Requirement Engineering

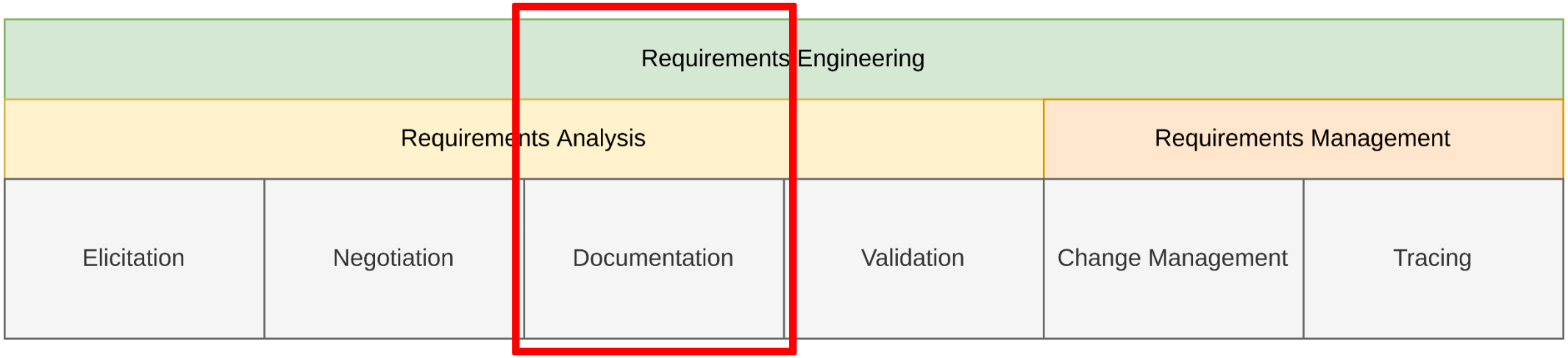
Lecture 4: Requirements Documentation

Part 1

Prof. Dr. Benjamin Leiding
Anant Sujatanagarjuna

General Requirements Engineering Process

Overview



Lecture 4: Requirements Documentation

Content

1. Documentation in General
2. Criteria for Good Requirements
3. Document Structure
4. Document Templates

DOCUMENTATION IN GENERAL

Documentation in General

Software Projects

- Require **cooperation** of large numbers of people.
- May extend over a significant period of **time.**
- Are typically based on **contracts.**
- May require **maintenance** over many years.
- Complete **change of personnel** might be required.

→ Documentation is a means to deal with these issues.

Documentation in General

Software Projects

- Require **cooperation** of large numbers of people
- May extend over a significant period of **time**
- Are typically based on **contracts**
- May require **maintenance** over many years
- Complete **change of personnel** might be required

→ Documentation is a means to deal with these issues.

Agile methods try to avoid documentation.

This leads to short term gains, but incurs long term costs

Documentation in General

Why?

- Requirements are the basis of system development
  - Influence analysis, design, implementation, and test phases
  - Requirements document has strong impact on the project
- Specifications have legal relevance
  - Often the foundation for contracts
  - Ambiguities may lead to legal conflicts
- Requirements are possibly extremely complex
  - Thousands of requirements
  - Complex interdependencies
- Requirements must be accessible
  - Without documentation access for all involved persons not possible

Documentation in General

Communication

Documentation serves communication purposes:

- Communication within the team
- Communication over time (e.g., change of personnel)
- Communication between development team and customer (contracts)

Depending on:

- The importance of these dimensions
- The involved stakeholder groups (e.g., capability to deal with notations)
- Project characteristics (e.g., reliabiliy, maintenance period)
- The amount, form and contents of the requirements documentation needs to be adapted

Documentation in General

Documentation vs Specification

Documentation

→ Refers to any form of information written down relating to a software or system artifact.

