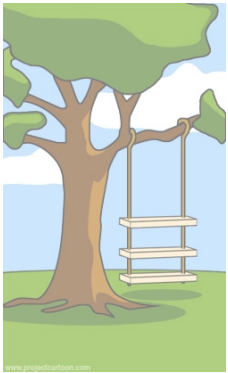
Software Engineering

2. Requirements | Thomas Thüm | October 27, 2021

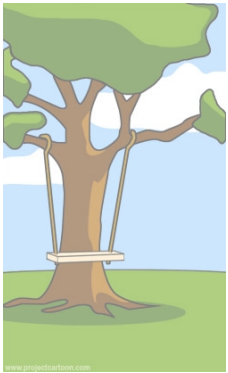
Software Engineering
Programming Languages

ulm university universität
uulm

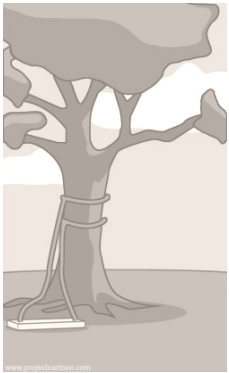
Software Engineering Projects



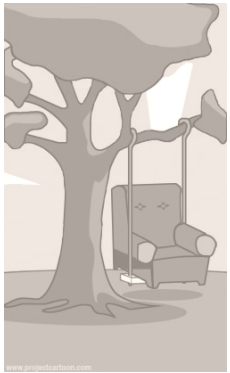
how the customer explained it



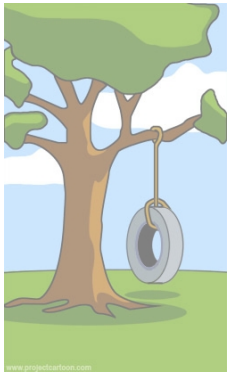
how the project leader understood it



how the programmer implemented it



how the business consultant described it



what the customer really needed

Lecture Overview

1. What are Requirements?
2. How to Elicit Good Requirements?
3. How to Document Requirements?

Lecture Contents

1. What are Requirements?
   - User Requirements
   - System Requirements
   - Functional Requirements
   - Non-Functional Requirements
   - Structure of a Requirements Document
   - Lessons Learned
2. How to Elicit Good Requirements?
3. How to Document Requirements?

User Requirements (Benutzeranforderungen)

User Requirement [Sommerville]

"User requirements are statements, in a natural language plus diagrams, of what services the system is expected to provide to system users and the constraints under which it must operate. The user requirements may vary from broad statements of the system features required to detailed, precise descriptions of the system functionality."

User Requirements Document (Lastenheft)

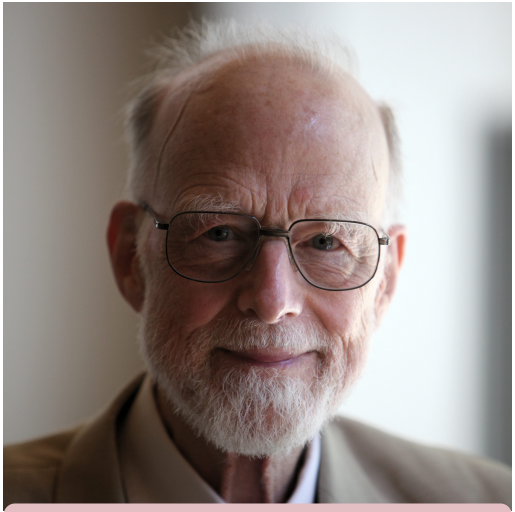
System description from user's point of view. What? For what? (Was? Wofür?)

Example

The Corona app should track contacts by means of random IDs and should store them locally for two weeks.

Example

If a user has a positive test result, he or she can inform tracked contacts about their increased risk by means of a QR code.



Tony Hoare (1969):

"The most important property of a program is whether it accomplishes the intention of its user."

System Requirements (Systemanforderungen)

System Requirement [adapted from Sommerville]

System requirements are more detailed descriptions of the software system's functions, services, and operational constraints. The system requirements document (aka. functional specification) should define exactly what is to be implemented. It may be part of the contract between the system buyer and the software developers.

