repository
- A change request tracker to manage globally the modification requests

The integration of a requirements management tool with tools supporting other engineering areas in a project can help to provide a global application lifecycle management view.

*[For trainings companies: give an overview of a tools platform showing the integration of a requirements management tool with tools supporting other processes.]*

*[For trainings companies: give an example of an organizational context and identify the proper tool to manage requirements.]*

## 7.3 Practical Examples for the Use of Tools

*[For trainings companies: give examples and demonstrate the practical use of tools.]*

# 8 References

## 8.1 Standards

IEEE Standard 610.12-1990 IEEE Standard Glossary of Software Engineering Terminology

IEEE Standard 829-1998 IEEE Standard for Software Test Documentation

IEEE Standard 830-1998 IEEE Recommended Practice for Software Requirements Specifications

IEEE Standard 1012-2004: IEEE Standard for Software Verification and Validation

IEEE Standard 1059-1993: IEEE Guide for Software Verification and Validation Plans

IEEE Standard 1220-1998: IEEE Standard for Application and Management of Systems Engineering Process

IEEE Standard 1233-1998 IEEE Guide for Developing System Requirements Specifications

IEEE Standard 1362-1998 IEEE Guide for Information Technology-System Definition – Concept of Operations (ConOps) Document

ISO 8402: Quality management and quality assurance

ISO 9000: Quality management systems

ISO 12207: Systems and software engineering – Software life cycle processes

ISO 15288: Systems and software engineering – System life cycle processes

ISO 15504-x: Information technology – Process assessment

ISO 29148: Systems and software engineering – Life cycle processes – Requirements engineering

ISO 31000: Risk Management - Principles and Guidelines on Implementation

IEC 31010: Risk Management - Risk Assessment Techniques

ISO/IEC 73: Risk Management – Vocabulary

ISO/EIC 25000: Systems and software engineering – Systems and software quality requirements and evaluation

## 8.2 REQB Documents

[REQB_APPROACH] REQB® Approach to Requirements Engineering, Version 1.0

[REQB_FL_SYL] REQB® Foundation Level Syllabus, Version 2.1

[REQB_GLO] REQB® Standard glossary of terms used in Requirements Engineering, Version 1.3

## 8.3 Referenced Books and Publications

[ApproachRE] ApproachRE, Approach to requirements engineering, version 1.0

[BABOK] BABOK: http://www.iiba.org/babok-guide.aspx, version 2.0

[Bohner] Bohner, S.A. and R.S. Arnold, Eds. *Software Change Impact Analysis*. Los Alamitos, California, USA, IEEE Computer Society Press 1996.

[Clements] Clements et al.: Software Products Lines – Practices and Patterns, 3rd edition, Addison-Wesley, Boston 2001

[CMMi] https://www.sei.cmu.edu/cmmi/

[Gilb] Gilb, T.: *What's Wrong with Requirements Specification?* See: www.gilb.com

[Jarke] Jarke, M.: Requirements Tracing, Communications of the ACM, 1998

[Kilpinen] Kilpinen, M.S.: *The Emergence of Change at the Systems Engineering and Software Design Interface: An Investigation of Impact Analysis. PhD Thesis*. University of Cambridge. Cambridge, UK 2008.

[Kotonya] Kotonya et al.: Requirements Engineering, Processes and Techniques, Wiley 2002

[Larman] Larman et al.: Craig Larman, and Bas Vodde, "Practices for Scaling Lean & Agile Development: Large, Multisite, & Offshore Product Development with Large-Scale Scrum", Addison-Wesley, 2010)

[Lawrence] Lawrence, Wiegers, Ebert "The Top Risks of Requirements Engineering", IEEE Software, November/December 2001

[Manifesto] Various contributors, Manifesto for Agile Software Development, 2001, www.agilemanifesto.org

[Pfleeger06] Pfleeger, S.L. and J.M. Atlee: *Software Engineering: Theory and Practice.* Upper Saddle River, New Jersey, USA, Prentice Hall 2006.

[RATIONAL SOFTWARE] The Five Levels of Requirements Management Maturity

[SEBOK] SEBOK; http://www.sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_%28SEBoK%29, version 1.2

[SEI] http://www.sei.cmu.edu

[Schwaber] Schwaber et al.: Ken Schwaber and Mike Beedle, "Agile Software Development with Scrum," Prentice Hall, 2001.

