IREB Certified Professional for Requirements Engineering
- Advanced Level Elicitation -

Syllabus

Version 2.0.0
February 14, 2019

ATTENTION: This syllabus will become effective on April 1, 2019 only!
Until March 31, 2019: Examinations are only possible on basis of syllabus version 1.0!
April 1, 2019 until June 30, 2019: Examinations based on syllabus 1.0 and on this syllabus 2.0.0 are possible!

Acknowledgements

Version 1.0 of this syllabus was created by Oliver Creighton, Dominik Häußer, Kim Lauenroth, Henriette Katharina Lingg, Thomas Mödl, Michael Richter, Chris Rupp, Dirk Schüpferling, Patrick Steiger and Malik Tayeh.

Version 2.0 is a major revision, created by Dominik Häußer, Kim Lauenroth, Hans van Loenhoud, Anja Schwarz and Patrick Steiger.

During this revision, feedback was provided by Juliane Blechinger, Nikolaos Kaintantzis, Kostas Kolovos, Michael Richter, Stefan Sturm and Roger Wouterse.

Reviews were performed by Birgit Penzenstadler (content) and Gareth Rogers (language).

Approved for release on July 11, 2018 by the IREB Council upon recommendation of Thorsten Weyer.

We thank everybody for their involvement.

Preamble

Purpose of the document

This syllabus defines the Advanced Level of the certification “Elicitation” established by the International Requirements Engineering Board (IREB). The syllabus provides training providers with the basis for creating their course materials. Students can use the syllabus to prepare themselves for the examination.

Contents of the syllabus

The module “Elicitation” of the Advanced Level addresses professionals with career profiles like *Requirements Engineering*, *business analysis*, *business engineering*, *and organizational design*, who wish to extend their knowledge and skills in the area of requirements elicitation.

Content scope

In the Advanced Level – as in the Foundation Level – Requirements Engineering principles are provided that are equally valid for any system – such as embedded systems, safety-critical systems, traditional information systems. This does not mean that the suitability of approaches for the individual areas, accounting for their particularities, cannot be dealt with in a training course. However, it is not the goal to present specific Requirements Engineering methods of a particular domain.

This syllabus is not based on any specific software development approach and associated process model that makes a statement about the planning, control and sequence of application of the addressed Requirements Engineering concepts and techniques in practice. It is not intended to particularly emphasize a specific approach, neither for Requirements Engineering nor for software engineering overall.

It defines what constitutes the knowledge of Requirements Engineers, but not the exact interfaces with other disciplines and processes of software engineering.

Level of Detail

The level of detail of this syllabus allows internationally consistent teaching and examination. To reach this goal, the syllabus contains the following:

- General educational objectives,
- Contents with a description of the educational objectives and
- References to further literature (where necessary).

Educational Objectives / Cognitive Knowledge Levels

Each module of the syllabus is assigned a cognitive level. A higher level includes the lower levels. The levels are classified as follows:

- **K1 (Remember)**: The candidate will recognize, remember and recall a term or concept - identify, remember, retrieve, recall, recognize, know
- **K2 (Understand)**: The candidate can select the reasons or explanations for statements related to the topic, and can summarize, compare, classify, categorize and give examples for the concept - summarize, generalize, abstract, classify, compare, map, contrast, exemplify, interpret, translate, represent, infer, conclude, categorize, construct models
- **K3 (Apply):** The candidate can select the correct application of a concept or technique and apply it to a given context - implement, execute, use, follow a procedure, apply a procedure

- **K4 (Analyze):** The candidate can separate information related to a procedure or technique into its constituent parts for better understanding, and can distinguish between facts and inferences. Typical application is to analyze a document, software or project situation and propose appropriate actions to solve a problem or task - analyze, organize, find coherence, integrate, outline, parse, structure, attribute, deconstruct, differentiate, discriminate, distinguish, focus, select

All terms defined in the glossary have to be known (K1), even if they are not explicitly mentioned in the educational objectives. The glossary is available for download on the IREB homepage at https://www.ireb.org/downloads/#cpre-glossary

This syllabus and the related handbook use the abbreviation "RE" for Requirements Engineering.

Structure of the Syllabus

The syllabus consists of five main chapters. Each chapter covers one educational unit (EU). Main chapter titles contain the cognitive level of their chapters, which is the highest level of their sub-chapters. Furthermore, the teaching time is suggested that is the minimum a course should invest for that chapter. Training companies are free to devote more time to the EUs and the exercises, but make sure that the proportions between the EUs are maintained. Important terms within the chapter are listed at the beginning of the chapter.

Example: EU2 Requirements sources (K3)
Duration: 2.5 hours
Terms: stakeholder, requirements source, relationship management, user, persona

This example shows that Chapter 2 contains education objectives at level K3 and two and a half hours are intended for teaching the material in this chapter.

Each chapter contains sub-chapters. Their titles also contain the cognitive level of their content.

Educational objectives (EO) are enumerated before the actual text. The numbering shows to which sub-chapter they belong.

Example: EO 2.1.1

This example shows that educational objective EO 2.1.1 is described in sub-chapter 2.1.

The Examination

This syllabus is the basis for the examination for the Elicitation Advanced Level.

A question in the examination can cover material from several chapters of the syllabus. All chapters (EU 1 to EU 5) of the syllabus can be examined.

The format of the examination is multiple-choice as well as an assessed written assignment; the details are set out in the examination regulations.

Examinations can be held immediately after a training course, but also independently from courses (e.g. in an examination center). A list of IREB licensed certification bodies can be found on the website https://www.ireb.org.

Version History

Version	Date	Comment
1.0	December 20, 2012	Initial version based on German version 1.0-2 from December 20, 2012
2.0.0	February 14, 2019	Major revision • **The name *Elicitation and Consolidation* of version 1.0 was changed into *Elicitation* (alone) to remove the ambiguous term '*consolidation*'.** • Chapter 1 (*A framework for structuring and managing requirements elicitation and conflict resolution*) is new. It focuses on the structuring and management of elicitation and conflict resolution activities. • Chapter 2 (*Requirements sources*) is updated to a more structured style. • Chapter 3 (*Elicitation Techniques*) is rewritten to give a more consistent and less detailed overview of elicitation techniques. • Chapter 4 (*Conflict Resolution*) is updated to a more structured style. Its title was changed to support a more consistent terminology. • Chapter 5 (*Skills of the Requirements Engineer*) is a structured and less detailed update from the original chapter 1 of version 1.0. The level of detail is reduced compared to version 1.0. More detailed information is published in the separate *Handbook Elicitation*. Cognitive Knowledge Levels according to the new IREB definition applied.

Contents

EU 1 A framework for structuring and managing requirements elicitation and conflict resolution (K3)

Duration: 1.5 hours

Terms: Elicitation activity, conflict resolution activity, technique, process pattern

Educational objectives:

EO 1.1.1 Understanding the scope of elicitation and conflict resolution in Requirements Engineering
EO 1.2.1 Understanding the challenges of planning elicitation and conflict resolution
EO 1.2.2 Understanding the factors relevant to the approach of planning elicitation and conflict resolution activities
EO 1.3.1 Applying the information structure for elicitation and conflict resolution activities
EO 1.3.2 Understanding the difference between short- and long-term elicitation and conflict resolution activities
EO 1.3.3 Understanding the importance of a setup phase for elicitation and conflict resolution
EO 1.3.4 Applying conscious planning and execution of elicitation and conflict resolution activities
EO 1.4.1 Understanding the importance of adjusting the elicitation and conflict resolution techniques to specific contexts
EO 1.4.2 Understanding the concept of process patterns

EU 1.1 The scope of elicitation and conflict resolution in Requirements Engineering (K2)

In accordance with the definition of Requirements Engineering as presented in [PoRu2015], the objective of requirements elicitation and conflict resolution is “knowing the relevant requirements”, “achieving a consensus among the stakeholders about these requirements” and “understanding […] the stakeholders’ desires and needs”.i

Within elicitation, it is the task of the Requirements Engineer to understand the stakeholders’ desires and needs while ensuring that the requirements from all relevant requirements sources have been collected. This includes identifying these sources, understanding the nature and importance of the different types of requirements and applying appropriate techniques to elicit them. A major point in elicitation is to turn implicit demands, wishes and expectations into explicit requirements [ISO29148].

