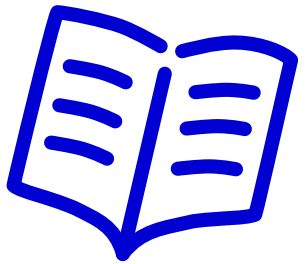
Requirements Engineering I

Chapter 3

Documenting Requirements



Chapter roadmap

3.1 Work products
The results of RE work

3.2 Classic specs
Classic requirements documentation

3.3 RE in agile
Agile work products

3.4 Glossary
Why terminology matters

3.5 Prototypes
What you see is what you get

3.6 Aspects to be documented
More than just functionality

3.7 How to document
Standards, templates, formality, precision, detail, depth

3.8 Quality
of requirements and work products

Motivation

Bridging the gap:

Stakeholders

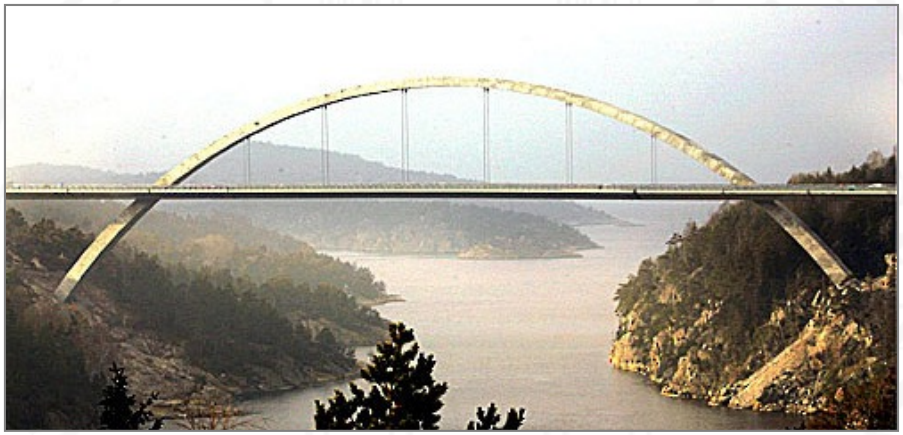


Photo © Lise Aserud / DPA

System builders

The need:

- Communicating requirements
- Having a basis for contracts and acceptance decisions

The means: Documented requirements

3.1 Requirements Engineering work products

DEFINITION. Work product – A recorded, intermediate or final result of information generated in a work process.
Synonym: artifact

Work products are characterized by their

- Purpose
- Representation (free text, structured text, lists, graphics, drawings,...)
- Size (single requirements, sets of requirements, documents (or document-like structures)
- Lifespan (temporary, evolving, durable)

Note that a work product may contain other work products

Work products and their purposes

Single requirements

- Sentence in natural language – expressing an individual requirement

> The control system shall prevent engine overspeed.

- User story – specifying a function or behavior from a stakeholder's perspective

> As a skier, I want to pass the chairlift gate so that I get access without presenting, scanning or inserting a ticket at the gate.

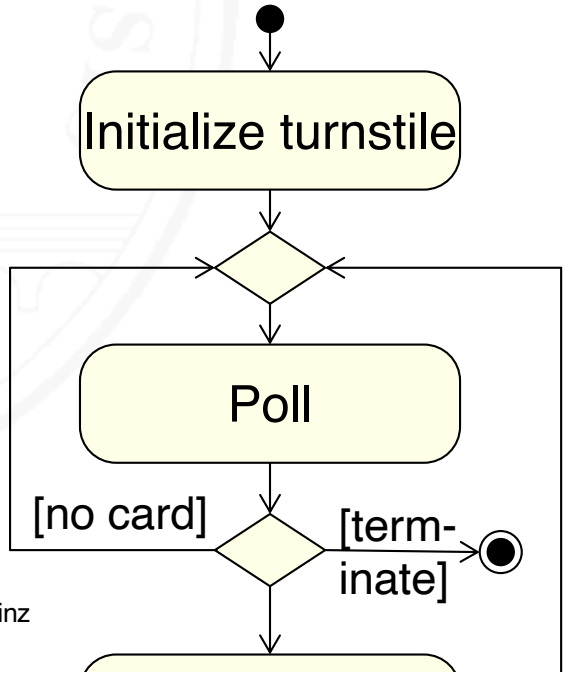
Work products and their purposes – 2

Sets of requirements

- Use case – specifying a system function from a stakeholder's perspective

USE CASE SetTurnstiles
Actor: Service Employee
Precondition: none
Normal flow:
1 Service Employee chooses turnstile setup. System displays controllable turnstiles: locked in red, normal in green, open in yello
2 Service Employee selects turnstiles s/he wants to modify. (...)

