People matter, results count.

Software engineering in the industrial practice

Requirement analysis and their management

Lecture TU Darmstadt WS 2015/2016
Christian Hinken
Darmstadt, 06.11.2015

Research and science live on the exchange of ideas, the clear arrangements are thereby useful:

If you have any questions, please contact:
Capgemini | Offenbach
Dr. Martin Girschick
Berliner Straße 76, 63065 Offenbach, Germany

Telephone +49 69 82901-376
Email: martin.girschick@capgemini.com

AGENDA

- The term “requirement”
- Process of requirements analysis
- Examples of requirements
- Requirement Management in agile projects

AGENDA

- **The term “requirement”**
- Process of requirements analysis
- Examples of requirements
- Requirement Management in agile projects

Projects often fail because of insufficiencies requirements

Reason for project fail	Share
Incomplete requirements	13,1 %
Insufficient user involvement	12,4%
Low capacities	10,5%
Overtaken unrealistic requirements	9,9 %
Changing requirements (moving targets)	8,7 %
Unreliable estimations based on unstable requirements	5%

Source: Chaos Report of Standish Group

A good requirements analysís is important
Requirements management is important

What is a requirement?

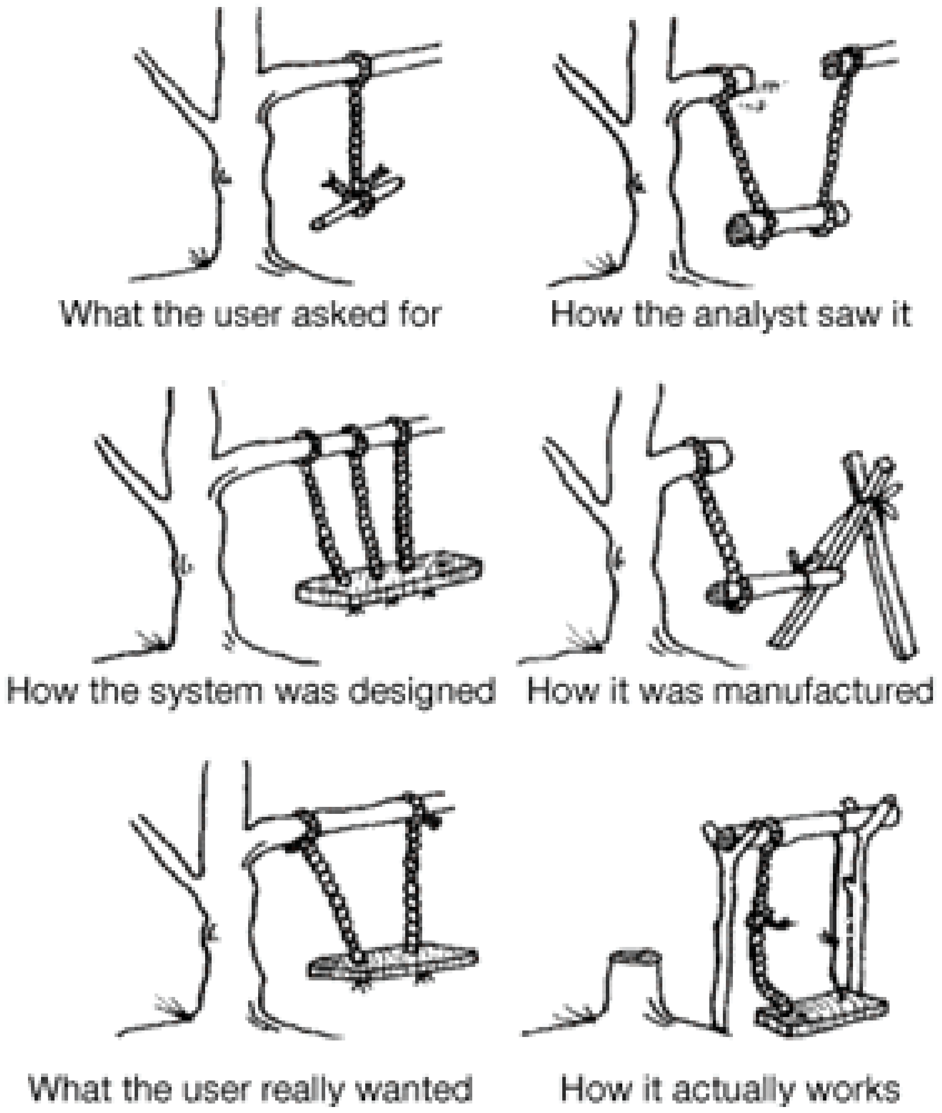
A requirement is a required feature of an application

It is used for communication between the parties involved in the application

It is a part of the contract with the vendor

It is a basis for the realization phase of the project

It is a basis for the approval of the specification and software



Important features of a requirement

Requirements are testable

- Not testable requirements are informations

Requirements describe a problem to be solved

- They do not describe the solution

Concerning the level of detail requirements are above system specifications

Requirement should fulfill as many as possible of the following conditions:

- It is correct and consistent
- It is implementable
- It is necessary and priorizited
- It is usable and usefull

Types of requirements: system requirements

- **User requirements describe the application as a whole from the user's perspective.**
  - e.g. written in form of a concept created by members of the business department
  - They form a basis for the definition of requirements (which follows after the user requirements)
- **Functional requirements** define desired **business** / **technical functionalities**.
- **Quality requirements** (known as **„non-functional requirements“**) define **criteria**, which must be **fulfilled by the application**.
  - No functionalities, but **testable** conditions the system must fulfill
- **Architecture requirements** define architectural conditions to be fulfilled by the application (akin with quality requirements)

System requirements

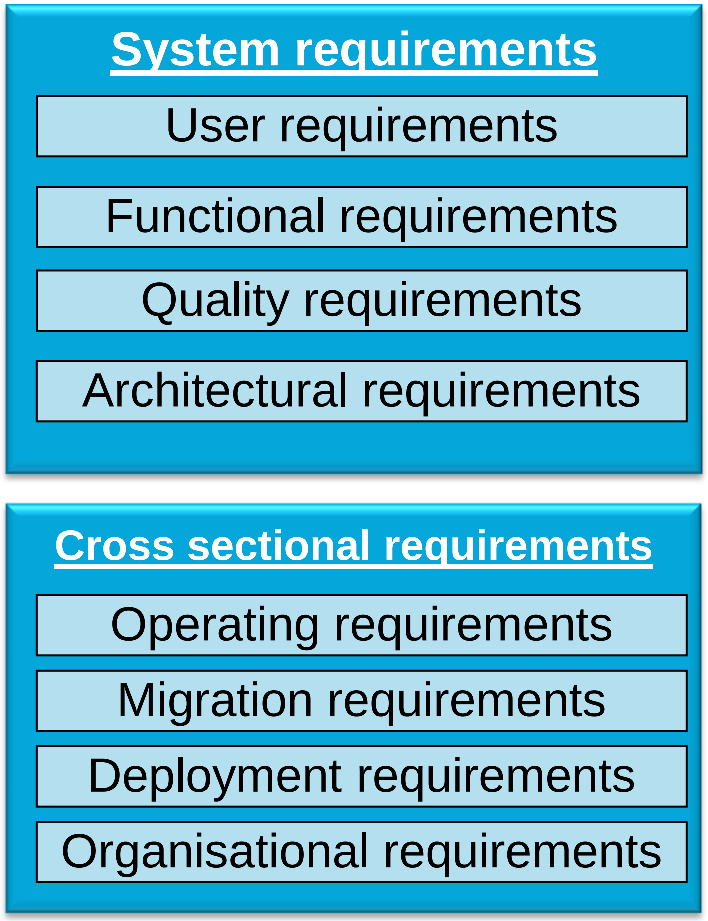
- User requirements
- Functional requirements
- Quality requirements
- Architectural requirements

