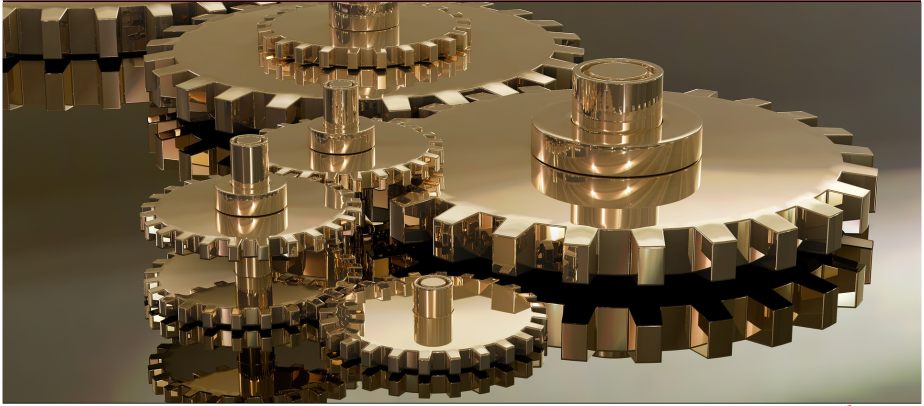
Software Engineering

Requirements Engineering

Prof. Dr. Reiner Hähnle
Fachgebiet Software Engineering



The CaSh Case Study

We use a case study to continuously illustrate the introduced concepts

The
Car Sharing
Company
Software

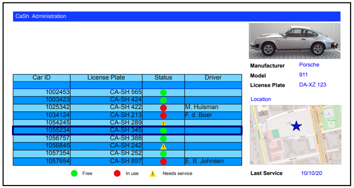
The CaSh Case Study

Description

Main Roles & Functionalities

- Role-independent
  - Authentication
- Administrator
  - Add/change new cars, rental locations
  - Billing
- User
  - Check availability
  - Request booking
  - Change booking
- Service Staff
  - Take out vehicle for service

The CaSh Case Study: Provision of services for different actors



CaSh Administration

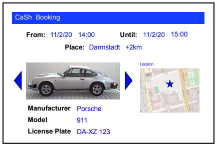
Car ID	License Plate	Status	Driver
1002453	CA-SH 565	●	
1003423	CA-SH 424	●	
1025342	CA-SH 422	●	M. Huisman
1034124	CA-SH 213	●	F. d. Boer
1054245	CA-SH 289	⚠	
1055234	CA-SH 345	●	
1056757	CA-SH 388	●	
1056845	CA-SH 242	⚠	
1057354	CA-SH 252	●	
1057654	CA-SH 897	●	E. B. Johnsen

● Free ● In use ⚠ Needs service

Manufacturer	Porsche
Model	911
License Plate	DA-XZ 123

Location

Last Service 10/10/20



CaSh Booking

From: 11/2/20 14:00 Until: 11/2/20 15:00

Place: Darmstadt +2km

Location

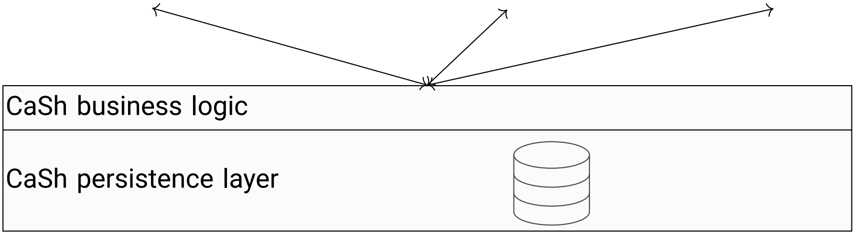
Manufacturer	Porsche
Model	911
License Plate	DA-XZ 123

• • •

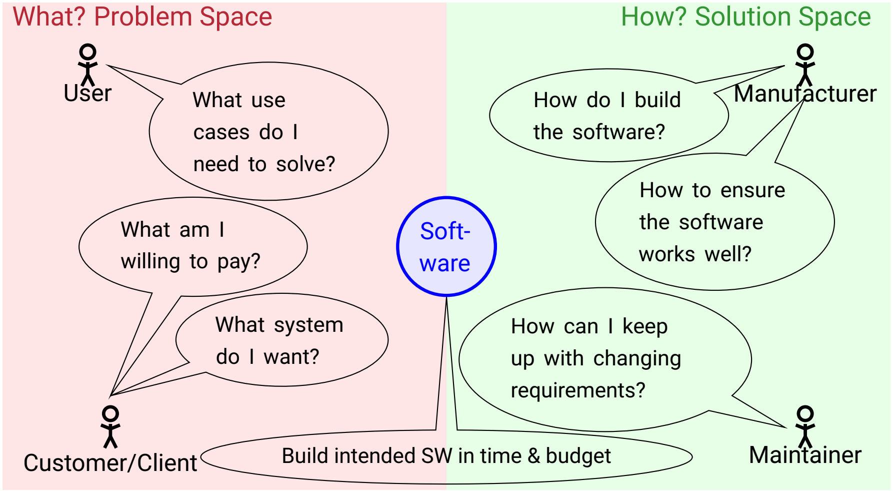
Adminstrative UI (Desktop)