During elicitation, conflicting requirements from different sources are often encountered. These conflicts have to be resolved, in order to create a single, consistent and agreed-on set that can serve as an input for the efficient development, maintenance and operation of an effective system.

EU 1.2 Factors relevant to the approach of planning elicitation and conflict resolution (K2)

Literature on software estimation [McCo2006] and results from industrial practice place a lot of responsibility on the discipline of Requirements Engineering for meeting the overall development objectives. From the perspective of Requirements Engineering, a significant part of this responsibility has to be placed on requirements elicitation and conflict resolution.

Both require a specific planning approach because of the following challenges:

- Requirements elicitation cannot be planned solely based on the expected size of the outcome, as no realistic expectation is available at the start of elicitation.
- Requirements conflicts cannot be planned or predicted. The Requirements Engineer has to react to the conflict as soon as it arises.

As a result, it is advisable to avoid detailed planning and instead define a coarse-grained upfront plan for requirements elicitation and conflict resolution. The planning and execution of elicitation and conflict resolution should be performed similarly to a research project. This means that the plan is iteratively revised as the activities proceed and more information becomes available.

EU 1.3 Planning and executing requirements elicitation and conflict resolution (K4)

Although elicitation and conflict resolution require a specific planning approach, its planning and execution cannot be treated in isolation from other activities in system development. For the definition of a planning framework, it is assumed that every development that includes elicitation and conflict resolution activities uses some kind of plan to structure the effort and its tasks. As the work moves on, the plan needs to be maintained and updated.

Two types of activities can be included in any kind of plan:

- *Elicitation activities*: the identification of requirements sources and the elicitation of requirements.
- *Conflict resolution activities*: the actions needed to solve requirements conflicts and to arrive at a single agreed-on set of requirements.

An elicitation activity should provide the following information: The *elicitation objective*, the desired *result quality*, the selected *source(s)* and the selected *elicitation technique.*

A conflict resolution activity should provide the following information: The *involved requirement(s)*, the involved *source(s)*, the selected *conflict resolution technique*, and the *achieved result*.

In addition to information related to elicitation and conflict resolution, both activities may provide management information related to timing and resources.

In general, three different sets of elicitation and conflict resolution activities can be distinguished:

- Set 1 – *Executed* elicitation and conflict resolution activities
- Set 2 – *Short Term* elicitation and conflict resolution activities
- Set 3 – *Long Term* elicitation and conflict resolution activities

In the course of development, the set of executed activities will grow, as short-term activities are executed. Long-term activities will be detailed and become short-term activities, or will be refined by several short-term activities, or may be abandoned altogether if they no longer make sense. It is recommended to distinguish between the *setup phase* and the *execution phase* of elicitation and conflict resolution activities.

The following guidelines for the *setup phase* are important:

- Get an overview of the project situation, business case
- Determine elicitation objectives
- Plan for the systematic analysis of the system context
- Plan for the systematic identification of (multiple types of) requirements sources
- Consider relevant process patterns to define the activities
- Allow time and budget for conflict resolutions activities

The following guidelines for the *execution phase* are important:

- Consider elicitation and conflict resolution as time-boxed activities
- Question the plan after each activity (and revise if necessary)
- Schedule defensively, making use of short and long-term activities
- Incorporate slack to leave time for creativity and unexpected events
- Parallelize independent activities
- Combine elicitation activities that address the same requirements source
- Search for conflicts and react to them according to an agreed strategy

In addition, it is good practice to add a *closure phase* that focuses on learning from the project and improving the skills of the project participants. Guidelines are covered in EU 5.

EU 1.4 Process patterns (K2)

Every project is a unique event, so no general approach exists that fits all elicitation needs. In this syllabus, the concept of *process patterns* is used to identify similarities between certain situations that can be used as a guideline for actual elicitation activities. If a single pattern does not fit, a combination or a sequence of patterns can be applied.

The concept of patterns was originally developed in an architectural context [AlIS1977]. In an elicitation context, a pattern describes a reusable method for requirements elicitation in a certain scope (e.g. domain, project situation).

The pattern contains information about the general method (phases, sequence of activities) and gives guidance for the elicitation activities, including the definition of elicitation objectives, selection of techniques, definition of the result quality, and possible requirements sources.

Patterns evolve in a specific context. We consider all patterns that potentially lead to new or enhanced requirements. They may also include other activities (e.g., testing, design, conflict resolution).

Examples of process patterns include:

- Waterfall/milestone-driven development
- Lean software development
- Agile software development
- Human-centered design
- Design thinking
- Embedded systems development
- System maintenance

The Requirements Engineer should actively search for patterns that are relevant for his or her own situation. Keep in mind that:

- Process patterns are good practices from literature and practical work, providing overviews that can be used as a starting point for defining elicitation activities in a comparable situation.
- Typically, the information provided is not sufficient for an immediate execution of the process. Analysis of similarities and differences between the pattern scope and the actual situation helps to identify a proper approach and to select useful techniques.
- The above-mentioned list of patterns is neither complete nor exhaustive. Furthermore, patterns can, and often should, be combined in various ways.
- Experienced Requirements Engineers are encouraged to develop and share their own patterns.

EU 2 Requirements sources (K3)

Duration: 2.5 hours

Terms: stakeholder, requirements source, relationship management, user, persona

Educational objectives:

EO 2.1.1 Understanding the importance of systematic and pragmatic identification of requirements sources in the system context
EO 2.2.1.1 Understanding typical stakeholder groups
EO 2.2.1.2 Applying the systematic identification and selection of stakeholders
EO 2.2.2.1 Applying stakeholder relationship management for preventing and resolving problems with stakeholders
EO 2.2.3.1 Applying a documentation schema for the stakeholders involved
EO 2.2.4.1 Understanding the significance of the user as a stakeholder
EO 2.2.4.2 Applying personas
EO 2.3.1.1 Understanding typical candidate documents
EO 2.3.1.2 Applying the systematic identification and selection of documents
EO 2.3.2 Applying a documentation schema for the documents considered
EO 2.4.1.1 Understanding typical types of systems
EO 2.4.1.2 Applying the systematic identification and selection of systems
EO 2.4.2 Applying a documentation schema for the systems considered

EU 2.1 Fundamentals of requirements sources (K3)

The quality and completeness of requirements depend greatly on the requirements sources involved. Missing a relevant source will lead to an incomplete understanding of the requirements. During development, the Requirements Engineer has to identify and involve all relevant requirements sources. As explained in the CPRE Foundation Level syllabus [FreA2017], the three most important types of requirements sources are stakeholders, documents and systems. Identification of requirements sources is an iterative and recursive process [ISO29148] and requires constant reconsideration.

The Requirements Engineer can choose from two different approaches towards the identification of requirements sources:

- *Pragmatic* identification: The Requirements Engineer uses his or her current knowledge and experience of the project and its context (e.g. domain knowledge) to name relevant stakeholders, documents and systems.
- *Systematic* identification: The Requirements Engineer applies a specific strategy to identify possible requirements sources by defining specific elicitation activities that focus on the identification of requirements sources.

Pragmatic and systematic identification complement each other and bear risks if used on their own. It is highly recommended to use a mixture of both to identify requirements sources in an efficient and effective way.

EU 2.2 Identify, classify, manage stakeholders (K3)

In the CPRE glossary [Glin2017], a stakeholder is defined as "a person or organization that has a (direct or indirect) influence on a system's requirements. Indirect influence also includes situations where a person or organization is impacted by the system."

EU 2.2.1 Identifying and selecting stakeholders as requirements sources (K3)

The Requirements Engineer has to identify all relevant stakeholders for the development effort.