System Requirements Document (Pflichtenheft)

System description from technical point of view. How? Whereby? (Wie? Womit?)

Example

If two devices are within a distance of 2m for at least 15 minutes they exchange their IDs via Bluetooth.

Example

After IDs have been exchanged, they are stored for two weeks.

Example

A new ID is generated every 24 hours and old IDs are stored for two weeks.

Functional Requirements

Functional Requirement [Sommerville]

"Functional requirements are statements of services the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations. In some cases, the functional requirements may also explicitly state what the system should not do."

Non-Functional Requirements

Non-Functional Requirement [Sommerville]

"Non-functional requirements are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process, and constraints imposed by standards. Non-functional requirements often apply to the system as a whole rather than individual system features or services."

Note

often more critical than functional requirements

Example

The app consumes less than 10MB of RAM.

Example

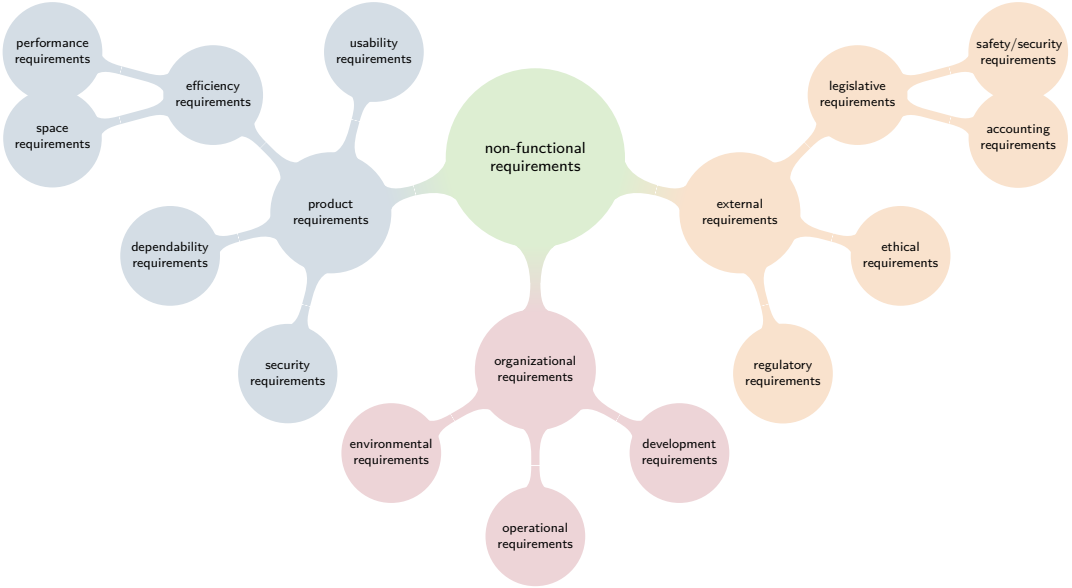
The app has no access to private data of the user.

Example

The app conforms to the GDPR. (DSGVO)

Example

The source code of the app is open source.


non-functional requirements
product requirements
performance requirements
space requirements
efficiency requirements
usability requirements
dependability requirements
security requirements
organizational requirements
environmental requirements
development requirements
operational requirements
external requirements
legislative requirements
safety/security requirements
accounting requirements
ethical requirements
regulatory requirements

Structure of a Requirements Document

Typical Structure [Sommerville]

Preface	expected readers, version history
Introduction	motivation/needs, collaboration with other system, strategic objectives
Glossary	technical terms
User Requirements Definition	functional and non-functional user requirements
System Architecture	distribution of functions across system modules, potential reuse
System Requirements Specification	detailed requirements, interfaces to other systems
System Models	graphical models illustrating the system with its environment
System Evolution	anticipated changes due to hardware evolution or changing needs
Appendices	hardware and database requirements, minimal/optimal system configuration
Index	index of diagrams/functions/terms/...

What are Requirements?