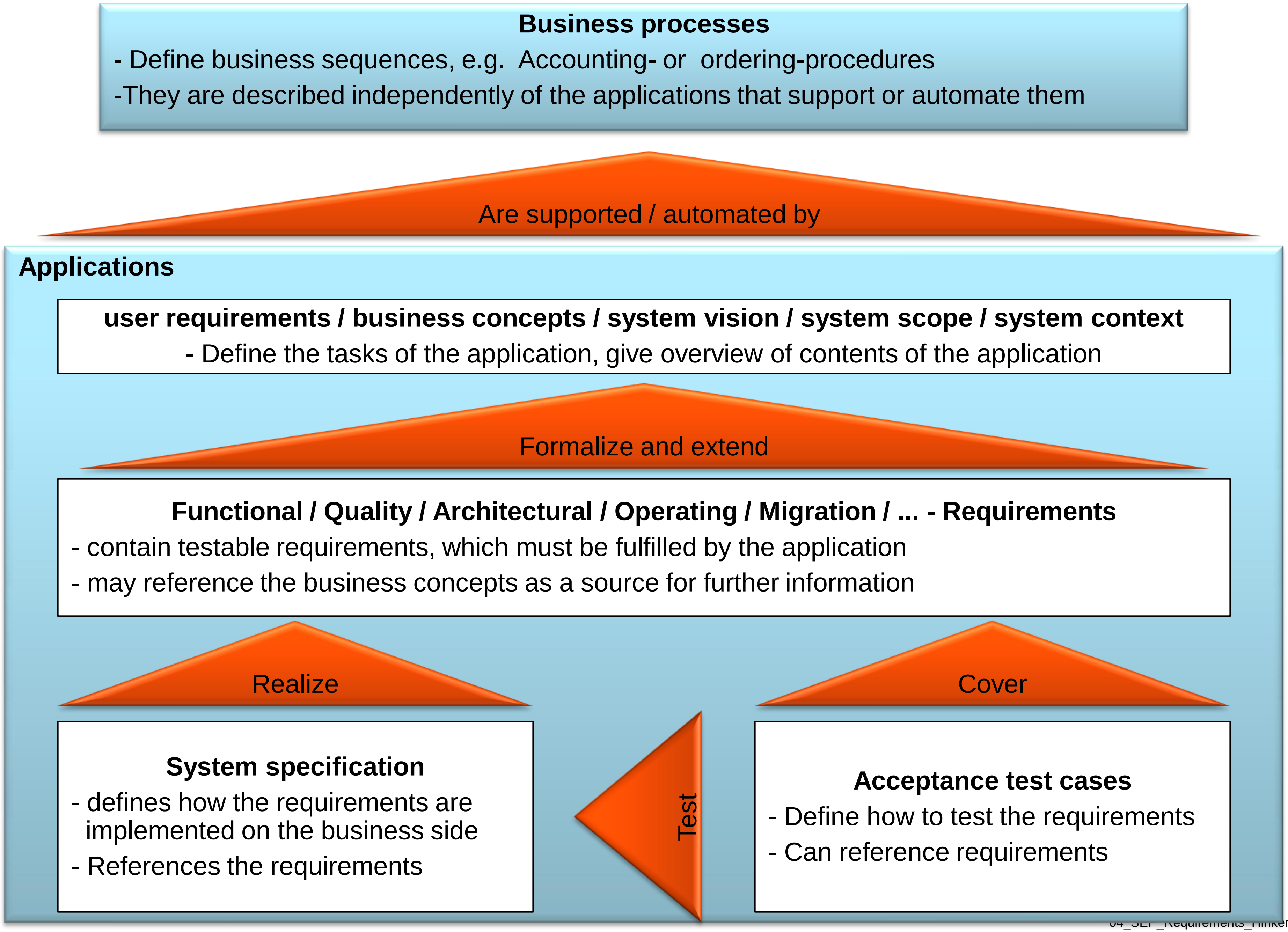
Cross sectional requirements

- Operating requirements
- Migration requirements
- Deployment requirements
- Organisational requirements

Types of requirements: cross sectional requirements

- **Operating requirements** define **operating features** of an application concerning categories like monitoring or configuration.
- **Migration requirements** are conditions as well as requirements for the replacement or adjustment of existing applications
- **Deployment requirements** are technical and organisational requirements concerning packaging and installation of the application.
- **Organisational requirements** usually refer to elements such as project structures, work environments or cooperation between vendor and client.





Business processes
- Define business sequences, e.g. Accounting- or ordering-procedures
-They are described independently of the applications that support or automate them

Applications

user requirements / business concepts / system vision / system scope / system context
- Define the tasks of the application, give overview of contents of the application

Functional / Quality / Architectural / Operating / Migration / ... - Requirements
- contain testable requirements, which must be fulfilled by the application
- may reference the business concepts as a source for further information

System specification
- defines how the requirements are implemented on the business side
- References the requirements

Acceptance test cases
- Define how to test the requirements
- Can reference requirements

AGENDA

- The term “requirement”
- **Process of requirements analysis**
- Examples of requirements
- Requirement Management in agile projects

The first step in Requirements Analysis: Identification of the stakeholders

Stakeholder

- Anyone with possible impact on the project
  - Team members, Sponsors, Business Partners
  - Customers, Vendors, Users, Administrators
  - Other Organisational Units, Managers

Stakeholder Register

- Contains important information about the stakeholder
- Contact information
- Role (Sponsor etc.)
- Category (supportive, passive, obstructive, ...)
- Expectations from project, information needs

Stakeholder identification

- Work iteratively, beginning by the sponsor
- Interview the stakeholders to identify new ones
- Store Stakeholder Information in Stakeholder register

Stakeholders provide the requirements of the project. Do not forget any of them:

- Missing requirements may cause massive problems later

- Missing involvement of stakeholders may cause problems during acceptance

Stakeholder Management is important during the whole duration of the project, not just during requirements analysis