End user UI (Mobile App, Web App)

Service UI (Tablet)

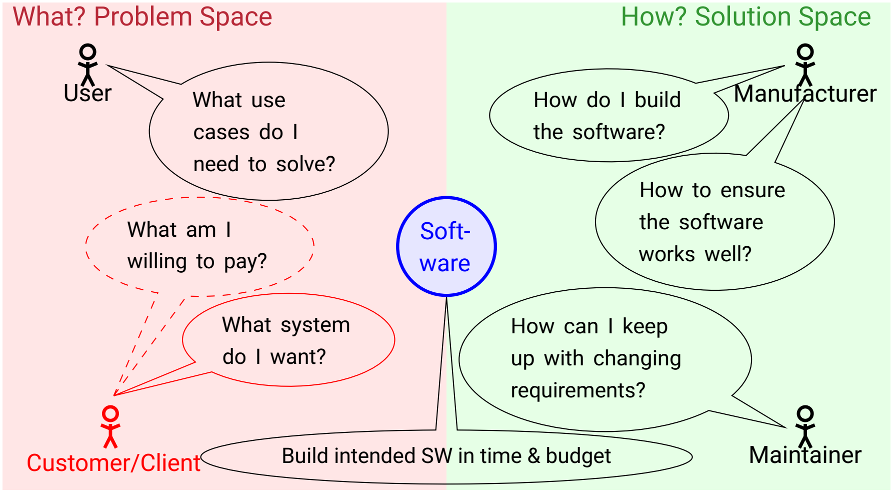


Requirements Engineering

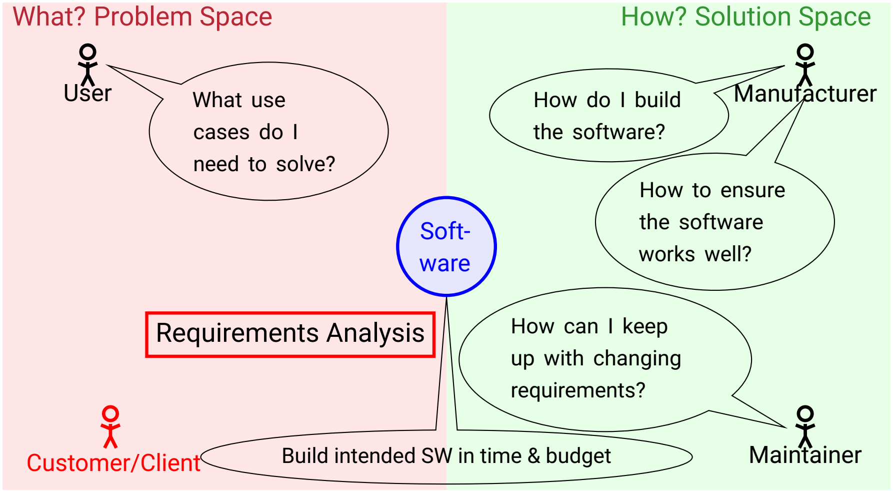


What? Problem Space
How? Solution Space
User
What use cases do I need to solve?
Manufacturer
How do I build the software?
How to ensure the software works well?
What am I willing to pay?
Soft-ware
What system do I want?
How can I keep up with changing requirements?
Customer/Client
Build intended SW in time & budget
Maintainer

Requirements Engineering


What? Problem Space
How? Solution Space
User
What use cases do I need to solve?
Manufacturer
How do I build the software?
How to ensure the software works well?
What am I willing to pay?
Soft-ware
What system do I want?
How can I keep up with changing requirements?
Customer/Client
Build intended SW in time & budget
Maintainer

Requirements Engineering


What? Problem Space
How? Solution Space
User
What use cases do I need to solve?
Manufacturer
How do I build the software?
How to ensure the software works well?
Soft-ware
Requirements Analysis
How can I keep up with changing requirements?
Customer/Client
Build intended SW in time & budget
Maintainer

Requirements Engineering
Objectives

Assume: You are contracted by client to develop CaSh

Requirements Engineering
Objectives

Assume: You are contracted by client to develop CaSh

Systematic approach to the question:
What has to be developed?