- Graphic model – specifying various aspects, e.g., context, activity, behavior



Work products and their purposes – 3

Sets of requirements

- Task description – specifying a task to perform

[Lauesen 2002]

Task:	**2.1 Record time actually worked**
Goal:	Provide basis for calculation and flex time
Frequency:	Most work is done according to the official plan. the plan may happen very two weeks for each p
Critical:	Nothing really.
Sub-tasks:	**Example of solution:**
1 Record work time (...)	Staff record deviating w online roster. Alternative

- External interface – specifying the information exchanged between a system and an actor in the system context

Desired_speed	IN	Integer (16 Bit)
Pedal_override	IN	Flag
Cruise_on	OUT	Flag

Work products and their purposes – 4

- Epic – providing a high-level view of a stakeholder need

  Let users create and manage their profile themselves.

- Feature – A distinguishing characteristic of a system that provides value for stakeholders

  Order history is downloadable by subscribers.

Work products and their purposes – 5

Documents and document-like structures

- System requirements specification,
  business requirements specification,
  stakeholder or user requirements specification
  – providing a baselined or released requirements document

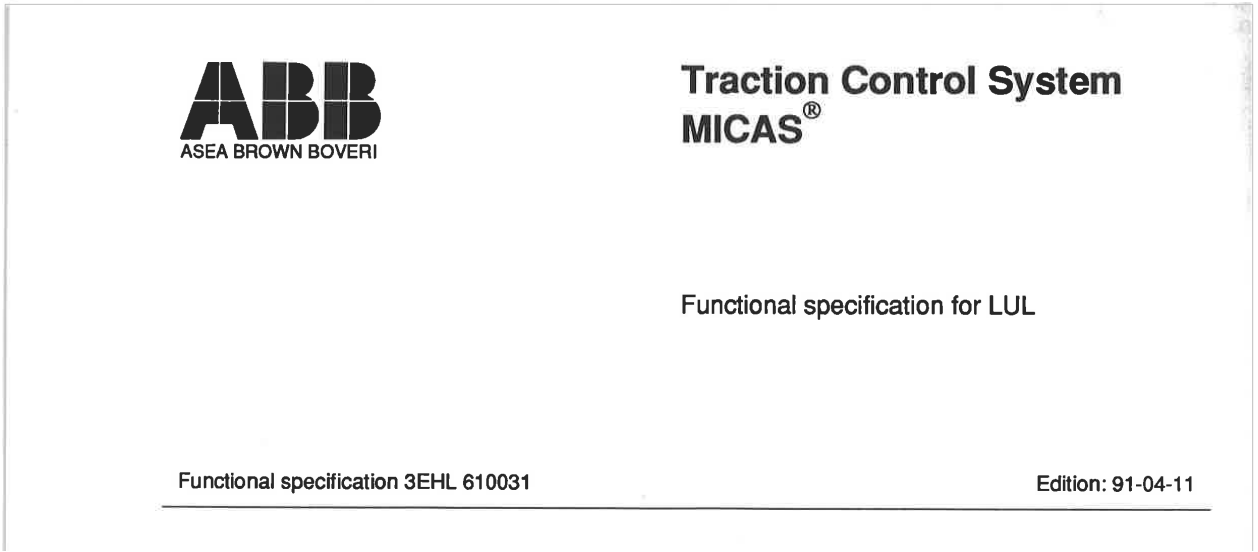


ABB
ASEA BROWN BOVERI

Traction Control System
MICAS®

Functional specification for LUL

Functional specification 3EHL 610031

Edition: 91-04-11

Work products and their purposes – 6

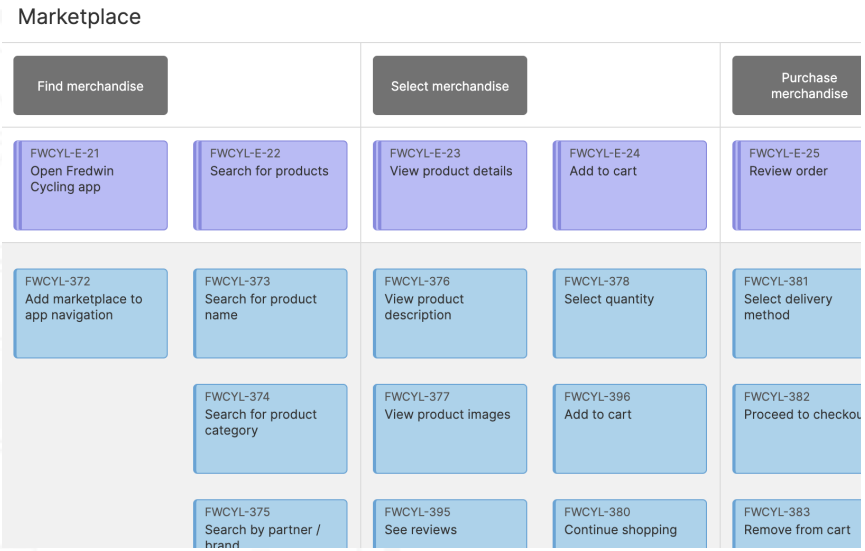
Documents and document-like structures

- Product and sprint backlog – managing a list of work items, including requirements

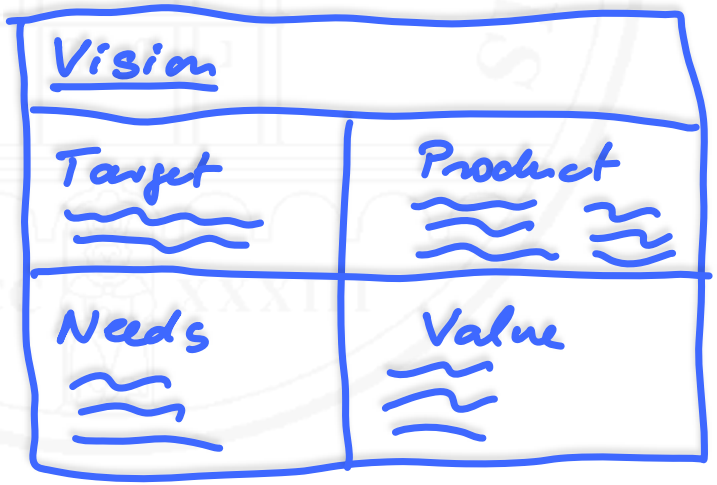
Pos	ItemID	Type	Content	Size
1	US-103	User Story	As a runner, I want to register at the portal to use the app.	L
2	US-107	User Story	As a runner, I want to configure my training parameters to prepare my first training.	XL
…				
18	US-243	User Story	As a coach, I want to see the runner's speed and heart rate in real time so that I can recognize when the runner is pushing too hard during a training session.	M
…				
31	T-2	Technical	Implement the training data transfer between app and portal.	M
…				

Work products and their purposes – 7

- Story map – visual arrangement of user stories