Requirements documentation

→ Any form of explicit documentation of a requirement

Requirement

→ Any (documented) information relating to a system that shall be developed

Note:

- This is sometimes used in contrast to specification, then requirements only refers to informal (i.e. abstract) requirements

Documentation in General

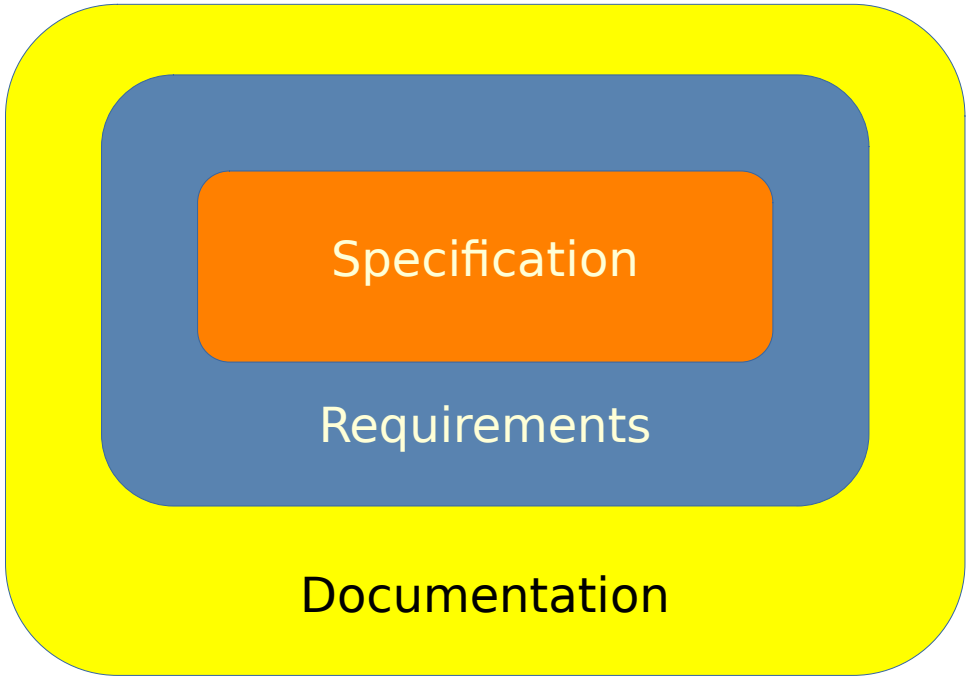
Documentation vs Specification

Specification

→ Requirements documentation that is in accordance with a certain specification approach. (not necessarily formal)