Requirements Engineering
Objectives

Assume: You are contracted by client to develop CaSh

Systematic approach to the question:
What has to be developed?

1. Understanding the problems in requirements elicitation
2. Different types of requirements
3. Requirement engineering workflow

4. Modeling and refining the requirements (**next lecture**)
   - Scenarios & Use cases
   - Notations: Textual & Graphical

Why Requirements Analysis is Difficult

Mary had a little lamb

What does the sentence mean?

Why Requirements Analysis is Difficult

Mary had a little lamb

What does the sentence mean?

Possible Meaning of “to have”

1. To hold in possession as a property
2. To trick or fool someone (been had by a someone)
3. To beget or bear (have a baby)
4. To partake (have as a meal)
5. . . .

Why Requirements Analysis is Difficult

Mary had a little lamb

What does the sentence mean?

Possible meaning of “lamb”

1. A young sheep less than one year
2. The young of various other animals (antelope, etc.)
3. A person as gentle or weak as a lamb
4. A gullible person easily cheated or deceived
5. The flesh of lamb used as food
6. . . .

Why Requirements Analysis is Difficult

Mary had a little lamb

What does the sentence mean?

Possible Meanings

have	lamb	meaning
1	1	Mary owned once a sheep under one year
3	2	Mary gave birth to an antelope
		. . .

Application to Case Study?

What is meant with . . . ?

- “The status of a car is in use, free or needs service.”

Application to Case Study?

What is meant with . . . ?

- “The status of a car is in use, free or needs service.”
  Inclusive or exclusive or?
- “Each car must have a unique license plate number.”

Application to Case Study?

What is meant with . . . ?

- “The status of a car is in use, free or needs service.”
  Inclusive or exclusive or?
- “Each car must have a unique license plate number.”
  What if the car has not yet been registered?
- “Das System muss sicher sein.”

Application to Case Study?

What is meant with …?

- “The status of a car is in use, free or needs service.”
  Inclusive or exclusive or?
- “Each car must have a unique license plate number.”
  What if the car has not yet been registered?
- “Das System muss sicher sein.”
  “Sicher” in welchem Sinne? Secure or safe?

Application to Case Study?

What is meant with . . . ?

- “The status of a car is in use, free or needs service.”
  Inclusive or exclusive or?
- “Each car must have a unique license plate number.”
  What if the car has not yet been registered?
- “Das System muss sicher sein.”
  “Sicher” in welchem Sinne? Secure or safe?
- Exercise: find ambiguous specifications in an API of your choice

What are Requirements?

Definition (Requirements)

Requirements are the descriptions of …

What are Requirements?

Definition (Requirements)

Requirements are the descriptions of . . .

- the services provided by the system and

Example (Provided Services by CaSh)

- Car booking
- Service booking
- Location tracking
- . . .

What are Requirements?

Definition (Requirements)

Requirements are the descriptions of …

- the services provided by the system and
- the operational constraints (Betriebsparameter)

Example (Operational constraints for CaSh)

- Database throughput (number of database queries per second)
- System memory
- Navigation systems GPS, Galileo, GLONASS
- …

What are Requirements?

Definition (Requirements)

Requirements are the descriptions of . . .

- the services provided by the system and
- the operational constraints (Betriebsparameter)

Requirements are written down

- in the System Requirements Specification (SRS) Document (German: “Pflichtenheft”)
- as user stories, structured natural language, use cases, state diagrams, . . . and stored in the product backlog (prioritized list of requirements)

What are Requirements?

Definition (Requirements)

Requirements are the descriptions of . . .

- the services provided by the system and
- the operational constraints (Betriebsparameter)

Requirements are written down

- in the System Requirements Specification (SRS) Document (German: “Pflichtenheft”)
- as user stories, structured natural language, use cases, state diagrams, . . . and stored in the product backlog (prioritized list of requirements)

Requirements are not solutions

Requirement Types

Many Different Types of Requirements Exist

- User requirements
- System requirements
- Functional requirements
- Non-functional requirements
- Domain requirements

Let’s discuss them in turn

Requirement Types
User Requirements

User Requirements

State in natural language and (semi-formal) diagrams:

- What are the services expected to be provided by the system
- What are the operational constraints

Often high-level and abstract descriptions (normally written by the customer)

Requirement Types
User Requirements

User Requirements

State in natural language and (semi-formal) diagrams:

- What are the services expected to be provided by the system
- What are the operational constraints

Often high-level and abstract descriptions (normally written by the customer)

Example

“CaSh shall keep track of all bookings as required by German law”
(in case of speeding tickets or accidents, for tax purposes, etc.)