The second step in Requirements Analysis: Developing the high level system scope

Define System Scope

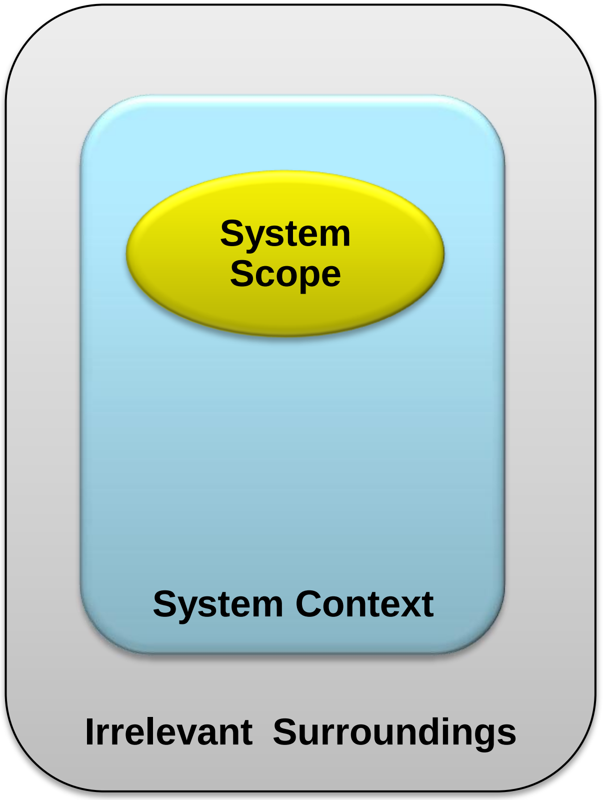
- System Scope contains anything that shall be done during the project
  - Applications, Documentation, ...
  - Define system scope with techniques like
    - Business Use Cases, Data Models (UML)
    - Definition of Data Sources, Data Sinks, ...

Define System Context

- System Context contains anything that
  - is not done in the project
  - but influences the project
- Examples: Neighbor systems, Users
- Necessary input for the requirements

Exclude everything else

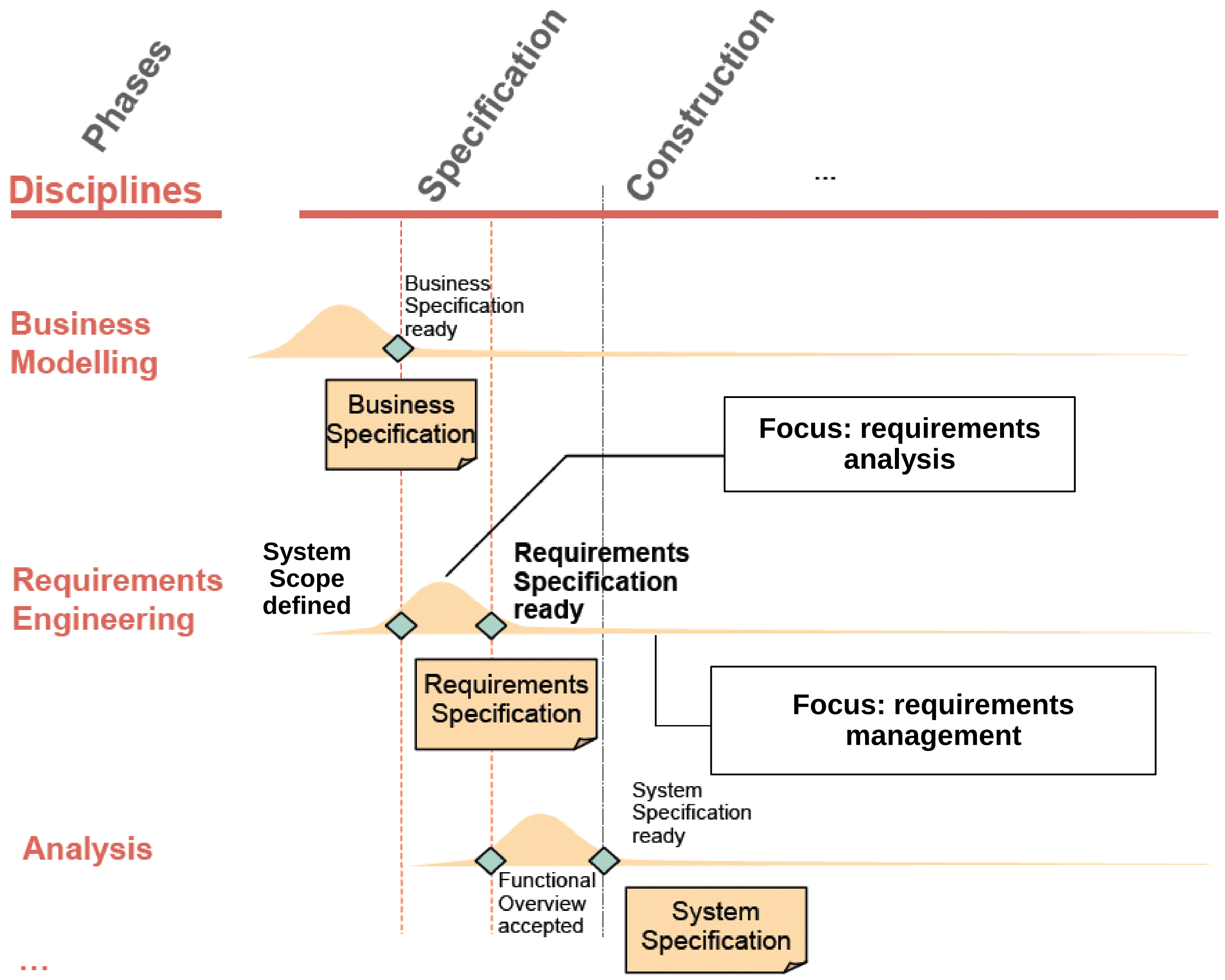
- Everything outside the system context is irrelevant for the requirements analysis
- There is no need to document or check anything in the irrelevant surroundings



The third step of Requirements analysis: Define, validate and prioritize requirements

This will be handled in detail in this lecture

Time of requirement engineering in a waterfall project


Phases
Specification
Construction
...
Disciplines
Business Modelling
Business Specification ready
Business Specification
Focus: requirements analysis
Requirements Engineering
System Scope defined
Requirements Specification ready
Requirements Specification
Focus: requirements management
Analysis
System Specification ready
Functional Overview accepted
System Specification
...