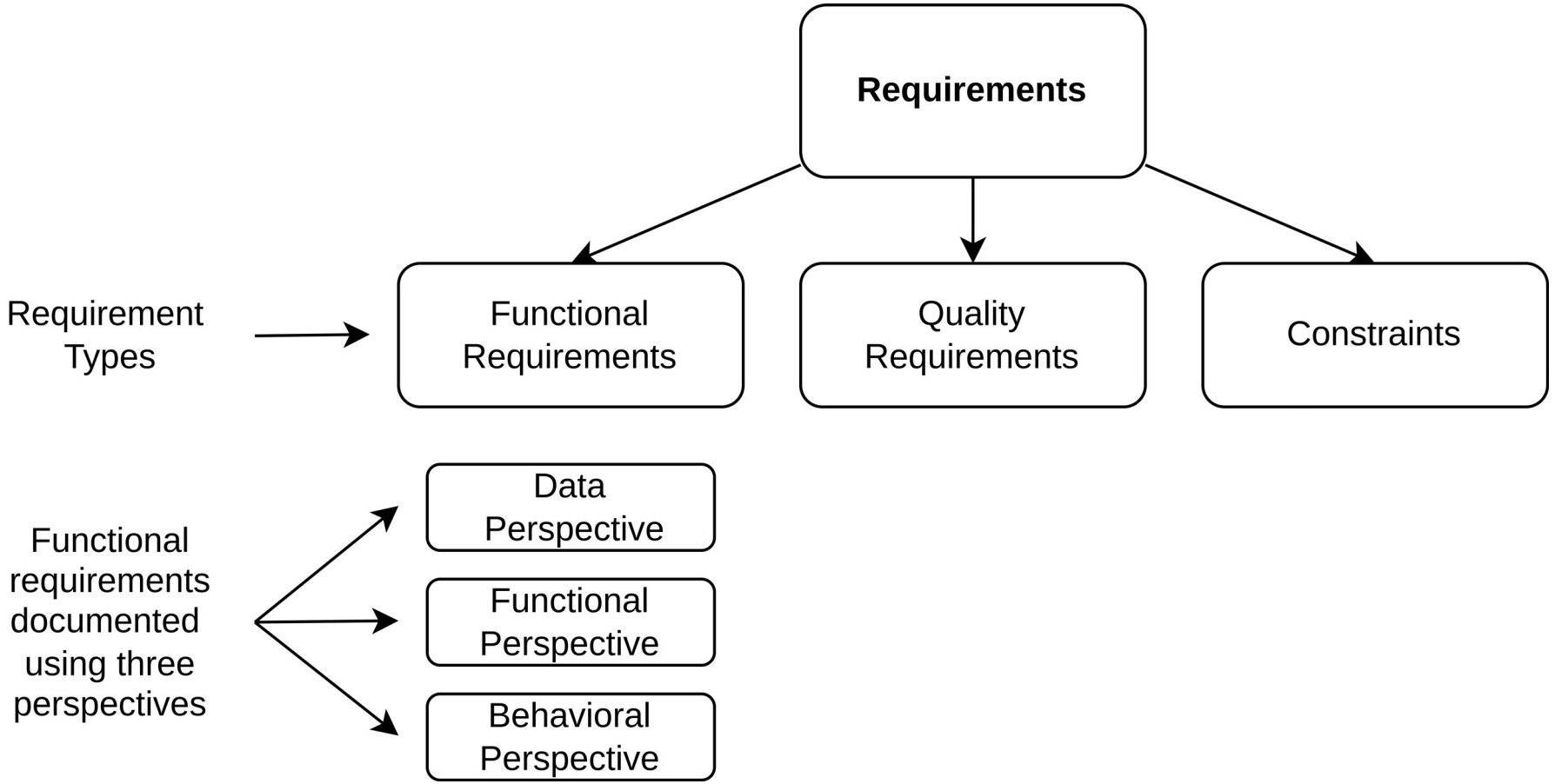
Note:

- This is sometimes also used to imply that the requirements are specified on the level of developer requirements (we will not use it this way).



Documentation in General

Perspectives on Functional Requirements



Documentation in General

Perspectives on Functional Requirements

- Three different perspectives
  - Data
  - Functional
  - Behavioral

Documentation in General

Perspectives on Functional Requirements

- Data perspective
  - Static-structural perspective
  - Structure of input/output data
  - Static structure of the system itself
    - Usage relationships
    - Dependencies

Documentation in General

Perspectives on Functional Requirements

- Functional perspective
  - Which information is received by the system and how is it manipulated?
  - Data-flow through the system
    - Dynamic!

- Behavioral perspective
  - State-oriented perspective on how the system is embedded into the system context
  - Documents reactions by the system on events in the system context
    - Reaction depends on the system’s state
    - Reactions include state transitions and effects on the system’s environment

Documentation in General

Types of Documentation

- **Three** different ways to document requirements
  - Natural language
  - Conceptual Models
  - Hybrid approach

- No “one best way”
  - Depends on the project
    - Requirements
    - People involved

Documentation in General

Types of Documentation – Natural Language

- Most commonly applied documentation form
  - Usually: prose
- Advantages
  - Can be read by every stakeholder
  - Applicable for miscellaneous purposes
    - Can be used for any kind of requirement
    - Well-suited for all three perspectives
- Drawbacks
  - Ambiguities possible
  - Unintentional mix of different kinds of requirements possible
  - Unintentional mix of different perspectives possible