- Vision – a conceptual imagination of a future system or product



Work products and their purposes – 8

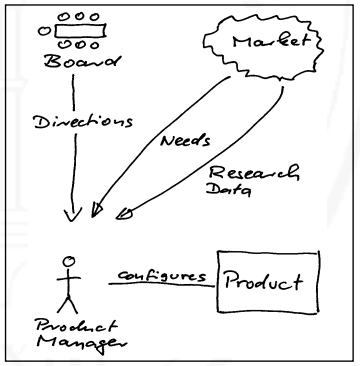
Other RE-related work products

- Glossary – providing an unambiguous and agreed common terminology

Safety shutdown — A situation where the operation of a device has to be shut down immediately due to the violation a safety condition.

Abbreviation: SSD
Synonym: Emergency shutdown (ESD

- Textual note or graphic sketch – serving for communication and understanding
- Prototype – understanding or validating requirements



3.2 Classic requirements specifications

Full-fledged requirements specifications are typically needed

- When customers want contractually fixed requirements, costs and deadlines
- When systems are built by an external contractor based on a set of given requirements (tendering, outsourcing)
- In regulated environments where regulators check compliance of developed systems to their requirements

Document types

[ISO/IEC/IEEE 2018]

- Stakeholder requirements specification (also called customer requirements specification; *Lastenheft* in German)
  What the stakeholders want (independent of any system providing it)
- System requirements specification (*Pflichtenheft* in German)
  The system or product to be developed and its context
- Software requirements specification
  If the system is a pure software system
- Business requirements specification
  High-level specification of business needs or goals

Stakeholder requirements specification

- Written when stakeholder needs shall be documented before any system development considerations are made
- Typically written by domain experts on the customer side (maybe with help of RE consultants)
- If a stakeholder requirements specification is written, it precedes and informs system or software requirements specifications

System/software requirements specification

- The classic form of a requirements specification
- No methodological difference between system requirements specification and software requirements specification
- Typically written by requirements engineers on the supplier side

Mini-Exercise

To which type of specification do the following requirements belong:

- System/software requirements specification?
- Stakeholder/user requirements specification?
- Business requirements specification?