Examples of stakeholders involved in requirement analysis and why they need the requirements

Client

Business team

- Specification and software approval
- Basis for creation of test cases
- Business documentation

Business department

- Basis for check of the conformity to the business target architecture
- Business documentation of applications

Cross section Architecture division

Requirements

IT

Software implementation

- Contractual document
- Basis for verification of conformity to corporate standards
- Basis to examine the feasibility
- Basis for examine the completeness / correctness

Operating team

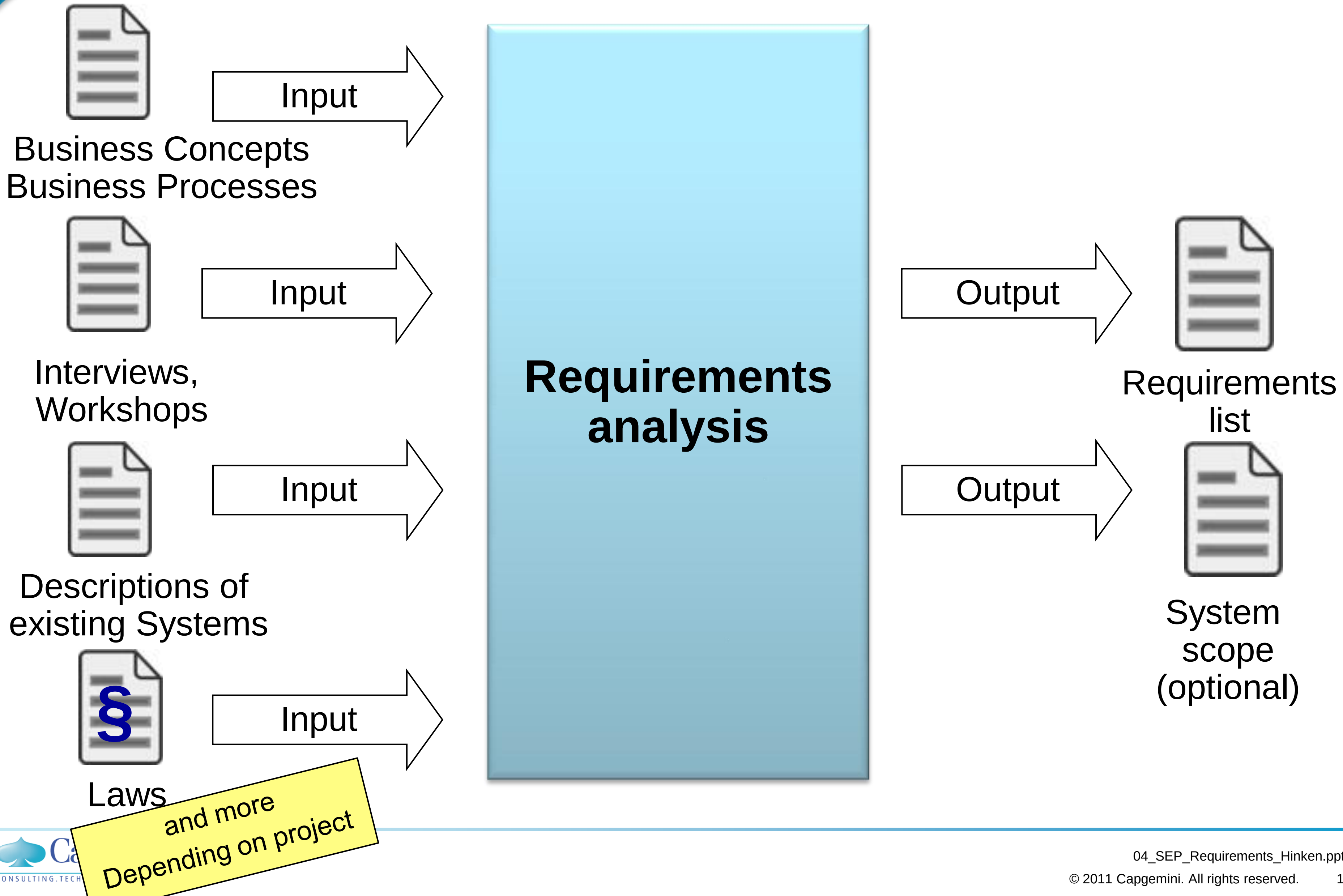
- Basis to examine the completeness of the operational requirements