Lessons Learned

- What kind of requirements exist and why?
- User and System Requirements Document (Lasten- und Pflichtenheft)
- Further Reading: Sommerville, Chapter 2.2.1 (p. 54f), Chapter 4.1 (p. 101–111), Chapter 4.4.4 (p. 126–128)

Practice

- See Moodle
- Give examples for functional, product, organizational, and external requirements
- Classify requirements of colleagues

Lecture Contents

1. What are Requirements?

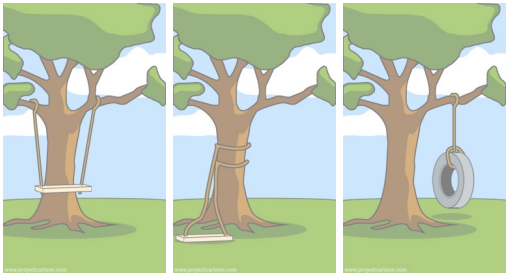
2. How to Elicit Good Requirements?
   - Imprecise Requirements
   - Complete and Consistent Requirements
   - Requirements Engineering Process
   - Why is Requirements Elicitation so Hard?
   - Example Dialog by Ludewig and Lichter
   - Requirements Elicitation Techniques
   - Requirements Validation
   - Lessons Learned

3. How to Document Requirements?

Imprecise Requirements

Sommerville:

"Imprecision in the requirements specification can lead to disputes between customers and software developers. It is natural for a system developer to interpret an ambiguous requirement in a way that simplifies its implementation. Often, however, this is not what the customer wants. New requirements have to be established and changes made to the system. Of course, this delays system delivery and increases costs."



how the project leader understood it | how the programmer implemented it | what the customer really needed

Complete and Consistent Requirements

Completeness and Consistency [Sommerville]

"Ideally, the functional requirements specification of a system should be both complete and consistent. **Completeness** means that all services and information required by the user should be defined. **Consistency** means that requirements should not be contradictory."

Further Desired Properties

clear, easy to understand, unambiguous, correct, verifiable, prioritized, changeable, traceable

Often not Feasible in Practice

mistakes, omission, implicit knowledge, many stakeholders (Akteure) with different backgrounds / expectations / inconsistent needs

Fred Brooks (1987):

"Much of the essence of building a program is in fact the debugging of the specification."

Requirements Engineering Process

Requirements Elicitation and Analysis

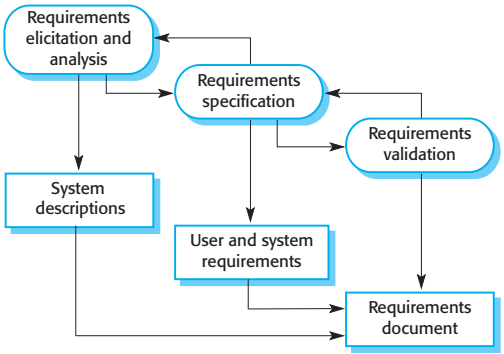
Requirements elicitation and analysis is the process of deriving the system requirements through observation of existing systems, discussions with potential users, or development of prototypes. (Anforderungsermittlung und -analyse) [adapted from Sommerville]

Requirements Specification [Sommerville]

"Requirements specification is the process of writing down the user and system requirements in a requirements document (aka. requirements specification)." (Anforderungsspezifikation)

Requirements Validation [Sommerville]

"Requirements validation is an activity that checks the requirements for realism, consistency, and completeness." (Anforderungsvalidierung)



Why is Requirements Elicitation so Hard?

- Stakeholders have difficulties to articulate what they want
- Stakeholders don't know what is (not) feasible
- Implicit knowledge and jargon in the customer's domain
- Dynamic business environment (e.g., new stakeholders)

Example Dialog by Ludewig and Lichter

- At the morning you unlock the door at the main entrance?
- Every morning?
- Even at the weekend?
- And during the plant shutdown?
- And if you are sick or in holidays?
- And if Mr. X is off?
- What does “morning” mean?
- Yes, as I said.
- Of course.
- No, at weekend the entrance remains closed.
- Clearly, it is then closed as well.
- Mr. X opens the door in this case.
- Then, a client knocks at the window to tell that the door is closed.
- ...