(a) As a chairlift manager, I want daily statistics about chairlift usage so that I have data for my business decisions.

(b) The new system shall reduce ticket handling costs by > 20% and increase sales by > 5 %.

(c) The system shall operate under harsh weather conditions (snowfall, strong wind, temp range of -30° to +36° C).

3.3 Requirements in agile development

No classic requirements specification document (unless mandated by regulators)

Various work products that ...

- ... specify requirements: vision, user stories, epics, use cases, models...
- ... have requirements-related content: Prototypes, mock-ups, storyboards, roadmap, early product versions (e.g., MVP – minimum viable product)

Value-driven creation of work products

3.4 Glossary

RE typically is a multi-person endeavor

→ Danger of missing shared understanding in terminology

DEFINITION. Glossary – A collection of definitions of terms that are relevant in some domain.

A glossary defines

- Context-specific terms
- Everyday terms that have a special meaning in the given context
- Abbreviations and acronyms

Safety shutdown — A situation where the operation of a device has to be shut down immediately due to the violation of a safety condition.
Abbreviation: SSD
Synonym: Emergency shutdown (ESD)

Rules for creating and maintaining a glossary

- Consistently structured
- Centrally managed
- Defined responsibilities for creation and maintenance
- Maintained over the entire course of a project
- Usage of terms as defined in the glossary is mandatory
- Stakeholders must agree upon the glossary
- Synonyms and Homonyms properly treated
  - Synonyms (different terms denoting the same thing) marked
  - Homonyms (same term for different things) avoided or marked

3.5 Prototypes

DEFINITION. Prototype – In software and systems engineering: A preliminary, partial realization of certain characteristics of a system.

Serves for exploring, communicating or validating concepts and requirements.

The realization may be in any physical form, from paper and sticky notes over clickable pages to executable source code.

In RE, a prototype is a means for

- specifying requirements by example
- validating requirements
- supporting stakeholder communication and shared understanding

Forms of Prototypes in RE

[Lichter et al. 1994]

- *Exploratory prototype*:
  - Creating shared understanding
  - Clarifying requirements
  - Validating requirements on different levels of fidelity
  - Thrown away after use

- *Evolutionary prototype*:
  - Pilot system forming the nucleus of a system to be developed
  - Final system evolves by incrementally extending and improving the prototype

Exploratory prototypes

- *Wireframe*
  - Low-fidelity prototype
  - Built with paper or other simple materials
  - Primarily serves for discussing and validating design ideas and user interface concepts
- *Mock-up*
  - Medium-fidelity prototype
  - Demonstrates characteristics of a user interface without implementing any real functionality
  - Real screens and click flows, but without functionality behind
  - Primarily serves for specifying and validating user interfaces

Exploratory prototypes – 2

- *Native prototype*
  - High-fidelity prototype
  - Implements critical parts of a system to an extent that stakeholders can work with the prototype
  - Primarily serves for validating that the prototyped part of the system will work and behave as expected

Exploratory prototypes can be expensive work products

- Choose proper level of fidelity
- Trade-off between cost and value gained

Mini-Exercise

What forms of prototypes do we have here?

(a)