A non-exhaustive list of stakeholder roles includes:

- Direct system users
- Business / process managers
- Clients and individual customers, customer-representing organizations
- Opponents and competitors
- IT staff
- Governmental and regulatory institutions

Potential sources for relevant stakeholder roles are:

- Checklists of typical stakeholder groups and roles (see above)
- Organization structures (e.g. organization charts of the company that will use the system to be built)
- Business process documentation (e.g. business processes to be supported by the system to be developed)
- Stakeholder categorization schemata (e.g. Alexander's onion model [Alex2009] or the Robertsons' generic stakeholder map [RoRo2013])

When *pragmatically identifying* stakeholders, Requirements Engineers use their current knowledge and experience of the context (e.g. domain) to name relevant stakeholder roles and their representatives (the stakeholders).

During *systematic stakeholder identification*, the Requirements Engineer defines elicitation objectives with a dedicated focus on the identification of stakeholders. Two different types of elicitation objectives should be considered:

- *Information-focused*: finding individual stakeholders required for certain information
- *Stakeholder-focused*: finding individual stakeholders representing certain stakeholder roles

Initially identified stakeholders are useful sources for identifying additional ones.

EU 2.2.2 Stakeholder relationship management (K3)

Problems with stakeholders typically arise if the rights and obligations of a stakeholder, in respect to the proposed system or the current project, are not clear or if the stakeholder's needs are not sufficiently addressed. Stakeholder relationship management is an effective way to counter problems with stakeholders.

[Bour2015] recommends the *stakeholder circle* for successful stakeholder relationship management. It consists of five steps:

1. Identification of all stakeholders
2. Prioritization to determine who is important
3. Visualization to understand the overall stakeholder community
4. Engagement through effective communication
5. Monitoring the effect of the communication

Active stakeholder relationship management [Bour2009] defines explicitly the rights and obligations of a stakeholder with respect to the development of the proposed system. Depending on the nature of the development, this can be formulated as a stakeholder agreement with the involved stakeholders.

EU 2.2.3 Documentation schema for the stakeholders involved (K3)

The CPRE Foundation Level syllabus [FreA2017] defines what information on stakeholders should at a minimum be documented. In addition, information on stakeholder classification and project-specific attributes should be considered.

According to [Alex2009], stakeholders can be classified by how much the new or modified system affects them:

- Stakeholders of the *system itself*: directly affected by the new or modified system (users, administrators, operators, …)
- Stakeholders in the *surrounding context*: indirectly affected by the new or modified system (business managers, project owners, sponsors, clients, …)
- Stakeholders from the *wider context*: having an indirect relationship to the new or modified system or to the development project (legislators, standard setting bodies, (non-) governmental organizations, competitors, IT staff)

It may also be useful to document additional information relevant for the specific development effort.

In defining additional information, the specific circumstances of the current context have to be considered. Possible influencing factors are:

- *Public relevance*: In a context of high public relevance, it may be useful to document how much a stakeholder knows or can influence public opinion.
- *Time criticality*: In a context with a very strict timeframe, the availability or response time of a stakeholder can be very important information when critical decisions are to be taken.

During the development, all stakeholder information must be continuously updated and adapted to the specific circumstances.

Some commonly used forms of documentation are stakeholder table, stakeholder database, and stakeholder mind map.

EU 2.2.4 The user as a special stakeholder group (K3)

In principle, every system will eventually have users. However, not all systems have direct interaction with humans: some deliver their functionality through other systems. For interactive systems with a human interface, all direct users of the system are of prime interest for the Requirements Engineer.

In-house users (in-company, individually known and involved) are significantly different from outside users (e.g. buyers of consumer products; outside of the company, generally not individually known and not directly involved).

Usually, the number of (potential) users does not allow involving all individuals in the elicitation process. For this reason, the actual users can be aggregated into user groups, based on user analysis or on the domain knowledge of other stakeholders.

A common way to represent user groups is the use of personas [Coop2004]. Personas are fictitious individuals, representing typical user groups of the system with similar needs, goals, behaviors or attitudes. Personas are modeled from data collected about real users through *user research* [BaCC2015]. If no relevant user research data is (yet) available, *provisional personas*, also called *ad-hoc personas* [CECN2014] can be created.

The user groups or personas should be prioritized to define the primary and the secondary user groups/personas. The system, especially its user interface, will be optimized for the primary user group.

The concept of the user's experience (UX) especially addresses the creation of a great experience for users. A definition of user experience is provided in an ISO standard. [ISO9241-210] defines user experience as "a person's perceptions and responses that result from the use or anticipated use of a product, system or service". Knowing ideas and principles of user experience is valuable for the development of interactive systems.

EU 2.3 Identify, classify, manage documents (K3)

Documents are another valuable source for requirements. They are used to transfer concepts between humans over time and distance.

EU 2.3.1 Identifying and selecting documents as requirements sources (K3)

Possible types of documents used as requirements sources are:

- Technical standards, legislation, internal regulations
- Requirements documents (e.g., of similar systems or of the system to be replaced)
- User manuals (e.g., of competitor systems)
- Strategy papers
- Goal documentation
- Business Process Documentation

When *pragmatically identifying* documents, Requirements Engineers use their current knowledge and experience of the context (e.g. domain) to name relevant documents and document types.

During *systematic document identification*, the Requirements Engineer defines elicitation objectives with a dedicated focus on the identification of documents. Two different types of elicitation objectives have to be considered:

- *Information-focused*: finding documents for certain required information
- *Document-focused*: finding documents of certain types considered relevant for the development

For *systematic document identification*, the Requirements Engineer can:

- Search for representatives of typical document categories
- Search for references in already identified documents to other possibly relevant documents
- Ask already identified stakeholders for relevant documentation
- Search for documentation on already identified relevant systems (see EU 2.4)

To decide whether a document is relevant as a requirements source or not, the Requirements Engineer needs to establish specific criteria.

EU 2.3.2 Documentation schema for the documents (K3)

At a minimum, the following information should be recorded for documents that are potentially to be used as sources of requirements:

- Document title
- Place where the document is kept
- Version of the document
- Short description (what kind of information the document can provide)
- Relevance

Depending on the context, additional information may also be relevant.

Documents always have a certain relation to stakeholders, which may also be recorded, e.g.

- Stakeholders, mentioning the relevance of the document
- Author, issuing organization
- Organizations using the document in their processes
- Organizations involved in verifying the adherence

The Requirements Engineer has to keep the information about documentation up to date. This includes reconsidering whether additional documents have become relevant, or whether documents identified earlier have lost relevance. Special attention should be given to changes, updates and version numbering.

EU 2.4 Identify, classify, manage systems (K3)

In the context (both direct and broad) of a system, other systems can be identified as sources of requirements.

EU 2.4.1 Identifying and selecting systems as requirements sources (K3)

Possible types of systems used as requirements sources are:

- Interfacing systems including legacy systems
- Systems sharing a platform / environment / ecosystem
- Competitor systems
- Systems with similar data, functionality or user interfaces
- Predecessor system(s) to be replaced
- Future systems (under construction or even only planned)

When *pragmatically identifying* systems, Requirements Engineers use their current knowledge and experience of the project and its context (e.g. domain) to name relevant systems and system types.

During *systematic system identification*, the Requirements Engineer defines elicitation objectives with a dedicated focus on the identification of systems. Two different types of elicitation objective have to be considered:

- *Information-focused*: finding systems that contain certain required information
- *System-focused*: finding systems of certain types considered relevant for the development project

For *systematic identification*, the Requirements Engineer can:

- Use the system context documentation
- Ask already identified stakeholders for information on relevant systems
- Search already identified documents for information on relevant systems
- Use idea-generating techniques to identify possible analogous systems
- Conduct market research to identify competitor systems
- Consider legacy systems

EU 2.4.2 Documentation schema for systems (K3)

Systems used as a source of requirements have to be documented with at least the following information:

- Name of the system
- Type of system (e.g., competitor system, predecessor system, interfacing system, ...)
- A brief description on data, functionality, processes, user groups, …

Depending on the context, additional information may be relevant.