Requirement Types
System Requirements

System Requirements (Systemanforderungen)

Precise and detailed specification of the system's

- functions and services,
- operational constraints

Requirement Types
System Requirements

System Requirements (Systemanforderungen)

Precise and detailed specification of the system’s

- functions and services,
- operational constraints

Example (CaSh System Requirements)

- “Upon successful completion of a booking, the user must be shown an overview of the booking details.”
- “Booking details must be stored by CaSh for 10 years from the booking date onward.”
- “Booking details consist of pickup and return date, car data (type, license plate number, . . .), user name + address, and payment information.”

Requirement Types

System Requirements

System Requirements (Systemanforderungen)

Precise and detailed specification of the system's

- functions and services,
- operational constraints

Characteristics of system requirements:

- Refinement of user requirements (as seen)
- Determine the system interface (functional, not technical interface)
  $\Rightarrow$ Demarcate the solution space
- Recorded as part of the system requirements document (functional specification) and typically part of the contract with client (Deutsch: "Grundlage für das Pflichtenheft")
- Authored by software developer or (better) business analyst in collaboration with client

Requirement Types
Functional Requirements

Functional requirement ($\neq$ functional specification)

Functionality that is clearly identifiable and localized in the code

Requirement Types
Functional Requirements

Functional requirement ($\neq$ functional specification)

Functionality that is clearly identifiable and localized in the code

Functional Requirements Specify . . .

- Services to be provided by the system , for instance:
  - Authentication
  - Searching for available cars
  - Sending confirmation emails

Requirement Types

Functional Requirements

Functional requirement ($\neq$ functional specification)

Functionality that is clearly identifiable and localized in the code

Functional Requirements Specify . . .

- Services to be provided by the system
- System reactions to specific inputs/events , for instance:
  - Error message when selected return date is before pick up date

Requirement Types
Functional Requirements

Functional requirement ($\neq$ functional specification)

Functionality that is clearly identifiable and localized in the code

Functional Requirements Specify . . .

- Services to be provided by the system
- System reactions to specific inputs/events
- System behavior in specific situations , for instance:
  - Network disruption during booking process

Requirement Types
Non-Functional Requirements

Non-functional Requirements (NFR)

Constraints on the services or functions offered by the system, including:

- Service level agreement (SLA)
- Constraints from development process (sequential/incremental)
- Alignment to standards (for example, protocols, laws)

Requirement Types
Non-Functional Requirements

Non-functional Requirements (NFR)

Constraints on the services or functions offered by the system, including:

- Service level agreement (SLA)
- Constraints from development process (sequential/incremental)
- Alignment to standards (for example, protocols, laws)

NFRs are often cross-cutting concerns that apply to the whole system

Meeting such requirements is usually impossible by adding a piece of code at a specific location or cannot be guaranteed by software alone

Requirement Types
Non-Functional Requirements

Non-functional Requirements (NFR)

Constraints on the services or functions offered by the system, including:

- Service level agreement (SLA)
- Constraints from development process (sequential/incremental)
- Alignment to standards (for example, protocols, laws)

Example (CaSh non-functional requirements)

- “The database must be able to process 100 queries per second”
- “User data must only be accessible to authorized persons”
- “The software development model must be CMMI Level 4 certified”
- “CaSh must provide the booking data for the accounting system”

Non-Functional Requirement Types

Non-functional requirements (Sommerville, Section 4.1.2)

Product requirements

- Reliability, availability
- Efficiency (performance, memory)
- Usability
- Portability

Non-Functional Requirement Types

Non-functional requirements (Sommerville, Section 4.1.2)

Product requirements

- Reliability, availability
- Efficiency (performance, memory)
- Usability
- Portability

Organisational requirements

- Delivery mode (beta, continuous, . . .)
- Implementation (programming language, frameworks, . . .)
- Standardization (ISO 9000, CMMI, . . .)

Non-Functional Requirement Types

Non-functional requirements (Sommerville, Section 4.1.2)

Product requirements

- Reliability, availability
- Efficiency (performance, memory)
- Usability
- Portability

Organisational requirements

- Delivery mode (beta, continuous, . . .)
- Implementation (programming language, frameworks, . . .)
- Standardization (ISO 9000, CMMI, . . .)

External requirements

- Interoperability (TUCaN–Moodle)
- Ethical aspects
- Legal aspects (safety, security, privacy, . . .)

Functional vs. Non-Functional Requirements

Observations

Non-functional requirements . . .

- may result in the identification of functional requirements

Functional vs. Non-Functional Requirements

Observations

Non-functional requirements . . .

- may result in the identification of functional requirements
- are often more mission-critical than individual functional requirements

Example (Relative importance of non-functional requirements)

A car sharing system that does not support to confirm a booking by email is still usable; if the system is not secure or reliable, it is worthless.