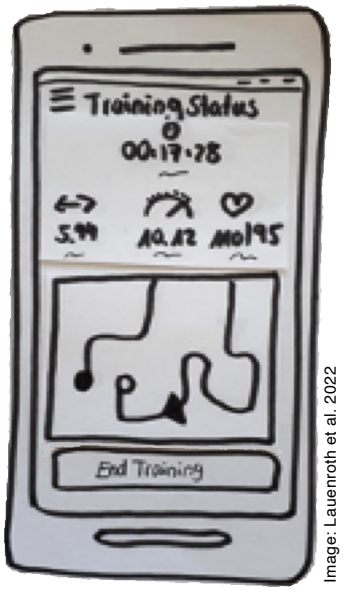
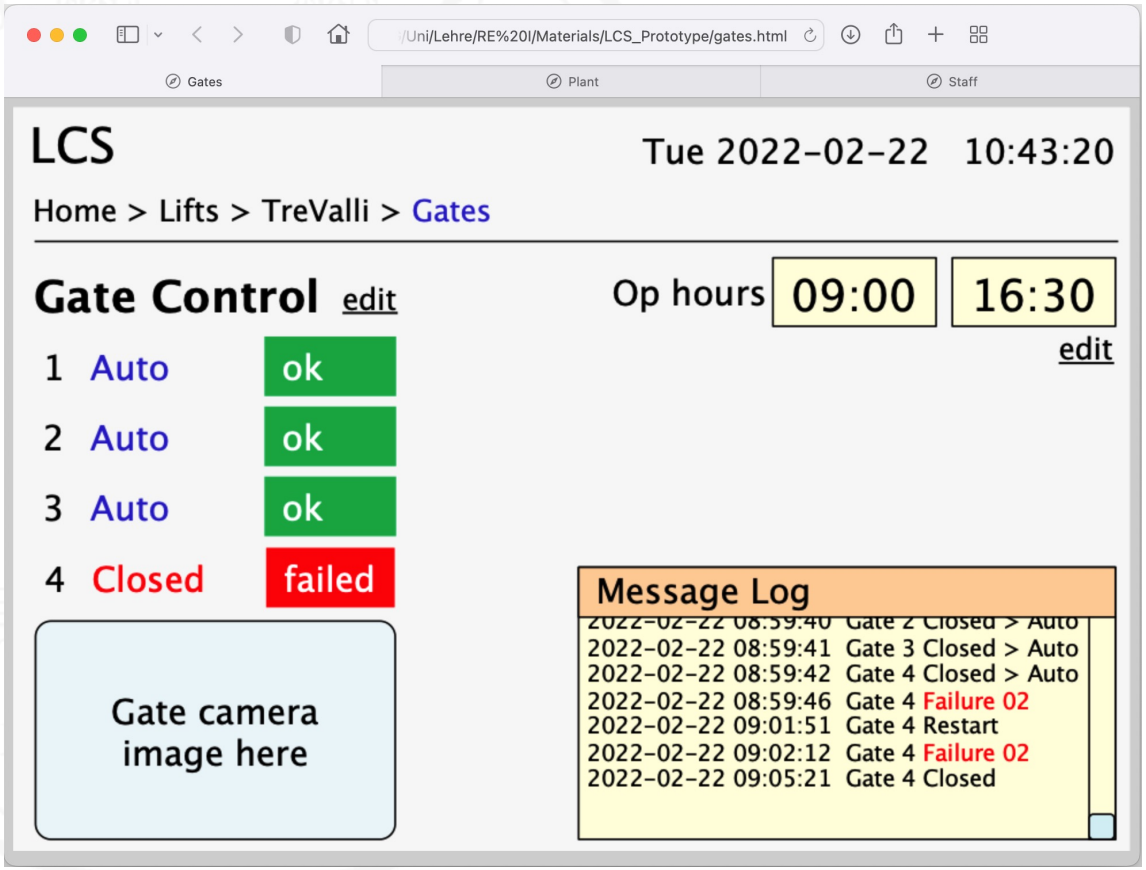


Image: Lauenroth et al. 2022

(b)



3.6 Aspects to be documented

Independently of any language, method, and documentation style, four aspects need to be documented:

- Functionality
  - Structure and Data: Static structure, (persistent) data
  - Function and Flow: Functions (results, preconditions, processing), flow of control and data
  - State and Behavior: State-dependent dynamic system behavior as observable by users
  - Both normal and abnormal cases must be specified

Aspects to be documented – 2

- Quality

  Performance

  - Data volume
  - Reaction time
  - Processing speed
  - Specify measurable values if possible
  - Specify more than just average values

  Specific Qualities

  - "-ilities" such as Usability, Reliability, Availability, etc.

Aspects to be documented – 3

- Constraints

  Restrictions that must be obeyed / satisfied

  - Technical: given interfaces or protocols, etc.
  - Legal: laws, standards, regulations
  - Organizational: given structures, policies, processes
  - Cultural: culturally shaped user habits and expectations
  - Environmental: e.g., energy consumption, heat dissipation
  - Physical: laws of physics, properties of materials
  - Solutions / restrictions demanded by important stakeholders

Aspects to be documented – 4

- Context and Boundary
  - Domain requirements and domain assumptions in the context of a system
  - Embedding of a system in its context
  - Interaction between a system and the actors in the context

3.7 How to document

Sample standards for classic requirements documents

ISO/IEC/IEEE Std 29148

- Templates for business, stakeholder, system & software specs
- Flat and rather unsystematic → not such useful