Special attention should be paid to directly interfacing systems. These can be categorized as:

- Data sources, providing data
- Data sinks, using data
- Supporting systems like an operating system (OS) or Database Management System (DBMS)

Systems always have a certain relation to stakeholders, which may also be recorded, e.g.,

- Stakeholders/Organizations that use the system in direct or indirect ways in their processes
- Stakeholders/Organizations that operate the system
- Stakeholders/Organizations that design, develop, or market the system
- Stakeholders/Organizations that maintain the system, offer support or training
- Organizations that observe the system (e.g. governments, NGOs)

Information about systems is usually present in documents. These documents should be managed separately as requirements sources (see EU 2.3).

The Requirements Engineer has to keep the documentation of potential source systems up to date. This includes reconsidering whether additional systems have become relevant, or whether systems identified earlier have lost relevance. Special attention should be given to changes, updates and version numbering.

EU 3 Elicitation Techniques (K3)

Duration: 8.0 hours

Terms: Elicitation technique, attribute, classification, thinking tools

Educational objectives:

EO 3.1 Understanding the difference between gathering techniques, design and idea-generating techniques, and supporting techniques/thinking tools
EO 3.1.1 Applying interview, questionnaire, and workshops as examples for questioning techniques
EO 3.1.2 Applying field observation, apprenticing, and contextual inquiry as examples for observation techniques
EO 3.1.3 Applying system archeology, perspective-based reading, and re-use as examples for artifact-based techniques
EO 3.2 Applying preconditions for creativity
EO 3.2.1 Applying brainstorming and analogy techniques as examples for idea-generating techniques
EO 3.2.2 Applying prototyping, scenarios, and storyboards as examples for design techniques
EO 3.3.1 Understanding and using abstraction levels, problems and goals, models, transformation effects and mind-mapping as examples for thinking tools
EO 3.4 Understanding elicitation technique attributes as exemplary approach for structuring elicitation techniques

This EU differentiates between gathering techniques (EU 3.1), design-/idea-generating techniques (EU 3.2) and thinking tools (EU 3.3). This differentiation is of course an artificial one. In practice, there is no clear separation between the techniques. However, for presentation and teaching purposes, the differentiation is important to structure the techniques and to learn the main focus of the techniques.

EU 3.4 provides typical identifying characteristics of elicitation techniques. These may be used to describe new techniques and to give general guidelines as to which identifying characteristics are potentially useful in a given project situation.

EU 3.1 Gathering techniques (K4)

Gathering techniques are established techniques for requirements elicitation. They help to elicit satisfiers and dissatisfiers.

EU 3.1.1 Questioning techniques (K3)

Questioning techniques aim to pose appropriate questions to stakeholders. An important distinction is between open-ended and closed-ended questions.

EU 3.1.1.1 Interview (K3)

In an interview, the Requirements Engineer asks one or more stakeholders questions in order to elicit new requirements or to refine existing ones. It requires thorough preparation. During the interview, the answers must be recorded in a suitable way to support post-processing of the interview results. There are different types of interviews, e.g. interviews with a defined set of questions or interviews with an open set of questions. [Port2013], [BaCC2015]

EU 3.1.1.2 Questionnaire (K2)

Several people are asked to answer in writing the same set of questions, presented in a structured way. Quantitative questionnaires are mainly used to confirm previously elicited requirements, whereas qualitative questionnaires are more suited to the elicitation of new requirements. The former can be evaluated quickly and deliver statistical information, the latter tend to deliver complex results and are thus usually more time-consuming to prepare and to evaluate [BaCC2015], [Harr2014].

EU 3.1.1.3 Requirements Workshops (K3)

Workshop is an umbrella term for group-oriented techniques. They can be conducted in very different ways and may include other elicitation techniques or even process patterns (e.g. a design thinking workshop within an agile development). Workshop formats range from small informal meetings to organized events with several hundred stakeholders. [Gott2002]

EU 3.1.2 Observation techniques (K3)

Observation techniques aim at extracting requirements from observation of, e.g. processes, users, or typical usage situations.

Special attention should be given to the investigators' simplification bias [BaCC2015]: inexperienced (novice to the domain) observers have the tendency to simplify the expert user's problem-solving strategies while observing them. Thus, it is highly recommended to talk to subject matter experts before using observation techniques, and/or let subject matter experts review the observation notes, to minimize this bias.

EU 3.1.2.1 Field observation (K3)

The Requirements Engineer observes the stakeholders during their work in their usual environment without interfering. The observations made are used to derive requirements which have to be confirmed by review or further elicitation techniques.

EU 3.1.2.2 Apprenticing (K2)

The Requirements Engineer conducts a short hands-on training in the environment in which the system to be developed/improved will later be used or is already in use. Experienced subject matter experts teach the Requirements Engineer to empower him/her to better understand the domain and therefore to better elicit requirements.

EU 3.1.2.3 Contextual Inquiry (K3)

Contextual inquiry (CI) is an iterative, field data-gathering technique where the Requirements Engineer studies a few carefully selected users in depth to arrive at a fuller understanding of the work practice across the entire user base [BeHo1998].

CI is based on four principles:

- *Context*: go to the user's own context to observe them performing their tasks
- *Partnership*: ask them about their work and engage them in uncovering unarticulated aspects of work
- *Interpretation*: develop a shared understanding with the user about the aspects of work that matter
- *Focus*: in the preparation of the CI, define elicitation objectives and direct your investigation to gather the relevant data in order to reach the objectives

EU 3.1.3 Artifact-based techniques (K3)

Artifacts are products of human work, such as IT systems, documents, images, audio and video files, etc. Some types of artifacts are relevant as sources of requirements. Usually, it is a time-consuming task to examine artifacts in detail.

EU 3.1.3.1 System archaeology (K3)

System archaeology is a technique to elicit information regarding a new system from the documentation, UI or code of a legacy or competitor system. It is recommended to start analyzing documents like specifications, test documentation or user manuals first, as they contain information similar to requirements. With the help of system archaeology, it can be ensured that no requirements implemented in the current system get lost.

EU 3.1.3.2 Perspective-based reading (K3)

The Requirements Engineer uses a specific perspective, e.g. usage perspective or data perspective, in order to retrieve relevant requirements from a document [Pohl2010].

EU 3.1.3.3 Reuse of requirements (K3)

If similar projects or previous versions of the system to be developed exist within the company, requirements from those projects can be reused. Requirements considered for reuse have to be confirmed by review or additional elicitation techniques.

EU 3.1.3.4 Crowd-based Requirements Engineering (K2)

For some systems (e.g. mobile applications), requirements can be collected from "the crowd". This contains explicit data (e.g. feedback, reviews) as well as implicit data (e.g. usage data, error logs) [MNJR2015], [GrDA2015].

EU 3.2 Design and idea-generating techniques (K4)

Traditionally, Requirements Engineering literature talks about creativity techniques. They aim at creating ideas to find solutions for a given question, problem, or goal. Popular examples of such techniques are brainstorming [Osbo1979] or 6-thinking hats [DeBo2006]. In requirements elicitation, creativity techniques are used to create new or innovative requirements that often are delighters.

Outside the software and Requirements Engineering community, the broader term design techniques has emerged. Design techniques subsume creativity techniques for idea generation and provide additional or combined techniques to elaborate ideas and gain further insights for a given idea [Kuma2013]. Popular techniques for this purpose include prototyping (e.g., mock-ups), storyboarding and scenarios.

Preconditions for creativity

Creativity arises not by command, but by chance. Creativity is most likely to occur when all four of the following preconditions are met [KrSc2017]:

- *Chance* – and therefore time – for an idea to come up
- *Knowledge* of the subject matter, which raises the odds for an idea that makes the difference
- *Motivation*, as our brain can only be creative if there is a direct benefit for its owner
- *Safety and security*, as useless ideas must not have negative consequences

Idea-generating and design techniques help in some or all of these aspects to create a suitable environment for new ideas and innovations to evolve.