Implementor

Implementor

- Basis for system specification, system design, implementation, test model

Process of requirements analysis



Requirements analysis:
Details about the definition of the system scope and system context

Analysis of already written business concepts

- These documents often provide a good overview over the application to build

- They contain detailed business information about the application
  - This information can be analyzed to define requirements and scope
  - Sometimes they contain definition of processes as “rough use cases”
  - These use cases can be referred to directly in the requirements

Ascertaining the information about the system scope / system context

- If the information is not already present in the business concepts

- By interviews and workshops with the stakeholders
  - starting out from existing business processes, other documents

- Goal: To ascertain
  - Global scope definition, Goals / rough contents of the system

Documentation of the system scope / system context

- There are no restrictions for the contents of this document
- The necessary formalities depend on the goal of the document

- The typical goal of the document is to
  - Provide the reader with overview concerning functionality, data, interfaces of the application
  - Define the goals of the application. Define exclusions from application scope
  - Define overview about the processes in the application

Requirements analysis
Details about the Definition of the requirements

Requirements are typically defined in Workshops with stakeholders:

- Sponsor, business department members
- project managers
- users
- Business analysts
- Technical staff, operations staff

In the workshops, requirements are defined on the basis of

- The business concepts
- The business processes
- the project plan (e.g. Requirements for migrations, parallel operations)
- Provisions of the customer (e.g. usage of V-Modell XT, defined milestones)

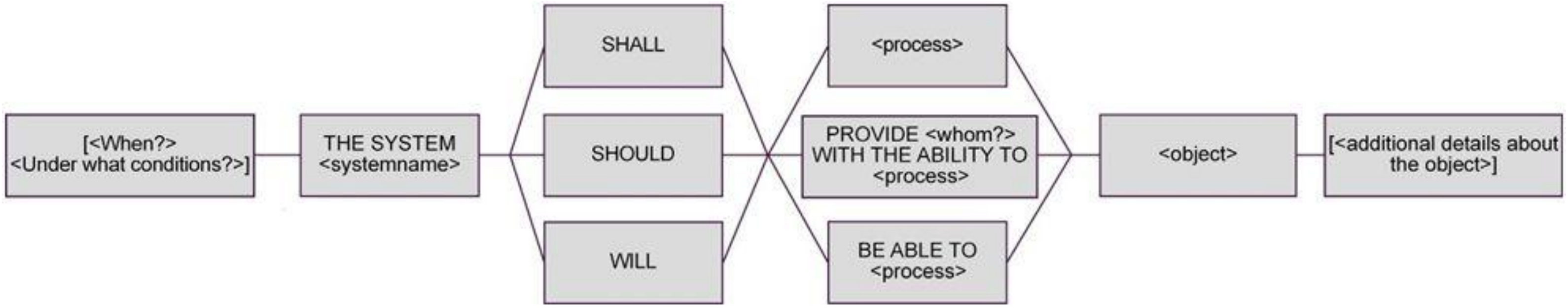
After each workshop, requirements are documented

- first at a rough level, then refined further and further
- important: Do not get lost in details. Helpful:
  - first set up a rough breakdown of the system as “business components”
  - then define sequences (rough use cases) to group requirements of these sequences
- Use templates and checklists to ensure the quality of the requirements (see later)

Utilities

- Marking of business concept content already specified as requirements
- Historicize requirements: Do this from the beginning
- Discuss the requirements very intensively during workshops
- Use tools like Excel, JIRA, ..

Excursion: requirements creation with SOPHIST template



The SOPHIST GmbH created an approach for evaluation of requirements.