Ensuring Verifiability of Non-Functional Requirements

Formulate non-functional requirements so that they can be later verified

Example (Typical usability requirement)

“The user interface should be easy to use.”

Ensuring Verifiability of Non-Functional Requirements

Formulate non-functional requirements so that they can be later verified

Example (Typical usability requirement)

"The user interface should be easy to use."

Problems with this Requirement

Ensuring Verifiability of Non-Functional Requirements

Formulate non-functional requirements so that they can be later verified

Example (Typical usability requirement)

"The user interface should be easy to use."

Problems with this Requirement

- How to measure "easy"?

Ensuring Verifiability of Non-Functional Requirements

Formulate non-functional requirements so that they can be later verified

Example (Typical usability requirement)

"The user interface should be easy to use."

Problems with this Requirement

- How to measure "easy"?
- Easy for whom?
  - Expert user
  - Average user
  - Persons with a handicap (accessibility)

Ensuring Verifiability of Non-Functional Requirements

Formulate non-functional requirements so that they can be later verified

Example (Typical usability requirement)

“The user interface should be easy to use.”

Concretise Formulation into Quantifiable Requirements

- “Agents can assist in bookings and manage the car pool after one day of training”
- “An average end user can complete a booking in less than 5 minutes”
- “The user interface must be barrier free according to European law”

Requirement Types
Domain Requirements

Domain Requirements

Derived from application domain rather than from the needs of the user

- Expressed in domain-specific language and hard to understand by software engineers
- Often implicitly assumed as obvious to domain experts
- Functional or non-functional

Involvement of client is a must

Requirement Types
Domain Requirements

Domain Requirements

Derived from application domain rather than from the needs of the user

- Expressed in domain-specific language and hard to understand by software engineers
- Often implicitly assumed as obvious to domain experts
- Functional or non-functional

Involvement of client is a must

Example

Car sharing companies are required to check that new customers have a driving license.

How to Come Up with Requirements? Requirements Engineering Process

Requirements Engineering (RE)

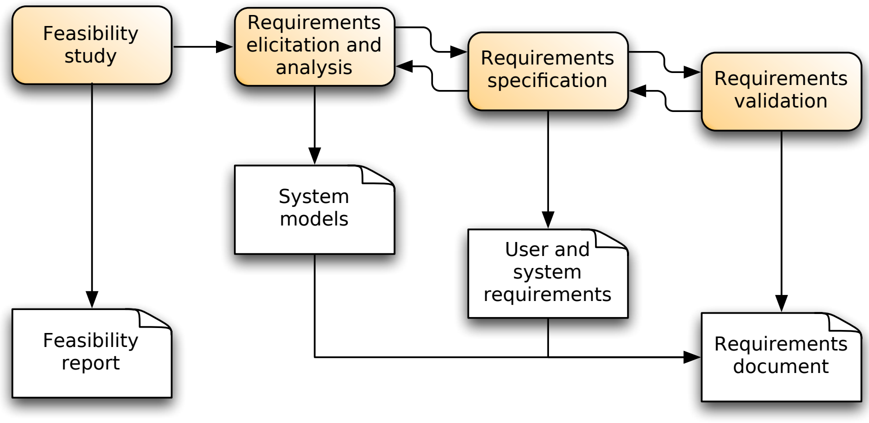
Requirements engineering is the process of

- finding,
- analyzing,
- documenting, and
- validating

software requirements

The system requirements document is created and maintained during RE

Requirements Engineering Process—Flow



(from: Sommerville, Software Engineering, Pearson)

RE Process

Feasibility Study (Machbarkeitsstudie)

Objective of a Feasibility Study

Obtain a justified recommendation whether the requirements engineering and system development process should be started (or continued) based on:

- Preliminary business requirements
- Outline description of the system
- Description of how the system is intended to support business processes

RE Process
Feasibility Study (Machbarkeitsstudie)

Objective of a Feasibility Study

Obtain a justified recommendation whether the requirements engineering and system development process should be started (or continued) based on:

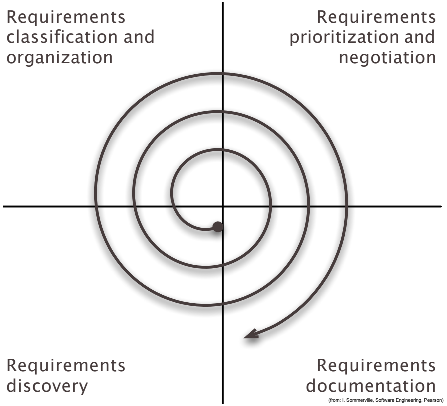
- Preliminary business requirements
- Outline description of the system
- Description of how the system is intended to support business processes

Feasibility Report

- Does the system contribute to the overall objectives of the organization?
- Can the system be implemented using current technology and within given cost and schedule constraints?
- Can the system be integrated with other systems used by the company?