Documentation in General

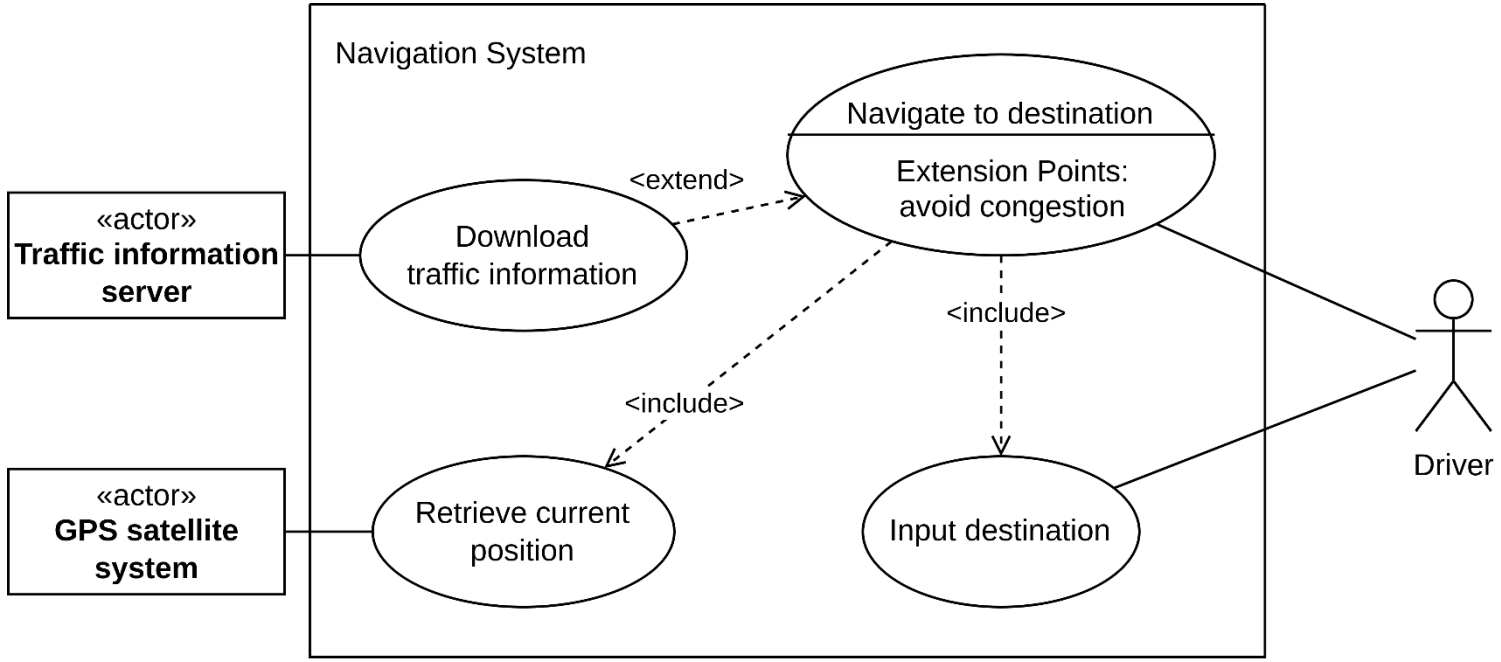
Types of Documentation – Conceptual Models

- Different kinds of conceptual models
  - Cannot be used universally
  - Have to fit the type of the requirements and the perspective
- Advantages
  - Correct use → no switch of perspective
  - Compact documentation
    - Easy for trained readers
  - Less ambiguity than natural language
- Drawbacks
  - Knowledge about modeling required

Documentation in General

Types of Documentation – Conceptual Model Types

- Use case diagrams
  - Quick overview of system functionalities
  - Do not describe responsibilities in detail