EU 3.2.1 Brainstorming (K3)

Brainstorming was developed in the 1940s-1950s by Alex F. Osborn [Osbo1979]. Like most creativity techniques, the crucial point of brainstorming is the separation of finding ideas from the analysis of ideas. It is conducted in groups of about 5-10 people and lasts about 20 minutes. A moderator ensures the orderly conduct of the brainstorming and the documentation of ideas.

Many different variants have evolved over time, e.g. brainstorming paradox, method 6-3-5, brainwriting.

EU 3.2.2 Analogy techniques (K2)

Analogy techniques (e.g. bisociation [Koes1964]) are techniques that help to come up with ideas for critical and also complex topics. They use analogies to support thinking and generating ideas. Their success or failure is mainly influenced by the quality of the analogy. The relevance of similar systems is discussed in EU 2.4.

EU 3.2.3 Prototyping (K3)

Prototyping is an umbrella term and refers to the creation of various types of early samples or models built to gain live experiences with a concept or process.

For requirements elicitation, the term prototype intentionally does not only refer to implementing prototypes in software. Instead, it also refers to everything that can represent requirements of a system to be developed (e.g. sketch of user interface, physical mock-up, video). The purpose of prototyping in requirements elicitation is the simulation of the new system and the exploration of requirements by stimulation of agreement and objection or clarification and amendment.

A prototype can be evaluated by application of a user walkthrough [ShRP2007] or user/usability testing [RuCh2008]. Often, the outcome of such an evaluation is a set of new requirements.

EU 3.2.4 Scenarios and storyboards (K3)

Usage scenarios describe in the form of a realistic example, how a user will interact with the proposed system [RoCa2002].

A storyboard is a visualized scenario. It looks like a comic, with a set of pictures and/or screenshots, and therefore visualizes how a system or product is to be used. The reflection on a concrete example allows clients and users to envision requirements in the actual application situation and thus review and amend them [RiFl2014].

EU 3.3 Thinking tools (K2)

The elicitation techniques introduced so far represent techniques that describe a certain way of gathering information or producing a certain artefact for the purpose of requirements elicitation.

In this section, techniques are presented that cross-cut these types of techniques since they foster a certain way of thinking. We call them supporting techniques and thinking tools, because they are not applied by themselves, but in conjunction with other techniques.

EU 3.3.1 Thinking in abstraction levels (K2)

Abstraction levels are a powerful thinking tool in requirements elicitation [GoWo2005], [Laue2014]. They are often used as a kind of process model to structure the elicitation work, i.e. first elicit requirements only on the highest level and continue with further levels. It can be further used to structure requirements information obtained, to identify gaps in requirements or unnecessary requirements and to focus the elicitation activities to a certain abstraction level. For example, in a workshop with users, it is advisable to focus on the system context since the users are the experts for the system context. Talking about the data structures of a system may not be appropriate since the users do not care about the internal data structures.

EU 3.3.2 Thinking in terms of problems and goals (K2)

A requirement is "*a condition or capability needed by a user to solve a problem or achieve an objective*" (a goal), see CPRE Glossary [Glin2017]. Thinking in terms of problems and goals thus is a core competence for the Requirements Engineer.

A *problem* is the state of a certain aspect in the context of a stakeholder, that is experienced as negative. A problem can exist in the present (an actual problem).

A *goal* is the state of a certain aspect in the context of a stakeholder that is expected to be positive. A goal exists in the future only.

Problems and goals do not exist in real world: they are mental constructs of stakeholders. The same topic can be conceived as a problem by one stakeholder and serve as a goal for another. Problems and goals can only be known by communicating with the pertaining stakeholders.

Problem and goal are interconnected by another mental construct: The *solution* is a roadmap for a certain intervention in the context of the stakeholder. Usually, more than one solution may solve the problem and reach the goal (to a certain extent). For more information, see [LoLS2017].

Thinking in terms of problems and goals enables the Requirements Engineer to analyze and uncover the complete network of problems, solutions and goals. In literature, there are several approaches that focus on problems, e.g., Problem Frames [Jack2001], or goals, e.g., KAOS [Lams2009].

EU 3.3.3 Avoidance of transformation effects (K2)

In the CPRE Foundation Level syllabus [FreA2017], transformational effects are discussed in the context of the output of Requirements Engineering, being the documentation.

Requirements engineers should also be aware of these (and other) transformational effects in their input, as they frequently occur during elicitation activities in communication with stakeholders or when reading documents. Encountering this kind of effects is a trigger for additional elicitation efforts that will probably reveal additional or detailed requirements.

EU 3.3.4 Thinking in terms of models (K2)

The CPRE Foundation Level syllabus [FreA2017] introduces several types of models (e.g. data flow diagrams, activity diagrams) for documenting requirements. Models allow focus on a specific perspective of a system: data, function, behavior. Models can also serve as a thinking tool if the Requirements Engineer wants to focus on a specific perspective during a particular elicitation activity, e.g., discuss an activity diagram in a stakeholder interview or develop a data flow diagram in a workshop with stakeholders. However, the Requirements Engineer should keep in mind that models are only useful if the modelling language is understood by all involved stakeholders.

EU 3.3.5 Mind mapping (K3)

Mind mapping is a graphical thinking tool [Buza1993]. By putting a main topic in the center and spreading out the ideas in branches, thoughts and ideas can be sorted and structured. Text and images should both be used as well as color. “Boring” representations (straight lines, only one color) should be avoided to make the representation more “stimulating” for the brain.

EU 3.4 Example for structuring elicitation techniques: attributes (K2)

Requirements engineers should carefully select which elicitation techniques to use based on the specific context and needs of the situation at hand. To support this selection, techniques can be classified by certain attributes. An example of useful attributes is presented in Table 1.

The very nature of an elicitation technique can be described by a combination of these attributes.

For example, the *"interview"* technique is characterized by the attributes *"conversational"* and *"questioning"*. An interview might also be *"observational"* in case the Requirements Engineer conducts the interview at the location of an intended end user. However, *"observational"* is not a core attribute of interviews, as they could also be conducted by phone or at other locations without relevant observations being possible.

Table 1 defines relevant attributes. The Requirements Engineer should consider the availability and characteristics of stakeholders, customer needs, the project objectives and constraints, the domain and the context in which she/he is working (see EU 1.3) when selecting an elicitation technique. Classifying a long list of available techniques by relevant attributes can help to select the techniques to be used in a specific situation. *"There are good practices in context, but there are no best practices"* [KaBa2012]: every situation may require a specific combination of techniques to be successful.

Table 1 Attributes for classifying elicitation techniques.

Attribute	Short description	Aiming at the following goals	Suitable in the following situations
Conversational	A dialogue between Requirements Engineer and stakeholder(s)	To understand the system context; to elicit goals and obtain an overview of satisfiers (Kano)	When (relevant) stakeholders are available for oral information exchange
Questioning	Asking stakeholders (at least partly) prepared questions to learn about facts or about their opinion	To elicit goals and satisfiers; to verify dissatisfiers; to obtain stakeholder's opinion or additional information on previously elicited requirements; to elicit detailed information; to clarify specific requirements	If relevant questions can be formulated upfront; if some form of communication with stakeholders is possible; if complicated subject matter is concerned
Observational	Observing stakeholders' behaviors in a live situation, usually operating an existing system or performing specific tasks	To gather information about the stakeholder's actual behavior; to elicit dissatisfiers; to analyze usability requirements; to collect data about the user's context	If stakeholders cannot be addressed directly or if they are unable to state their needs and actions (detailed enough); when in doubt on congruence between actual and stated situation; to improve understanding the users' needs; to improve understanding of the project (e.g. in preparation for other elicitation techniques)
Provoking (dis-) agreement	Demonstrating relevant aspects of a solution to get affirmative or contradicting feedback from stakeholders	To make requirements tangible for stakeholders; to evaluate previously elicited requirements; to get feedback on variants of a solution	If stakeholders have trouble imagining things; if the Requirements Engineer can explain or show aspects of the proposed solution to the stakeholders (or even let them use it); if stakeholders have trouble explaining what they need

Attribute	Short description	Aiming at the following goals	Suitable in the following situations
Artefact-based	Analyzing artefacts (e.g., documents, models, products or systems in use)	To derive requirements from existing artefacts; to elicit (dis-)satisfiers, especially constraints	When relevant artefacts are available and accessible; to improve understanding of the project and of the domain (e.g. in preparation for other elicitation techniques); if stakeholders are not directly available
Creativity-stimulating	Foster creativity and innovation	To elicit delighters; to come up with novel approaches	If innovation is needed; if a predetermined direction is absent; when other approaches fail
Experiencing	Experiencing the environment and problem space where the system to be developed will be used	To derive requirements from the real-life circumstances; to understand the problem to be solved from users in their work context; to gain empathy	If users and usability are key aspects of the project; when it is possible to access the environment where usage actually takes place

There are also other ways of categorizing elicitation techniques, e.g.