VOLERE

- System and project requirements in 27 chapters

IREB

- Simple template for system/software requirements specs

Enterprise-specific standards

- Imposed by customer or given by supplier

ISO/IEC/IEEE 29148 Standard

[ISO/IEC/IEEE 2018]

1. Introduction
   - 1.1 System purpose
   - 1.2 System scope
   - 1.3 System overview
     - 1.3.1 System context
     - 1.3.2 System functions
     - 1.3.3 User characteristics
2. References
3. System requirements
   - 3.1 Functional requirements
   - 3.2 Usability requirements
   - 3.3 Performance requirements
   - 3.4 Interface requirements
   - 3.5 System operations
   - 3.6 System modes and states
   - 3.7 Physical characteristics
   - 3.8 Environmental conditions
   - 3.9 Security requirements
   - 3.10 Information management requirements
   - 3.11 Policy and regulation requirements
   - 3.12 System lifecycle sustainment requirements
   - 3.13 Packaging, handling, shipping and transportation requirements
4. Verification

   (parallel to subsections in Sect. 3)
5. Appendices
   - 5.1 Assumptions and dependencies
   - 5.2 Acronyms and abbreviations

VOLERE

[Robertson and Robertson 2012]
[https://www.volere.org/templates/]

Project Drivers
1. The Purpose of the Project
2. The Stakeholders

Project Constraints
3. Mandated Constraints
4. Naming Conventions and Terminology
5. Relevant Facts and Assumptions

Context and Functionality
6. The Scope of the Work
7. Business Data Model & Data Dictionary
8. The Scope of the Product
9. Functional Requirements

Non-Functional Requirements
10. Look and Feel Requirements
11. Usability and Humanity Requirements
12. Performance Requirements
13. Operational & Environmental Requirements
14. Maintainability and Support Requirements
15. Security Requirements
16. Cultural Requirements
17. Compliance Requirements

Project & Product Issues
18. Open Issues
19. Off-the-Shelf Solutions
20. New Problems
21. Tasks
22. Migration to the New Product
23. Risks
24. Costs
25. User Documentation and Training
26. Waiting Room
27. Ideas for Solutions

Subtitles added by MG, inspired by an earlier version of the template

A simple document template

[Glinz et al. 2022]

Part I: Introduction

1. System purpose
2. Scope of system development
3. Stakeholders

Part II: System Overview

4. System vision and goals
5. System context and boundary
6. Overall system structure
7. User Characteristics

Part III: System requirements

Organized hierarchically according to system structure

Per sub-system/component:

- Functional requirements (structure and data – function and flow – state and behavior)
- Quality requirements
- Constraints
- Interfaces

References

Appendices

- Glossary
- Assumptions and dependencies

Documenting agile requirements

- General guideline: do things only if they add value
- A simple approach:
  - Writing user stories (using a standard template; cf. Chapter 6)
  - Adding detail with acceptance criteria
  - Structuring stories with epics and features
  - Organizing stories in a product backlog

[Leffingwell 2011]

➔ Implementation-oriented approach

➔ Does not scale well to settings with > 1 agile team

Scaled agile requirements documentation

An approach that scales

- Requirements @ business / system level:
  - System vision
  - For business systems: Models of business processes & data
  - For technical systems: Models of system behavior & functions
  - Domain models
  - Textual quality requirements and constraints
- Requirements @ user level: Use cases – value for users
- Work products persistently stored in a structured repository
  → preserves the value of the requirements

Scaled agile requirements documentation – 2

Mapping to agile development: story mapping

- Decomposing the business/system perspective
- Mapping use cases to user stories & acceptance criteria
- Structuring into epics and features
- Prioritizing and determining sequence of implementation
- Coordinating agile teams

How to document – language options

Informally

- Plain natural language (narrative text)

Semi-formally

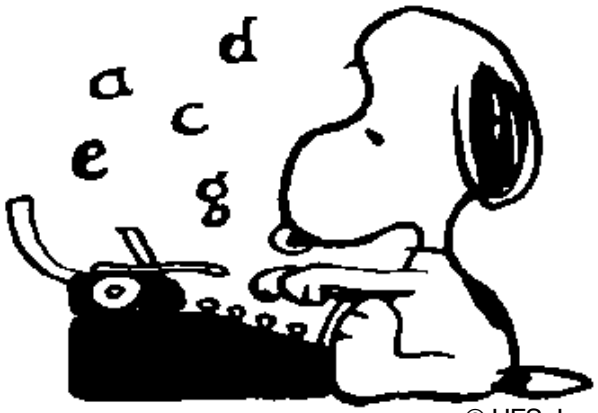
- Structured natural language (using templates or forms)
- Graphic models — Typically as diagrams which are enriched with natural language text