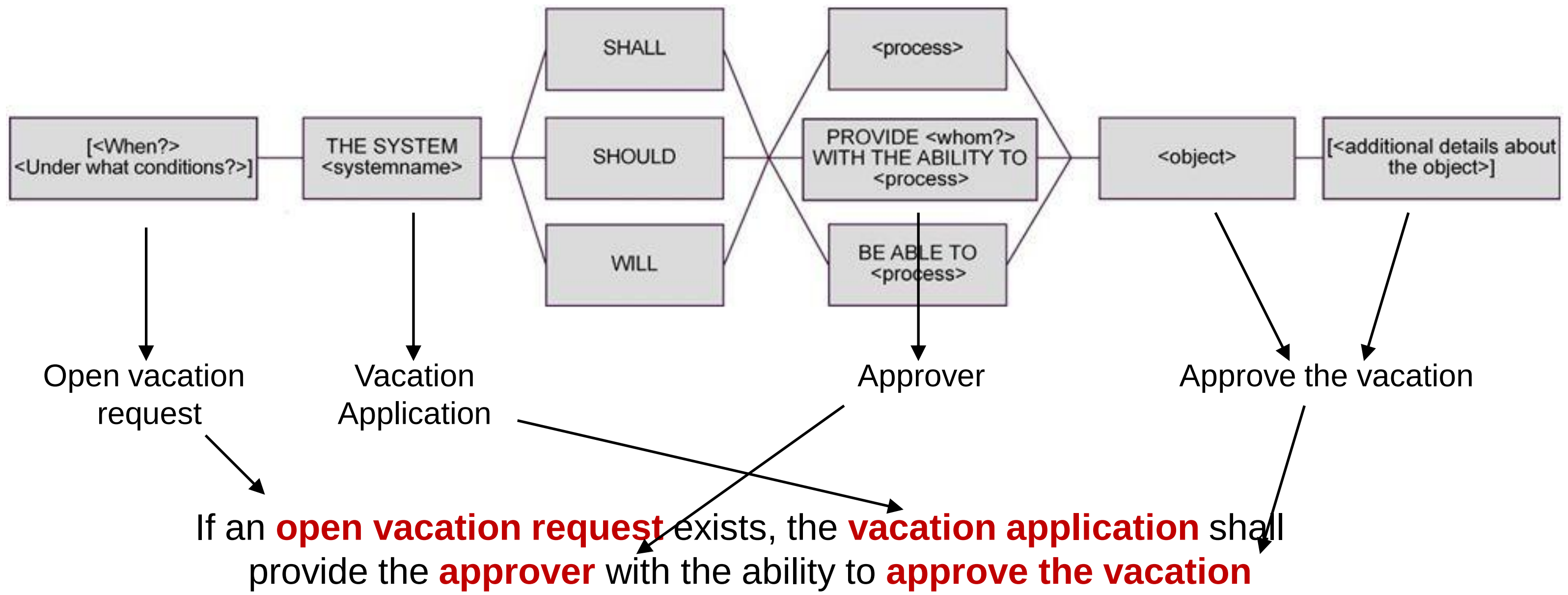
- Main areas are requirement definition using „natural language“.

The focus is thereby preventing transformations (distortion, falsification) of information in „natural language“ formulations.

- Their approach includes among other things a pattern.
- This should lead to requirements in which the actors involved, the conditions and the actions are always described as clearly as possible.

The goal of this approach is to create precise, focused and understandable requirements.

Example for using of SOPHIST pattern



If an **open vacation request** exists, the **vacation application** shall provide the **approver** with the ability to **approve the vacation**

The actors are called „service provider (Vacation Application)“ and „service recipient (Approver)“.

The condition is called „open vacation application“.

The activity is called „approve the vacation request“.

Sophist pattern is good method, if required level of detail in the requirements allows this:

- Since it allows only one sentence, analyst is encouraged to describe activity roughly (dangerous).
- Requirements, which describe internal logic, do not need actors.

Conclusion: Good approach, but not a dogma! Best Practice; use, if it works.

Requirements analysis: Details about requirement validation and prioritization

Step 1: Structuring of requirements

- according to application components
- according to use cases
- Cross section requirements according to categories

Step 2: Validation of each singular requirement

- Is the requirement complete?
- Is the requirement formulated precisely and consistently? Is the actor stated?
- Is the requirement implementable, testable?
- Which risk does the requirement present for the project?
- ...

Step 3: Validation of whole requirement set

- Is the number of requirements reasonable?
  - Rough rule: a thousand requirements are too many, less than a hundred too little
- Have all kinds of requirements been considered?
  - Quality requirements, operating requirements, migration requirements, ...?

Step 4: Prioritization of requirements

- Depending on business value / risk / dependencies

AGENDA

- The term “requirement”
- Process of requirements analysis
- **Examples of requirements**
- Requirement Management in agile projects

Example for requirement: content of a requirement

ID:	Unique identifier of the request
Type:	The type of requirement (see previous slides)
Description:	Description of a requirement should allow to objectively test the requirement.
Version:	For every change of a requirement a new version is created.
Source:	Information from which document, workshop etc. requirement comes.
Priority:	Importance of the requirement
Date of change:	Creation date of last requirement version.
Deleted:	Flag if the requirement is deleted.

gen am Beispiel

Bad Requirement Description
Descriptions of Quality Requirements have to allow for tests. A requirement for a whole year is not suitable for tests: This is an information, not a requirement. It should be .part of the business process.

Bad Requirement description
Phrases like „if possible“ make a requirement vague und do not support tests.
Such phrases should be averted.

Business use cases group requirements
In this requirement, we omit the process details: They are defined in the business concepts and referenced here.