- Kano Model see [FreA2017]
- Design Innovation Process [Kuma2013]

EU 4 Conflict Resolution (K3)

Duration: 2.0 hours

Terms: Necessity, consistency, completeness, feasibility, requirements conflict, social conflict

Educational objectives:

EO 4.1.1 Understanding the difference between requirements conflicts and other social conflicts
EO 4.1.2 Applying the identification of conflicts
EO 4.2.1 Applying the classification of conflict types
EO 4.2.2 Understanding as a Requirements Engineer which conflicts to solve and which to delegate
EO 4.3.1 Applying selection of suitable negotiation techniques based on characteristics of the conflict
EO 4.3.2 Applying use of agreement, compromise, variant construction, voting, and overruling as examples of negotiation techniques
EO 4.4.1 Understanding the documentation of requirements conflict resolutions

During elicitation, the Requirements Engineer uncovers, gathers and designs a broad collection of requirements. Elicitation techniques by themselves do not ensure that this collection as a whole is clear, complete, consistent, unambiguous and acceptable. For the final set of requirements, however, all stakeholders have to understand and agree on all requirements that are relevant to them. If some stakeholders do not agree, this situation is to be recognized as a requirements conflict that should be resolved accordingly.

Conflict resolution in the broad sense consists of four tasks:

- Conflict identification
- Conflict analysis
- Conflict resolution
- Documentation of conflict resolution

Conflict identification and analysis is an ongoing activity in Requirements Engineering and is a prerequisite for resolving any conflict. Once a requirements conflict has been identified, the Requirements Engineer should initiate conflict resolution activities to select a proper resolution technique and to document its outcome.

EU 4.1 Conflict identification (K2)

Conflicts in general are a subject of social sciences and typically referred to as "social conflict" to indicate that a conflict arises between people. A social conflict can be defined as follows: "… an interaction between actors (individuals, groups, organizations and so on), where at least one actor sees incompatibilities in the thinking, imagination, perception, feeling, and/or wanting with another actor (other actors) in a way, that in the realization there is impairment by another actor (the other actors)." [Glas2004]

A requirements conflict can be interpreted as a special type of social conflict and is defined as follows: "A conflict in Requirements Engineering (requirements conflict) is an incompatibility of requirements, based on a contradictory perception of two or more stakeholders." [RueA2014]. There are several indicators by which conflicts can be detected. Indicators can be observed in communication and documentation.

Commonly encountered indicators in *communication* are:

- Denial
- Indifference
- Pedantry
- Questions of detail
- Incorrect interpretation
- Concealment
- Delegation

Commonly encountered indicators in *documentation* are:

- Contradictory statements by stakeholders
- Conflicting results from analysis of documents or systems
- Inconsistent requirements in detail
- Inconsistent usage of terms in specification

Most conflicts tend to be hidden and can only be detected by carefully monitoring these indicators. If one of the indicators occurs, this does not mean that a requirements conflict is present. However, the Requirements Engineer should continuously pay attention. Through most of the requirements elicitation activities, she/he is stimulating the stakeholders to state their positions clearly, thus in some cases revealing unexpected problems or existing conflicts.

EU 4.2 Conflict analysis (K3)

Once a conflict has been identified, the Requirements Engineer has to clarify, whether or not the identified conflict is a requirements conflict. This distinction is important since the resolution of a requirements conflict is the prime responsibility of the Requirements Engineer whereas other conflicts have to be resolved by other participants (e.g. a project manager).

Analyzing the characteristics of a requirements conflict helps the Requirements Engineer understand its nature. The following characteristics [RueA2014] of a conflict can help to understand its nature and to find a proper solution:

- Type of the conflict
- Subject matter of the conflict
- Affected requirements
- Involved stakeholders
- Opinions of the various stakeholders
- Cause of the conflict
- Progress/history of the conflict
- Consequences of the conflict
- Resulting risks

The type of the conflict is important for deciding if a given conflict is a requirements conflict or not. Five different types of conflicts are distinguished [Moor2014]:

- Interest conflict
- Data conflict
- Value conflict
- Structural conflict
- Relationship conflict

Most requirements conflicts can be categorized as interest conflicts, data conflicts and value conflicts. Structural and relationship conflicts are usually not related to requirements and in that case should be resolved by other participants.

However, most conflicts show characteristics of more than one type as different causes might interact. Therefore, Requirements Engineers should pay attention to all kinds of conflict, even if a solution is not within their responsibility.

EU 4.3 Conflict resolution (K4)

A prerequisite for the selection of a proper resolution technique is an in-depth understanding of the nature of the requirements conflict. The following general resolution techniques can be distinguished (see [FreA2017]):

- Agreement
- Compromise
- Voting
- Definition of variants
- Overruling

In addition, there are several auxiliary techniques, for example:

- Non-violent communication [Rose2015]
- Negotiation techniques [FiUP2012]
- Consider-all-facts [DeBo2006]
- Plus-minus-interesting [DeBo2006]
- Decision matrix [BiAB2006], [IsNe2013]

Based on the characteristics of a conflict, suitable conflict resolution techniques should be selected.

EU 4.4 Documentation of conflict resolution (K2)

After its resolution, the conflict should be properly documented. Apart from the characteristics of the conflict mentioned in EU 4.2, this should include in particular:

- Assumptions concerning the conflict and its resolution
- Constraints influencing the choice of conflict resolution technique and/or the resolution
- Potential alternatives considered
- Conflict resolution including reasons for the chosen resolution
- Decision-makers and other contributors

If not documented, stakeholders could simply forget or ignore decisions made, or try to change decisions afterwards. This often occurs in situations where the requirements conflict itself is resolved, but an underlying (other) social conflict is not resolved.

EU 5 Skills of the Requirements Engineer (K2)

Duration: 1.5 hours

Terms: Skills, communication models, interpretation, response, self-reflection

Educational objectives:
EO 5.1.1 Understanding the required skills in the areas of elicitation
EO 5.2.1 Understanding the fundamentals of communication theory
EO 5.3.1 Applying self-reflection on personal skills in requirements elicitation
EO 5.4.1 Understanding provisions for personal development
EO 5.5.1 Understanding learning from previous experiences

EU 5.1 Required skills in the areas of elicitation (K2)

In the CPRE Foundation Level [FreA2017], communication skills, analytical thinking, empathy, conflict resolution skills, moderation skills, self-confidence and the ability to convince are presented as the required (soft) skills of a Requirements Engineer. For the elicitation of requirements at the Advanced Level, the following characteristics are also relevant:

- Self-awareness
- Contextual awareness
- Motivating nature
- Leadership
- Flexibility
- Reflection
- Neutrality
- Intercultural competency
- Ethical conscience

Of all these skills, communication skills are the key success factor for the Requirements Engineer. All interaction between the Requirements Engineer and the stakeholders, being the prime sources of requirements, is a form of communication and all above-mentioned skills play a role in it.

EU 5.2 Communication theory and communication models (K2)

Communication is about *sharing meaningful concepts* between individuals. During communication, information may be lost, added, distorted or misinterpreted. The Requirements Engineer should take care to prevent these problems as far as possible.

Understanding the theory and models and being able to integrate this knowledge into daily communication activities will improve the Requirements Engineer's communication and lead to better results.