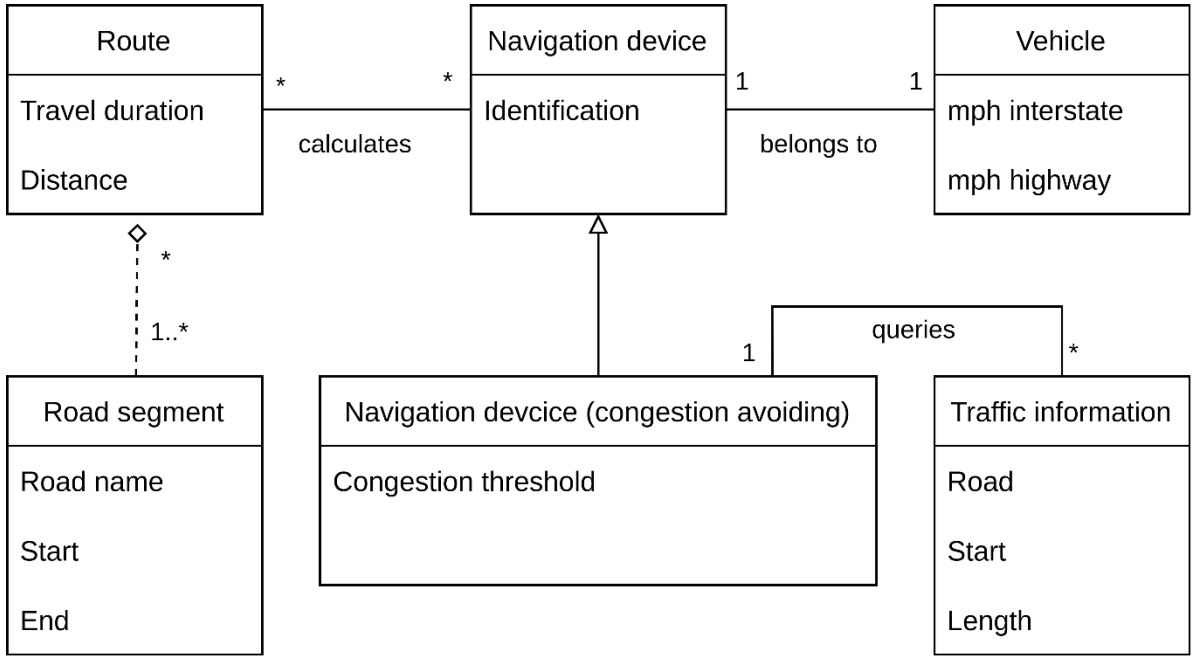


Based on K. Pohl, C. Rupp (2011) – Requirements Engineering Fundamentals: A study guide for Requirements Engineering Foundation Level

Documentation in General

Types of Documentation – Conceptual Model Types

- Class diagrams
  - Capture static structure of data
  - Structural dependencies between the system and the context

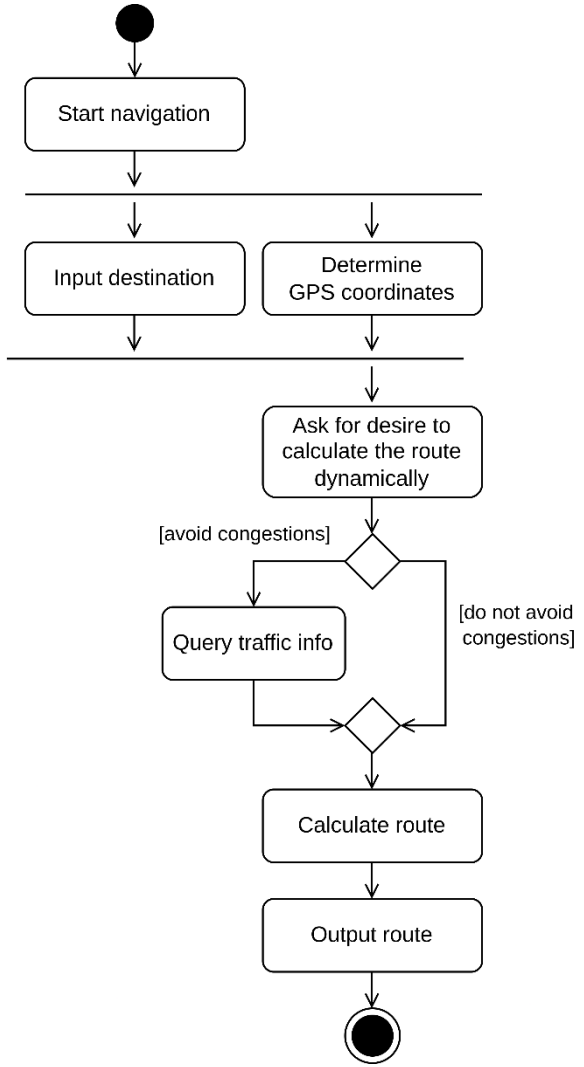


Based on K. Pohl, C. Rupp (2011) – Requirements Engineering Fundamentals: A study guide for Requirements Engineering Foundation Level

Documentation in General

Types of Documentation – Conceptual Model Types

- Activity diagrams
  - Business processes, sequence-oriented dependencies
  - Sequential character of use cases