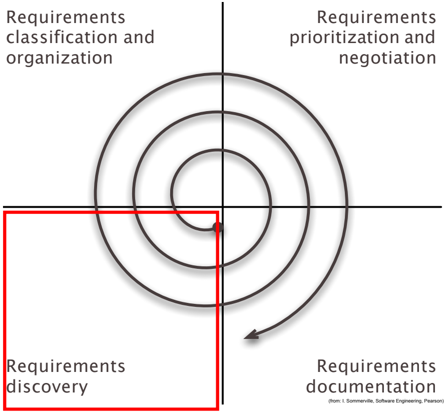
RE Process

Requirements Elicitation (dt: Erhebung) and Analysis


Requirements classification and organization
Requirements prioritization and negotiation
Requirements discovery
Requirements documentation
(from: I. Sommerville, Software Engineering, Pearson)

RE Process

Requirements Elicitation (dt: Erhebung) and Analysis



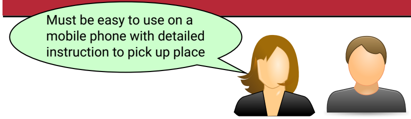
Requirements classification and organization
Requirements prioritization and negotiation
Requirements discovery
Requirements documentation
(from: I. Sommerville, Software Engineering, Pearson)

Systematic requirement discovery
Viewpoint-oriented Approach

Generic types of viewpoints

RE Process

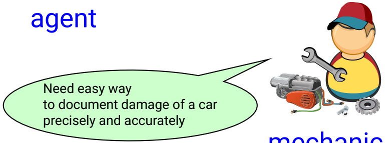
Requirements Elicitation (dt: Erhebung) and Analysis


Must be easy to use on a mobile phone with detailed instruction to pick up place

end users

Need quick, well-arranged and complete overview of all data of a customer

agent


Need easy way to document damage of a car precisely and accurately

mechanic

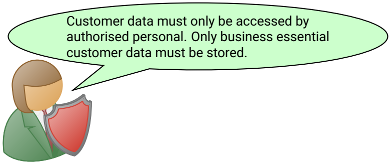
Systematic requirement discovery
Viewpoint-oriented Approach

Generic types of viewpoints

- Interaction viewpoints
  Persons (or systems) that will directly interact with the system such as end users, administrative & service personal
  —direct stakeholders—

RE Process

Requirements Elicitation (dt: Erhebung) and Analysis


Customer data must only be accessed by authorised personal. Only business essential customer data must be stored.

data protection officer

Systematic requirement discovery
Viewpoint-oriented Approach

Generic types of viewpoints

- Interaction viewpoints
  Persons (or systems) that will directly interact with the system such as end users, administrative & service personal
  —direct stakeholders—
- Indirect viewpoints
  Stakeholders that influence the requirements, but who will not directly use the system, e.g. CFO (finances), data protection official
  —indirect stakeholders—

RE Process

Requirements Elicitation (dt: Erhebung) and Analysis

DSGVO

Datenschutz-Grundverordnung

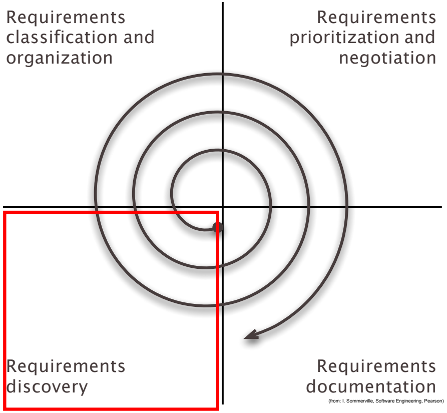
Systematic requirement discovery
Viewpoint-oriented Approach

Generic types of viewpoints

- Interaction viewpoints
  Persons (or systems) that will directly interact with the system such as end users, administrative & service personal
  —direct stakeholders—
- Indirect viewpoints
  Stakeholders that influence the requirements, but who will not directly use the system, e.g. CFO (finances), data protection official
  —indirect stakeholders—
- (Application) Domain viewpoints
  Domain characteristics & constraints that influence the system requirements
  E.g., legal regulations on booking details and storage duration

RE Process

Requirements Elicitation (dt: Erhebung) and Analysis



Requirements classification and organization
Requirements prioritization and negotiation
Requirements discovery
Requirements documentation
(from: I. Sommerville, Software Engineering, Pearson)

Systematic requirement discovery
Viewpoint-oriented Approach

- Develop more specific viewpoints during elicitation
- Use most important viewpoints to discover requirements

We want to build an app helping you in your daily work. May I ask you some questions?
Sure, I help you.