A fundamental understanding of communication theory can be obtained by studying the following communication models:

- The Shannon-Weaver model [ShWe1971]
- The circular model of communication [Schr1971]
- The "four-sides" model of Schulz von Thun [Schu1981]

EU 5.3 Self-reflection on personal skills in requirements elicitation (K3)

This syllabus and the corresponding training lay out the foundation for successful application of the presented methods and techniques. However, the development and improvement of personal skills for the elicitation of requirements is a long-term learning process.

Even if the Requirements Engineering of a development is considered a success, there are typically several opportunities for improvement. For example:

- Has a technique delivered the expected results / contributed to the development?
- Did the stakeholder(s) accept the elicitation or conflict resolution techniques applied?
- Was the effort for a technique justifiable with respect to the contribution to the development?
- Which technique might have allowed eliciting requirements that came up late in the development at an earlier time?

The proper assessment of one's own abilities can be made on the one hand by direct behavioral observation and on the other hand by subsequent analysis. In a direct observation the focus should be placed on one or at most two characteristics to obtain an accurate and reliable monitoring result (e.g., observation of one's own reflective communication during an interview). To assess your skills in a subsequent analysis, the response of other people is an important source (e.g. 360° feedback [LeLu2009]). An assessment sheet in respect of the previously defined capabilities also is a suitable measuring instrument [SmMa2004].

EU 5.4 Opportunities for personal development (K2)

Insufficient practical experience is very often presented as a reason for not applying a specific elicitation or conflict resolution technique. Such an attitude might be understandable in terms of project success (the Requirements Engineer applies the techniques he/she knows best to ensure project success); in terms of personal development, this attitude is not helpful. A proven alternative is the application of unfamiliar techniques in a low-risk setting (e.g., perform apprenticing with a small subgroup of stakeholders). It is further possible to apply an unfamiliar technique in parallel to a familiar technique (e.g., a questionnaire is applied in parallel to a series of interviews).

EU 5.5 Learning from previous experience (K2)

Essential components of a personal training process that fosters learning from previous experience are:

- Improvement in everyday work
- Regular measurement of your own ability profile
- Training measures
- Mentoring measures

References and further reading

[AlIS1977] C. Alexander, S. Ishikawa, M. Silverstein: A Pattern Language - Towns - Buildings - Construction. Oxford University Press, New York, 1977.

[Alex2005] I.F. Alexander: A Taxonomy of Stakeholders – Human Roles in System Development. International Journal of Technology and Human Interaction, Vol 1, 1: 23-59, Hershey, 2005.

[Alex2009] I. Alexander: Discovering requirements: how to specify products and services. John Wiley & Sons Ltd, Hoboken, 2009.

[BaCC2015] K. Baxter, C. Courage, K. Caine: Understanding Your Users: A Practical Guide to User Research Methods, 2nd edition. Morgan Kaufmann, Burlington, 2015.

[BaGr1975] R. Bandler, J. Grinder: The Structure of Magic I- A Book About Language and Therapy. Science and Behavior Books, Palo Alto, 1975.

[BaGr1976] R. Bandler, J. Grinder: The Structure of Magic II- About Communication and Change, Science and Behavior Books, Palo Alto, 1976.

[BeHo1998] H. Beyer, K. Holtzblatt: Contextual Design: Defining Customer-Centered Systems. Morgan Kaufmann, Burlington ,1998.

[Beve1957] W. I. B. Beveridge: The Art of Scientific Investigation. The Blackburn Press, Cambridge, 1957. The full text of this book is available on www.archive.org. Last visited February 2019.

[BiAB2006] S. Biffl, A. Aurum, B. Boehm: Value-Based Software Engineering. Springer-Verlag, Berlin, 2006.

[Bour2009] L. Bourne: Stakeholder Relationship Management: A Maturity Model for Organizational Implementation. Gower Publishing Ltd, Birlington, 2009.

[Bour2015] L. Bourne: Making Projects Work: Effective Stakeholder and Communication Management. CRC Press, Boca Raton, 2015.

[Buza1993] T. Buzan: The Mind Map Book: How to Use Radiant Thinking to Maximize Your Brain's Untapped Potential. BBC Books, London, 1993.

[CaGr2012] E. Cameron, M. Green: Making Sense of Change Management: A Complete Guide to the Models Tools and Techniques of Organizational Chang. 3rd ed., Kogan Page, London, 2012.

[Cock2001] A. Cockburn: Writing Effective Use Cases. Addison-Wesley, Boston, 2001.

[CECN2014] A. Cooper, R. Reimann, D. Cronin, C. Noessel: About Face: The Essentials of Interaction Design. 4th Edition, John Wiley & Sons, Inc, Hoboken, 2014.

[Cohn2004] M. Cohn: User Stories Applied: For Agile Software Development. Addison-Wesley, Boston, 2004.

[Coop2004] A. Cooper: The Inmates Are Running the Asylum-Why High-tech Products Drive Us Crazy and How to Restore the Sanity. Que, Indianapolis, 2004.

[CoSh2007] T. Colburn, G. Shute: Abstraction in Computer Science. Minds & Machines, Vol. 17, pp. 169–184, Springer Science+Business Media B.V., Dordrecht, 2007.

[CrOB2006] O. Creighton, M. Ott, B. Bruegge: Software Cinema: Video-based Requirements Engineering - 14th IEEE International Requirements Engineering Conference. IEEE Computer Society, Washington DC, 2006.

[DeBo2006] E. DeBono: EdwardDeBono's Thinking Course - Powerful Tools to Transform Your Thinking. BBC Active, London, 2006.

[GrDA2015] Eduard C. Groen, J. Doerr, S. Adam: Towards Crowd-Based Requirements Engineering - A Research Preview- Requirements Engineering- Foundation for Software Quality - 21st International Working Conference- REFSQ 2015. Essen, Germany, March 23-26, 2015. Proceedings. pp. 247-253., Cham, 2015.

[FiUP2012] R. Fisher, W. Ury, B. Patton: Getting to Yes - Negotiating an agreement without giving in, 3rd rev. ed. Random House Business, New York, 2012.

[Glas1982] F. Glasl: The process of conflict escalation and roles of third parties. In: G.B.J. Bomers and R.B. Peterson, (eds) Conflict management and industrial relations, Springer-Science+Business Media, Dodrecht, 1982.

[Glas2004] F. Glasl: Konfliktmanagement: Ein Handbuch für Führungskräfte, Beraterinnen und Berater. 11th ed., Freies Geistesleben, Stuttgart, 2004.

[Glin2017] M. Glinz: A Glossary of Requirements Engineering Terminology (Version 1.7 May 2017), IREB

[Gott2002] E. Gottesdiener: Requirements by Collaboration: Workshops for Defining Needs, Addison-Wesley Professional, Boston, 2002.

[GoWo2005] T. Gorschek, C. Wohlin: Requirements Abstraction Model. Requirements Engineering Journal Vol. 11, No. 1, pp. 79-101. http://dx.doi.org/10.1007/s00766-005-0020-7, 2005. Last visited February 2019.

[Harr2014] D. F. Harris: The Complete Guide to Writing Questionnaires- How to Get Better Information for Better Decisions, I&M Press, North Carolina, 2014.

[HoDC2007] P. Holman, T. Devane, S. Cady: The Change Handbook. The Definitive Resource on Today´s Best methods for Engaging Whole Systems. McGraw-Hill Professional Pub Group West, New York, 2007.

[IGKCD2002] A. Iles, D. Glaser, M. Kam, J. Cannyand, E. Do: Learning via Distributed Dialogue: Live Notes and Handheld Wireless Technology. Proc. Conf. Computer Support for Collaborative Learning, Hillsdale, 2002.

[FreA2017] K. Frühauf et al.: IREB Certified Professional for Requirements Engineering Foundation Level Syllabus (version 2.2.1). IREB e.V., Karlsruhe, 2017.

[IsNe2013] A. Ishizaka, P. Nemery: Multi-criteria decision analysis. Methods and software. John Wiley & Sons, Ltd, Chichester, 2013.