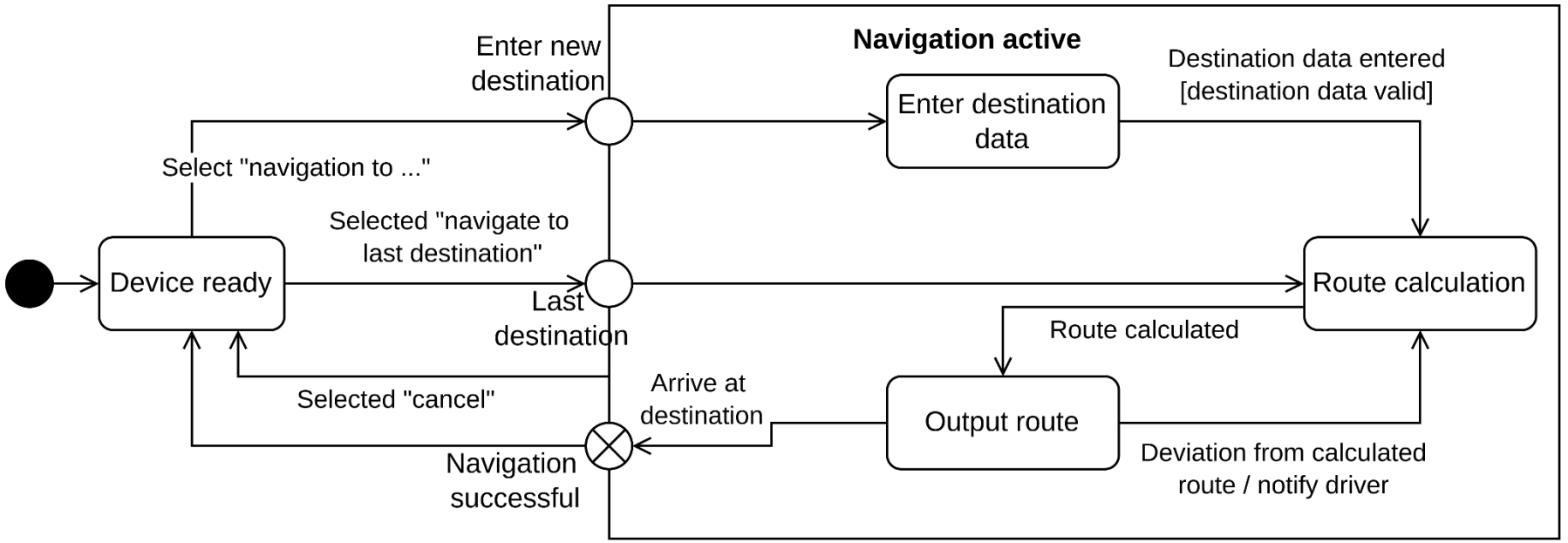


Based on K. Pohl, C. Rupp (2011) – Requirements Engineering Fundamentals: A study guide for Requirements Engineering Foundation Level

Documentation in General

Types of Documentation – Conceptual Model Types

- State diagram
  - Event-driven behavior of a system



Based on K. Pohl, C. Rupp (2011) – Requirements Engineering Fundamentals: A study guide for Requirements Engineering Foundation Level

Documentation in General

Types of Documentation – Conceptual Models

- No single type of documentation has to be selected
  - Depends on the view, audience, ...
- Hybrid = natural language + conceptual models
  - Not redundant, but complementary
  - Balance out each others weaknesses
- Example
  - Description of the general architecture in natural language
  - Details in form of models

CRITERIA FOR GOOD REQUIREMENTS

Criteria for Good Requirements

Characteristics of Good Requirements

- Agreed
  - Requirement reflects the correct and agreed upon opinion of all stakeholders
- Ranked (IEEE Std. 830-1998)
  - E.g., by legal obligations or priorities
  - Especially important if not all functionalities are provided at with the same release
- Unambiguous (IEEE Std. 830-1998)
  - Can be understood in only one way
  - Conceptual models help to achieve this
- Valid and up-to-date
  - Must be consistent with the system context and not outdated