Systematic requirement elicitation
Techniques: Interviews

Closed interviews:
 Predefined set of questions

Open interviews:
 No predefined agenda

RE process

Requirements Elicitation and Analysis

We want to build an app helping you in your daily work. May I ask you some questions?
Sure, I help you.

Systematic requirement elicitation
Techniques: Interviews

Closed interviews:

Predefined set of questions

Open interviews:

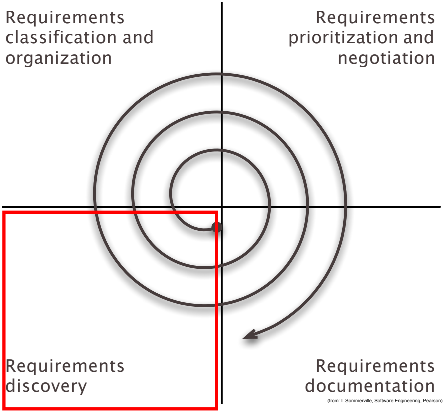
No predefined agenda

Limitation: Interviews should only be used as a supplement

- Bias of interviewee (e.g., afraid to loose job)
- Interviewee uses/assumes implicit domain knowledge

RE process

Requirements Elicitation and Analysis



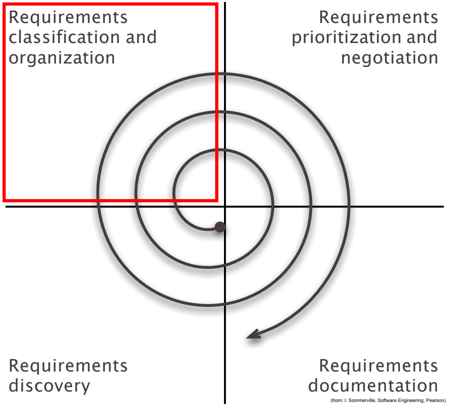
Requirements classification and organization
Requirements prioritization and negotiation
Requirements discovery
Requirements documentation
(from: I. Sommerville, Software Engineering, Pearson)

Systematic requirement elicitation
Further Techniques (next lecture)

Scenario	Sequence of interactions with system
Use Case	Related scenarios comprising a task

Requirements Elicitation and Analysis

Requirements Classification and Organization

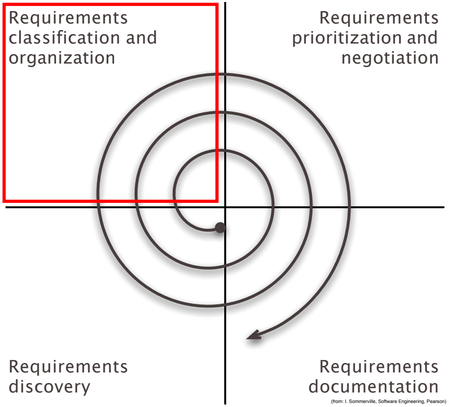

Requirements classification and organization
Requirements prioritization and negotiation
Requirements discovery
Requirements documentation
(from: I. Sommerville, Software Engineering, Pearson)

Classification & Organization

- Given: Unstructured collection of requirements
- Goal: Requirements grouped & organized into coherent clusters

Requirements Elicitation and Analysis

Requirements Classification and Organization



Requirements classification and organization
Requirements prioritization and negotiation
Requirements discovery
Requirements documentation
(from: I. Sommerville, Software Engineering, Pearson)

Classification & Organization

One possible model for categorizing requirements: The FURPS+ model

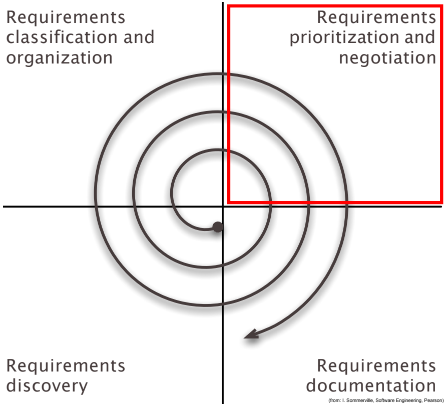
- Functional
- Usability
- Reliability
- Performance
- Supportability

+ Implementation

- Interface
- Operations
- Packaging
- Legal

Requirements Elicitation and Analysis

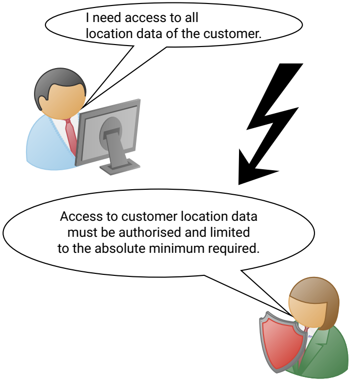
Requirements Prioritization and Negotiation