Formally

- Formal models, typically based on mathematical logic and set theory

General rules for requirements documentation

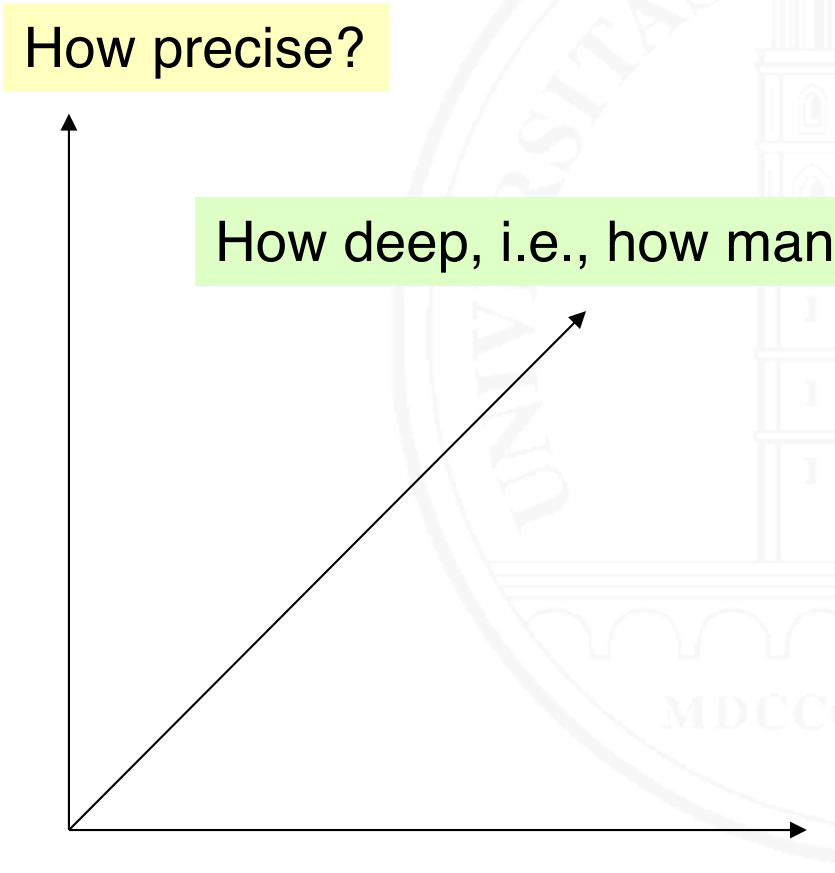
- Specify requirements as small, identifiable units whenever possible
- Record metadata such as source, author, date, status
- Use structure templates
- Adapt the degree of detail to the risk associated with a requirement
- Specify normal and exceptional cases
- Don't forget quality requirements and constraints



© UFS, Inc.

Precision – Detail – Depth

Three dimensions:



Dimensions influence each other:

- More precision → more detail
- More detail → more depth

Precision: reduce ambiguity

Restrict your language

Use a glossary

Define acceptance test cases

Quantify where appropriate

Formalize

Snoopy quantifies a love letter

Detail

What's better?

"The participant entry form has fields for name, first name, sex, ..."

"The participant entry form has the following fields (in this order): Name (40 characters, required), First Name (40 characters, required), Sex (two radio buttons labeled male and female, selections exclude each other, no default, required),..."

It depends.