Source: Ludewig and Lichter

Requirements Elicitation Techniques

- Open interviews: talk to users what they do
- Closed interviews: stakeholders answer predefined questions
- Ethnography: observation of stakeholder's work
- Prototyping
- Feedback loops

Sommerville:

"You need to spend time understanding how people work, what they produce, how they use other systems, and how they may need to change to accommodate a new system."

In Practice

Combinations of numerous techniques used

Requirements Validation

Motivation: the later problems with requirements are detected, the more costly it will be

Validity checks	do requirements (still) reflect real needs?
Consistency checks	are there contradictory/redundant requirements?
Completeness checks	are all functions and constrains documented?
Realism checks	is it feasible within budget and schedule?
Verifiability	can we test the requirements?

Checked by reviews, prototyping, and test-case creation

How to Elicit Good Requirements?

Lessons Learned

- What are desired and undesired properties for requirements?
- What is the requirements engineering process?
- What are techniques for requirements elicitation and why is it hard?
- Further Reading: Sommerville, Chapter 4.2–4.3 (p. 111–119), Chapter 4.5 (p. 129f)

Practice

- See Moodle
- Give example for a bad requirement
- Improve a requirement of a colleague

Lecture Contents

1. What are Requirements?
2. How to Elicit Good Requirements?
3. How to Document Requirements?
   - Natural Language Specification
   - Structured Specifications
   - CoronaWarnApp User Stories
   - Use Case Diagrams
   - Include and Extend Relationships
   - Lessons Learned

Natural Language Specification

- Used since 1950s
- Pros: expressive, intuitive, universal
- Cons: vague, ambiguous, interpretation depends on reader's background

Style Guidelines

- one or two short sentences of natural language
- one message per sentence
- use active voice (e.g., "the system . . . ")
- consistent use of language (i.e., avoid synonyms)
- use shall for mandatory and should for desirable requirements
- use text highlighting
- avoid jargon, abbreviations, acronyms
- provide a rationale

Structured Specifications

- Use of templates rather than free-form text
- Distinguishes between function, description, input (origin), output (destination), pre- and postconditions, side effects, rationale, dependencies to other requirements, . . . or any subset thereof
- Pros: same as before
- Cons: similar as before, but less variability

Example

Function	Exchange of IDs
Description	Two devices send their IDs to each other.
Precondition	Devices have been within a distance of 2m for at least 15 minutes.
Postcondition	IDs are stored in the local database.
Rationale	Contact tracing requires to temporarily identify the device of contacts.

CoronaWarnApp User Stories

Show the number of stored keys in the local database #65

Closed **asdil12** opened this issue on May 31 · 1 comment

asdil12 commented on May 31

Feature description

The app receives keys from other smartphones via bluetooth and stores the keys for the last 14 days (IIRC) in the local database. Even though the app doesn't know if multiple keys in the local db correspond to one single person in real life, it would be nice to have this number as it could be used as a rough indicator if the app is working at all and give some transparency how well it is performing.

Also it could be helpful by providing some feedback to the user how well he is keeping distance to other people eg. by looking at the counter before and after going to the supermarket.

The counter could be put into a debug submenu to prevent confusing users with less technical knowledge.

Problem and motivation

As a technically interested person I would like to gain some insignts und transparency how the app is performing.

Is this something you're interested in working on

No

CoronaWarnApp User Stories

Enhance Logging With Timber #24

Closed **Magoli1** opened this issue on May 30 · 1 comment

Magoli1 commented on May 30

Current Implementation

Currently the app uses the native android logging framework. To do that, every class holds a reference to a dedicated logging tag, which is given the logger, in case it is used. Example:

cwa-app-android/Corona-Warn-App/src/main/java/de/rki/coronawarnapp/receiver/ExposureStateUpdateReceiver.kt
Line 38 in 43412bd

```
38    private val TAG: String? = ExposureStateUpdateReceiver::class.simpleName
```

Suggested Enhancement

There is a leightweight library called Timber, which automatically sets the logging tags to the calling class. We suggest to change the implementation for all logging calls towards Timber.

Expected Benefits

Timber optimizes the developer experience inseveral cases, eg. it automatically makes sure that the tag is at most 23 characters long (more details here). We could alsoget rid of the tag in every class that is currently hardcoded.