Requirements classification and organization
Requirements prioritization and negotiation
Requirements discovery
Requirements documentation
(from: I. Sommerville, Software Engineering, Pearson)

Prioritization & Negotiation

Requirements Elicitation and Analysis

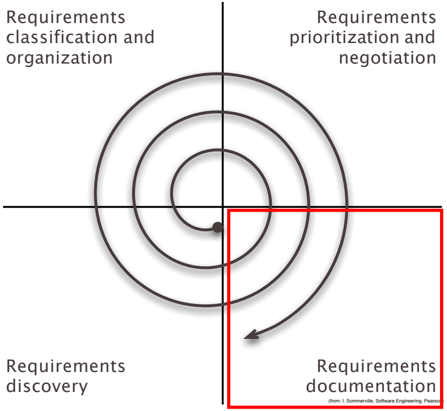
Requirements Prioritization and Negotiation


I need access to all location data of the customer.
Access to customer location data must be authorised and limited to the absolute minimum required.

Prioritization & Negotiation

- Requirements are prioritized
- Conflicts are found and resolved through negotiation

Requirements Elicitation and Analysis
Requirements Documentation


Requirements classification and organization
Requirements prioritization and negotiation
Requirements discovery
Requirements documentation
(from: I. Sommerville, Software Engineering, Pearson)

Documentation

The requirements are documented and used as input for the next iteration.

The produced documents may be formal or informal.

Requirements Documentation
Software Requirements Specification Document (SRS)

Target group of SRS

Diverse set of stakeholders:

- Client, prospective users
- Managers (both on client and manufacturer side)
- Developers, system test and system maintenance engineers
- ⇒ Anyone concerned with ordering, using, manufacturing, maintaining

Requirements Documentation
Software Requirements Specification Document (SRS)

Target group of SRS

Diverse set of stakeholders:

- Client, prospective users
- Managers (both on client and manufacturer side)
- Developers, system test and system maintenance engineers
- $\Rightarrow$ Anyone concerned with ordering, using, manufacturing, maintaining

Level of detail depends on …

- Type of system
- Development process
- Location of manufacture: external contractor or in-house

Requirements Documentation

Structure of the Software Requirement Document

Requirements specification document
(shortened from: ISO/IEC/IEEE 29148:2011)

1. Introduction
   a. Purpose of the requirements document
   b. Scope of the product
   c. Definitions, acronyms and abbreviations
   d. References
   e. Overview
2. General description
   a. Product perspective
   b. Product functions
   c. User characteristics
   d. Limitations
   e. Assumptions and dependencies
3. Specific requirements
4. Appendices, Index, etc.

Requirements Documentation

Structure of the Software Requirement Document

TECHNISCHE UNIVERSITÄT DARMSTADT

Requirements specification document
(shortened from: ISO/IEC/IEEE 29148:2011)

1. Introduction
    a. Purpose of the requirements document
    b. Scope of the product
    c. Definitions, acronyms and abbreviations
    d. References
    e. Overview
2. General description
    a. Product perspective
    b. Product functions
    c. User characteristics
    d. Limitations
    e. Assumptions and dependencies
3. Specific requirements
4. Appendices, Index, etc.

- 1.b Scope of product: Can also contain what is not in scope

Software Engineering Group

Requirements Documentation

Structure of the Software Requirement Document

Requirements specification document
(shortened from: ISO/IEC/IEEE 29148:2011)

1. Introduction
   a. Purpose of the requirements document
   b. Scope of the product
   c. Definitions, acronyms and abbreviations
   d. References
   e. Overview
2. General description
   a. Product perspective
   b. Product functions
   c. User characteristics
   d. Limitations
   e. Assumptions and dependencies
3. Specific requirements
4. Appendices, Index, etc.

- 1.b Scope of product: Can also contain what is not in scope
- 1.c Glossary: Explains ambiguous or technical terms, expand abbreviations

Requirements Engineering Process

Requirement Validation

Requirement Validation Checkpoints

Validity	Do the requirements capture the needed features? Is additional or other functionality needed?
Consistency	Check that the requirements are not conflicting
Completeness	Do the requirements define all functions and constraints as intended by the system user?
Realism	Can the requirements reasonably be implemented? (Refinement of feasibility study)
Verifiability	What are criteria when a requirement is considered fulfilled?
Traceability	Is each requirement traceable to its source (where does each requirement derive from)?

Literature

- Ian Sommerville, Software Engineering, 10th edition, Chapter 4, Pearson Education, 2018

  TUDa ULB eBook (German edition)
- Ulrike Hammerschall and Gerd Beneken, Software Requirements, Pearson, 2013

  TUDa ULB eBook (German edition)