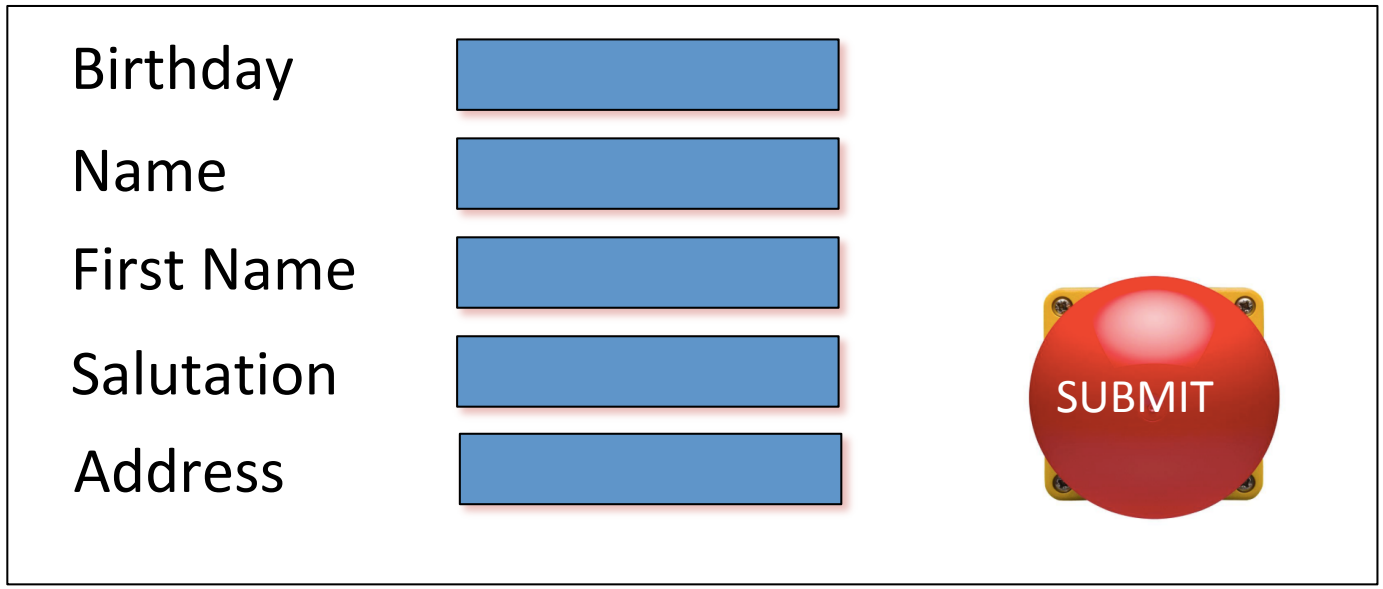
- Degree of implicit shared understanding of problem
- Degree of freedom left to designers and programmers
- Cost vs. value of detailed specification
- The risk you are willing to take

The risk of specifying too little detail

What the customer specified

"The registration entry form shall have fields for name, first name, salutation, birthday, and address and a submit-button."

What the supplier delivered



Depth

The more precise, the more information is needed

➔ Preserve readability with a hierarchical structure

"…

4.3 Administration of participants

4.3.1 Entering a new participant

4.3.1.1 New participant entry form

4.3.1.2 New participant confirmation

4.3.2 Updating a participant record

…"

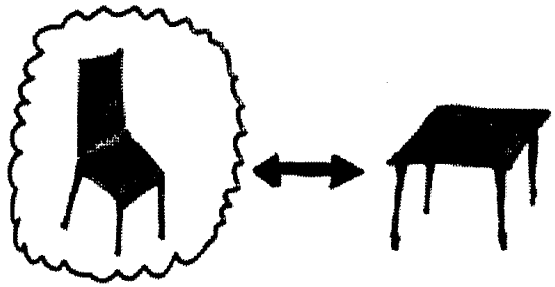
3.8 Quality of documented requirements

Two aspects of requirements quality

- Quality of individual requirements
- Quality of requirements work products, for example, a requirements specification

Hint: Don't confuse quality of requirements with quality requirements

Quality of individual requirements



For individual requirements, strive for requirements that are...

- Adequate — True and agreed stakeholder needs
- Understandable — Prerequisite for shared understanding
- Verifiable — Conformance of implementation can be checked
- Unambiguous — True shared understanding
- Complete — No missing parts
- Necessary — Part of the relevant system scope
- Feasible — Non-feasible requirements are a waste of effort

Quality of requirements work products

When creating a requirements work product, strive for a work product that is

• Consistent	No contradictions
• Complete	Contains the relevant requirements
• Conformant	Conforms to prescribed work product structure, format or style
• Modifiable	Because change will happen
• Non-redundant	Requirements do not overlap
• Structured	Improves readability of work product
• Traceable	Linked to related artifacts

Quality criteria are in the eye of the beholder

- No general consensus
- Different, overlapping sets of quality criteria used in
  - this course
  - RE textbooks
  - RE standards (e.g., ISO/IEC/IEEE 29148:2018)
  - Quasi-standards such as the IREB Certified Professional for Requirements Engineering (see https://www.ireb.org)

Not all qualities are equally important

- Adequacy and understandability are key
- Verifiability and Consistency are very important
- Achieving total completeness and unambiguity is neither possible nor economically feasible in most cases
- The importance of feasibility, traceability, conformance, etc. of requirements depends on the concrete project/situation

☞ Strive for value, not for blind satisfaction of requirements quality criteria!