[ISO29148] ISO/IEC/IEEE29148:2011. Systems and software engineering – Life cycle processes – Requirements engineering, International Organization for Standardization, Geneva, 2011.

[ISO9241-210] ISO9241-210:2010. Ergonomics of human-system interaction – Part 210: Human-centred design for interactive systems, International Organization for Standardization, Geneva, 2010.

[Jack2001] M. Jackson: Problem Frames – Anaylzing and structuring software development problems. Addison-Wesley, Boston, 2001.

[KaBa2012] C. Kaner, J. Bach: The Seven Basic Principles of the Context-Driven School. http://context-driven-testing.com/, 2012.

[Kell1984] J. F. Kelley: An iterative design methodology for user friendly natural language in office information applications. ACM Transactions on Office Information Systems, March1984, 2:1, 26-41. New York, 1984.

[Kell2002] H. Kellner: Kreativität im Projekt. Hanser Fachbuch, München, 2002.

[Koes1964] A. Koestler: The Act of Creation. Penguin Books, London, 1964.

[KoSo1998] G. Kotonya, I. Sommerville: Requirements Engineering: Processes and Techniques. Wiley Publishing, Hoboken, 1998.

[KrSc2017] I. Kreß, A. Schwarz: To Brainstorm or Not to Brainstorm – Neuropsychological Insights on Creativity. Requirements Engineering Magazin Vol. 2017-02. https://re-magazine.ireb.org/issues/2017-02-staying-on-the-right-path/to-brainstorm-or-not-to-brainstorm-neuropsychological-insights-on-creativity/. Last visited February 2019.

[Kuma2013] V. Kumar: 101 Design Methods – A Structured Approach for Driving Innovation in Your Organization. Wiley, 2013.

[Kuni2003] M. Kuniavsky: Observing the User Experience: A Practioners's Guide to User Research, Morgan Kaufmann, Burlington, 2003.

[Lams2009] A. van Lamsweerde: Requirements Engineering: From System Goals to UML Models to Software Specifications. JohnWiley&Sons, Hoboken, 2009.

[Laue2014] K. Lauenroth: What does it mean to say "requirement"? - An inquiry into the abilities of the human mind and the meaning of the word "requirement". Requirements Engineering Magazin Vol. 2014-01. http://re-magazine.ireb.org/issues/2014-1-learning-to-fly/what-does-it-mean-to-say-requirement, 2014. Last visited February 2019.

[Leff2011] D. Leffingwell: Agile Software Requirements. Lean Requirements Practices for Teams, Programs, and the Enterprise. Addison-Wesley, Boston, 2011.

[LeLu2009] R. Lepsinger, A.D. Lucia: The Art and Science of 360 Degree Feedback. 2nd ed., Wiley, San Francisco, 2009.

[LeWi2003] D. Leffingwell, D. Widrig: Managing Software Requirements: A Use Case Approach. Addison-Wesley, Boston, 2003.

[LoLS2017] H. van Loenhoud, K. Lauenroth, P. Steiger: The goal is to solve the problem - Some thoughts on problems and goals in the context of Requirements Engineering. Requirements Engineering Magazin Vol. 2017-02. http://re-magazine.ireb.org/issues/2017-02-staying-on-the-right-path/the-goal-is-to-solve-the-problem/. Last visited February 2019.

[MNJR2015] W. Maalej, M. Nayebi, T. Johann, G. Ruhe: Toward Data-Driven Requirements Engineering. IEEE Software Vol. 33, No. 1, pp.48-54, 2015.

[MaGi2001] N. Maiden, A. Gizikis: Where Do Requirements Come From? IEEESoftware18, 5: 10-12, Geneva, 2001.

[Mayh1999] D. J. Mayhew: The Usability Engineering Lifecycle. Morgan Kaufmann, Burlington, 1999.

[McCo2006] S. McConnell: Software Estimation – Demystifying the Black Art. Microsoft Press, Redmond, 2006.

[Moor2014] C. W. Moore: The Mediation Process – Practical Strategies for Resolving Conflicts. 4th ed., John Wiley & Sons, Hoboken, 2014.

[Niel1993] J. Nielsen: Usability Engineering. Morgan Kaufmann, Burlington, 1993.

[Osbo1979] A. F. Osborn: Applied Imagination. 3rd rev. ed., Charles Scribner´s Sons, New York, 1979.

[Parn1972] D.L. Parnas: On the Criteria To Be Used in Decomposing Systems into Modules (PDF). Communications of the ACM 15 (12), pp.: 1053–58. doi:10.1145/361598.361623, ACM New York, 1972.

[Pohl2010] K. Pohl: Requirements Engineering – Fundamentals, Principles, and Techniques. Springer, Berlin, 2010.

[PoRu2015] K. Pohl, C. Rupp: Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB compliant, Rocky Nook, Santa Barbara, 2015.

[Port2013] S. Portigal: Interviewing Users: How to Uncover Compelling Insights. Rosenfeld Media, Brooklyn, 2013.

[Reif 2012] D.J. Reifer: Software Change Management: Case Studies and Practical Advice. Microsoft Press, Redmond, 2012.

[RiFl2014] M. Richter, M. Flückiger: User-Centred Engineering: Creating Products for Humans. Springer, Heidelberg, 2014.

[Robs2011] C. Robson: Real World Research. John Wiley & Sons, Hoboken, 2011.

[RoCa2002] M. Rosson, J. M. Carroll: Usability Engineering: Scenario-Based Development of Human Computer Interaction. Morgan Kaufmann, Burlington, 2002.

[Rohr1969] B. Rohrbach: Kreativ nach Regeln–Methode 635, eine neue Technik zum Lösen von Problemen. Absatzwirtschaft 12, Heft 19:73-75, Meedia GmbH & Co.KG, Hamburg, 1969.

[RoRo2013] S. Robertson, J. Robertson: Mastering the Requirements Process: Getting Requirements Right. Third Edition, Pearson Education, London, 2013.

[Rose2015] M. B. Rosenberg: Nonviolent Communication - A Language of Life. 3rd rev. ed., Puddle Dancer Press (US), Encinitas, 2015.

[Royc1972] W. Royce: Managing the Development of Large Software Systems. Technical Papers of Western Electronic Show and Convention (WesCon) August 25–28, Los Angeles, 1972.

[RuCh2008] J. Rubin, D. Chisnell: Handbook of Usability Testing- How to Plan, Design, and Conduct Effective Tests.Wiley; Idianapolis, 2008.

[RueA2014] C. Rupp, die SOPHISTen: Requirements-Engineering und –Management - Aus der Praxis von klassisch bis agil. 6th ed., Carl Hanser Verlag, München, 2014. (selected chapters English version see http://www.sophist.de/en/infopool/downloads/). Last visited February 2019.

[Schr1971] W. L. Schramm: How communication works. in W. L. Schramm, ed., The Process and Effects of Mass Communication. rev. ed., University of Illinois Press, Champaign, 1971.

[Schu1981] F. Schulz von Thun: Miteinander reden 1 - Störungen und Klärungen. Psychologie der zwischenmenschlichen Kommunikation. Rowohlt, Reinbek, 1981.

[Shac1991] B. Shackel: Usability - Context, Framework, Definition, Design and Evaluation. In B. Shackel, S. Richardson (Eds.): Human Factors for Informatics Usability (p. 21-37), University Press, Cambridge, UK, 1991.

[ShRP2007] H. Sharp, Y. Rogers, J. Preece: Interaction Design: Beyond Human-Computer Interaction. John Wiley & Sons, Hoboken, 2007.

[ShWe1971] C. E. Shannon, W. Weaver: The Mathematical Theory of Communication University of Illinois Press, Champaign, 1971.

[SmMa2004] S. Smith, R. Mazin: The HR Answer Book: An Indispensable Guide for Managers and Human Resources Professionals, AMACOM, Hertogenbosch, 2004.

[Wieg2003] K. E. Wiegers: Software Requirements. Microsoft Press, Redmond, 2003.

[Ziel2007] P. Zielczynski: Requirements Management Using IBM Rational RequisitePro. Pearson Education (Us), New York, 2007.