[Scrum] http://www.scrum.org/scrumguidesTickITplus: http://www.tickitplus.org/

[Weigers05] Wiegers, K. E.: *Software Requirements*. Redmond 2005

[Weigers06] Wiegers, K. E.: *More About Software Requirements: Thorny Issues and Practical Advice*. Redmond, Washington 2006

## 8.4 Additional Reading Recommendations

The following publications provide good background material for requirements management and are recommended for anyone wishing to broaden their knowledge of the field.

Beck, K.: *Test Driven Development. By Example*. Amsterdam 2002

Beck, K.: *Refactoring*: *Improving the Design of Existing Code*. Addison-Wesley Longman 1999

Boehm, B.: *Software Engineering Economics*. Englewoods Cliffs, NJ 1981

Cockburn, A.: Agile *Software Development*. Addison Wesley 2002

Cockburn, A.: *Writing Effective Use Cases*. Amsterdam 2000

Cohn M.: *Estimating With Use Case Points*, Fall 2005 issue of Methods & Tools

Cotterell, M. and Hughes, B.: *Software Project Management*, International Thomson Publishing 1995

Davis A. M.: *Operational Prototyping: A new Development Approach*. IEEE Software, September 1992. Page 71

Davis, A. M.: Just *Enough Requirements Management. Where Software Development Meets Marketing*, Dorset House, 2005, ISBN 0932633641

DeMarco, T.: *Controlling Software Projects: Management, Measurement and Estimates*. Prentice Hall 1986

DeMarco, Tom: *The Deadline: A Novel About Project Management*. New York 1997

Dorfman, M. S.: *Introduction to Risk Management and Insurance (9 ed*.). Englewood Cliffs, N.J: Prentice Hall 2007. ISBN 0-13-224227-3.

Evans, E. J.: *Domain-Driven Design*: *Tackling Complexity in the Heart of Software*. Amsterdam 2003

[Gilb,Graham] Gilb, T.; Graham, D.: *Software Inspection*. Reading, MA 1993

Graham, D. et al: *Foundations of Software Testing*. London 2007

Heumann, J.: *The Five Levels of Requirements Management Maturity,* see: http://www.therationaledge.com/content/feb_03/f_managementMaturity_jh.jsp

Hull, E. et al: *Requirements Engineering*. Oxford 2005

Jacobson, I. et al.: *The Unified Software Development Process*. Reading 1999

Jacobson, I.et al.: *Object-Oriented Software Engineering. A Use Case Driven Approach*. Addison-Wesley 1993

Lauesen, S.: *Software Requirements: Styles and Techniques*. London 2002

Mangold, P.: IT-*Projektmanagement kompakt*. Munich 2004

McConnell, S.: *Aufwandschätzung für Softwareprojekte*. Unterschleißheim 2006

McConnell, S.: *Rapid Development: Taming Wild Software Schedules (1st ed*.). Redmond, WA: Microsoft Press. ISBN 1-55615-900-5, 1996

Newman, W.M. and Lamming, M.G.: *Interactive System Design*, Harlow: Addison-Wesley 1995

Paulk, M., et al: *The Capability Maturity Model: Guidelines for Improving the Software Process*. Reading, MA 1995

[Pfleeger01] Pfleeger, S. L.: *Software Engineering: Theory and Practice, 2nd edition*. Englewood Cliffs, NJ 2001

Pohl, K.: *Requirements Engineering. Grundlagen, Prinzipien, Techniken*. Heidelberg 2007

Project Management Institute: *A Guide to the Project Management Body of Knowledge* (PMBOK ® Guide). PMI 2004

Robertson, S., Robertson, J.: *Mastering the Requirements Process*, Harlow 1999

Rupp, C.: *Requirements-Engineering und Management. Professionelle, Iterative Anforderungsanalyse in der Praxis*. Munich 2007

Sharp H., Finkelstein A. and Galal G.: *Stakeholder Identification in the Requirements Engineering Process*, 1999

Sommerville, I.: *Requirements Engineering*. West Sussex 2004

Sommerville, I.: *Software Engineering 8*. Harlow 2007

Sommerville, I.; Sawyer, P.: *Requirements Engineering: A Good Practice Guide*. Chichester 1997

Sommerville, I.; Kotonya, G.: *Requirements Engineering: Processes and Techniques*. Chichester 1998

Spillner, A. et al: *Software Testing Foundations*. Santa Barbara, CA 2007

SWEBOK: http://www.computer.org/portal/web/swebok, version 3.0

Thayer, R. H.; Dorfman, M.: *Software Requirements Engineering, 2nd edition*. Los Alamitos, CA 1997

V-Modell® XT: http://www.vmodellxt.de/

[Wiegers99] Wiegers, K.E.: *First Things First: Prioritizing Requirements.* Software Development, September 1999

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

# 9 Index