ID / Name	Require-ment Type	Descriptio	Remarks / open issues
Request Management			
		Automati Handling of requests	
ANV020	Use Case	- The appl cation can automatically store the data of a req est in system X after receiving it from a data - The case worker can manually create a request in sys em X. - If necess ary, the case worker must be able to conta t the client sending the request in order to obtai concernin the request storage	
ANV030	Functional Requirement	**Time of R equest Storage** The Applic ation shall store the Request in S stem X immediately after receiving it. Before the request i stored, no further handling of the request ay happen.	q
ANV040	Quality Requirement	**The Application shall handle a bout 2 Million Requests per Year.**	CH 18.04 2011: This is an informatio , no testable requiremen Therefore, it is removed fro the Requirement
ANV050	Functional Requirement	**Automatic storage of requests** Requests should be automatically stored, if possible.	
UEB400	Operational Requirement	**Formulation of error messages** If an error occurs while calling another system, the error message shall contain the name of the other system and a descriptive name of the operation that was called. After that, the technical error message shall follow.	CH 18.04.11: Transferred from the system specification
UEB900	Architecture Requirement	**Conformity to reference architecture** This use case shall be implemented in compenent XYZ, compliant to the reference architecture.	
ANV080	Functional Requirement	**Decision about the automatic storage of a request** During request handling, the application searches for the person the request belongs to. Using the similarity between the person and the request, the application decides whether to automatically store the request.	The rules for the decision about the automatic storage are defined in the business concep
ANV090	Functional Requirement	**Process of the search for the person the request belongs to** In order to search for a person, the application uses the personal data of the request to search automatically in s stem X. The similarity between the hit and the request are used to decide, if an automatic storage of the request is po ble.	

Requirements and Specification
If new requirements come up during the detailed system specification,, they have to be appended here.
Otherwise, the requirement would not be used later on (for acceptance, tests etc.)

Unique IDs
These are necessary for the identifications of requirements.

Overlapping requirements
ANV080 and ANV090 describe the same requirement in two ways. These kinds of requirements have to be merged during the requirement verification.

AGENDA

- The term “requirement”
- Process of requirements analysis
- Examples of requirements
- **Requirement Management in agile projects**

Agile projects and traditional requirements engineering at first glance

Agile Projects	Projects using Requirements Engineering
• Quick start of development	• Heavyweight Phase before project
• Focusing on face-to-face collaboration	• Documentation oriented
• Cost-Effective, modern	• Old-fashioned, time-intensive

Requirements aspects in agile project (Example: SCRUM)

User Stories are Requirements and are the core of SCRUM

- User stories are comparable to “user requirements” (see slide 8)
- Defined using Workshops, Interviews, Brainstorming (requirements eng. techniques)
- Detailed and made testable before implementation

Stakeholders are involved

- At least Product owner and development team
- Stakeholders prioritize the user stories (requirements)

Stories are reviewed after every sprint

- By the stakeholders. However: After the implementation

Stories are documented

- In the sprint log, which is typically even versioned

- Requirements are gathered in agile projects
- They are gathered, validated, documented and tested in every sprint
- The agile methodologies do not focus on this
- This does NOT mean that requirements engineering should not be done
- Or that you should forget all methodology: Use what you have learned.

Differences of Requirements Engineering in agile projects

Amount of documentation

- That is up to you: Story Cards, Backlog, Diagrams on Whiteboards, small concept documents...
- Check what is best for the project
  - Long-term projects need better documentation

Level of detail

- Decide at the last moment:
  - User Stories will be less detailed in the beginning
  - Before Implementation, they get more detailed

Amount of work

- RE in agile projects may be more work
  - Work is done in iterations every sprint
  - No continuous engineering of requirements
- Stress during the whole project: Requirements discovery and validation typically done in parallel

The requirements engineering method is a tool

- To structure the discovery, analysis and documentation

It is no dogma: Especially concerning amount and documentation

- Choose what fits your purpose best

Capgemini

CONSULTING.TECHNOLOGY.OUTSOURCING

People matter, results count.

www.capgemini.com

Requirements management creates benefits

Benefits for all parties involved In the arrangement of the contents of the project with project team	• When generating comprehensive discussion on application • Same picture among project team and business team
Benefits for project team At the construction of the system specification	• Ensuring of the completeness • Simplification of the structure
Benefits for business team During the approval of the system spezifikation	• Good entry point • Supported by reference to Specification • Facilitates examination of the completeness
Benefits for all parties involved By change management during realisation and maintance	• Prioritizing is possible. • Easy to modify
Benefits for business team and IT Testcase specification and approval of the application	• Good entry point for structuring of the test cases • Ensure the coverage
Benefits for management **For larger changes in the application**	• Prioritization of the project content about the requirements
Benefits for all parties involved For langer changes in the application	• Dependencies between systems are visible • Changes to systems can be assessed in terms of cross-effects.

Excursion: Mapping of requirements to realisation costs

Costs for requirement group 1

Costs for requirement group 2

. . .

Costs for requirement group n

Business basic costs

–For cross sectional business functionality (e.g. cross sectional message creation)
–By business requirements influenced but not single groups assignable

Technical basic costs

–Influenced by technical requirements (e.g. NFA conserning availability)
–Influenced by organisational requirements (e.g. parallel operations)

The requirements do not segment these costs, but influence them as a whole.

Requirements are often only part of a unit estimable

Estimate possible at grouping of the requirements of the respectively assessable units.

Hereby only estimate for part of the efforts possible: Base efforts not included