CoronaWarnApp User Stories

Do not log in production #235

Closed **IndianaDschones** opened this issue on Jun 7 · 4 comments

IndianaDschones commented on Jun 7 — Contributor

Current Implementation

The `release` build contains `Log` -statements of all levels.

Suggested Enhancement

A production build should not contain any `Log` -statements, see for example the Android Prepare for release guide.

`Log` statements can be removed using ProGuard rules.

Expected Benefits

- Compliance to Androids suggestions for `release` builds
- Security: logs can contain sensitive data which should never be logged
- Behaves like the iOS application will do, see corona-warn-app/cwa-app-ios#22

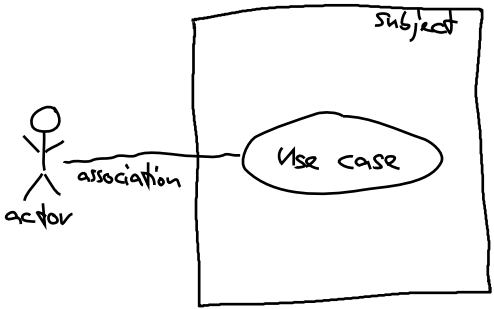
Use Case Diagrams (since 1993)

Use Case Diagram (Anwendungsfalldiagramm)

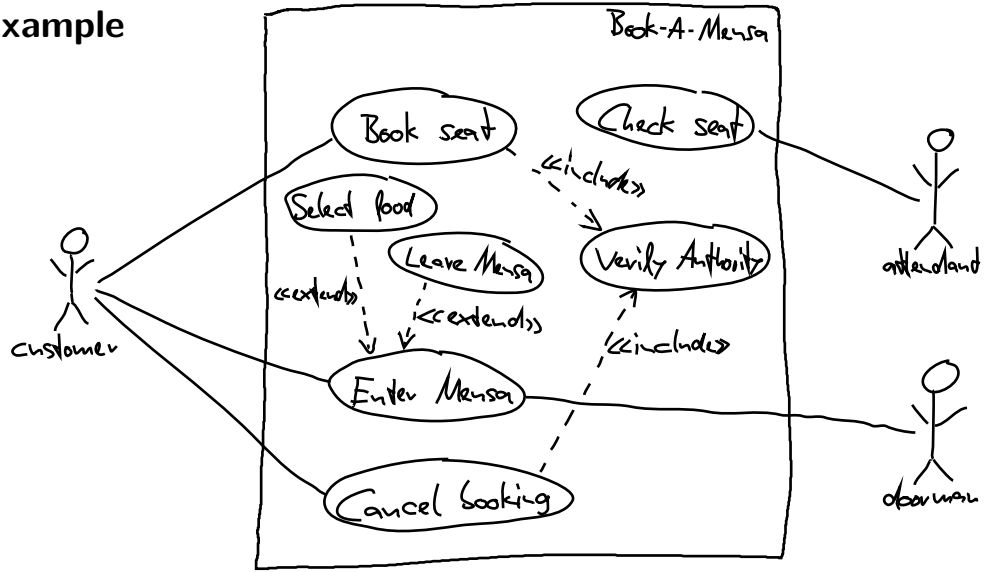
Use case diagrams are a means to capture the requirements of systems, i.e., what systems are supposed to do. The key concepts specified in this diagram are actors, use cases, and subjects:

- Each **subject** represents a system under consideration to which the use case applies. (System)
- Each users and any other system that may interact with a subject is represented as an **actor**. (Akteur)
- A **use case** is a specification of behavior in terms of verb and noun. (Anwendungsfall)

[adapted from UML 2.5.1]



Example


Book-A-Mensa
Book seat
Check seat
Select food
Leave Mensa
Verify Authority
Enter Mensa
Cancel booking
«include»
«extend»
«extend»
«include»
customer
attendant
observer

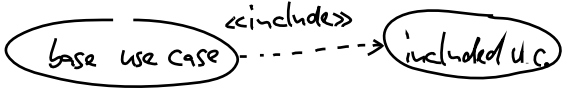
Include and Extend Relationships

Include Relationship

Motivation: make common parts of multiple use cases explicit

Relationship: a **base use case** may define an include relationship to an **included use case**

Meaning: included use case is always executed when the base use case is

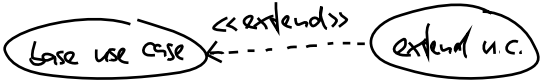


Extend Relationship

Motivation: make explicit that some use cases only happen under certain circumstances

Relationship: an **extend use case** may define an extend relationship to a **base use case**

Meaning: when the base use case is executed, the extend use case may or may not be executed



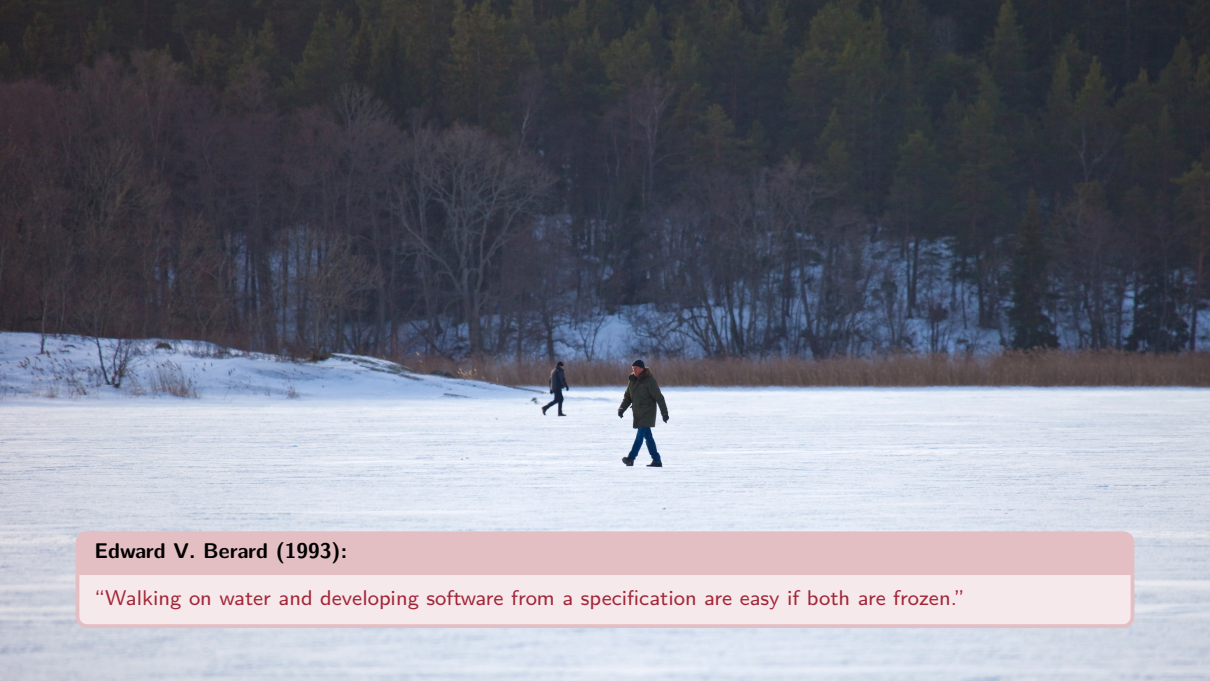
How to Document Requirements?

Lessons Learned

- Natural language and structured specification of requirements
- Graphical specification with use case diagrams (Anwendungsfalldiagramme)
- Further Reading: Sommerville, Chapter 4.4 (p. 120–126) and Chapter 5.2.1 (p. 144–146)
- Further Reading: UML 2.5.1, Chapter 18
- Next: How to model the system behavior in more detail?

Practice

- See Moodle
- Draw a use case diagram related to your requirements and upload it to Moodle
- Optional: watch a great tutorial on use case diagrams: `https://www.youtube.com/watch?v=zid-MVo7M-E` (contains 15s advertisement for a tool) (generalization and extension points are not needed in this course, can stop at 10:50)



Edward V. Berard (1993):

"Walking on water and developing software from a specification are easy if both are frozen."