IEEE 830-1998 (1998) – IEEE Recommended Practice for Software Requirements Specifications (https://standards.ieee.org/ieee/830/1222/)

Criteria for Good Requirements

Characteristics of Good Requirements

- Correct (IEEE Std. 830-1998)
  - The requirement represents the idea of the stakeholder
- Consistent (IEEE Std. 830-1998)
  - Requirements must not contradict each other
  - Requirements must be on the same level of abstraction/detail
  - Requirements should use the same documentation type

IEEE 830-1998 (1998) – IEEE Recommended Practice for Software Requirements Specifications (https://standards.ieee.org/ieee/830/1222/)

Criteria for Good Requirements

Characteristics of Good Requirements

- Verifiable (IEEE Std. 830-1998)
  - Definition of a requirement must allow for its verification → e.g., through tests or measurements
- Realizable
  - Requirements must be implementable within the scope of the project
- Traceable (IEEE Std. 830-1998)
  - Retracing of
    - the origin
    - the realization
    - Relationships to other documents → e.g., through unique identifiers

IEEE 830-1998 (1998) – IEEE Recommended Practice for Software Requirements Specifications (https://standards.ieee.org/ieee/830/1222/)

Criteria for Good Requirements

Characteristics of Good Requirements

- Complete (IEEE Std. 830-1998)
  - Requirements must fully describe the specified functionality
  - If not yet known → mark, e.g., as “tbd” (“to be determined”)
- Understandability
  - Requirements must be comprehensible for all stakeholders

IEEE 830-1998 (1998) – IEEE Recommended Practice for Software Requirements Specifications (https://standards.ieee.org/ieee/830/1222/)

Criteria for Good Requirements

Further Characteristics of Good Requirements

- Atomic (it is not possible to subdivide the requirement)

Criteria for Good Requirements

Example – Unambiguous

In order to retrieve money from the ATM (automatic teller machine), the customer needs to insert a valid card and type in his PIN. If the validation fails, the machine keeps the card.

When the driver turns the steering wheel, the direction of the car is changed accordingly.

Why are these **not** unambiguous?

Criteria for Good Requirements

Example – Atomic

As part of the web shop, the customers may search for goods using key words or product categories.

Why is this not atomic ?

DOCUMENT STRUCTURE

Document Structure

Characteristics of Good Requirements Documents

- Unambiguity and Consistency (IEEE Std. 830-1998)
  - Requires individual requirements also to be unambiguous and consistent
  - All requirements should be uniquely identifiable
- Modifiability and Extendibility (IEEE Std. 830-1998)
  - Must be easy to modify and to extend
  - Should be subject to a version control management system

IEEE 830-1998 (1998) – IEEE Recommended Practice for Software Requirements Specifications (https://standards.ieee.org/ieee/830/1222/)

Document Structure

Characteristics of Good Requirements Documents

- Completeness (IEEE Std. 830-1998)
  - Must contain **all** requirements
  - Each requirement must be documented completely
  - Required additional information must be present

Document Structure

Characteristics of Good Requirements Documents

- Includes:
  - Required reactions
  - Influential factors
  - Inputs
  - Error and exception cases
  - Quality requirements
  - Structural completeness
- Labels for figures and tables
- Consistent references and index directories

IEEE 830-1998 (1998) – IEEE Recommended Practice for Software Requirements Specifications (https://standards.ieee.org/ieee/830/1222/)

Document Structure

Characteristics of Good Requirements Documents

- Traceability (IEEE Std. 830-1998)
  - Relationships between requirements documents and other documents
  - Traceability of changes
- Clear Structure
  - Comprehensive and clear structure
  - Examples shown in the previous section

IEEE 830-1998 (1998) – IEEE Recommended Practice for Software Requirements Specifications (https://standards.ieee.org/ieee/830/1222/)

Document Structure
Types of Documents – Characterizing Developer vs. User Requirements

- Lastenheft (engl. product requirements document)
  - Written by the customer
  - Defines what is expected / has to be provided
  - Legally: may define a call for bids

Document Structure
Types of Documents – Characterizing Developer vs. User Requirements

- Pflichtenheft (engl. scope statement)
  - Written by the development organization
  - Defines what will be delivered
    (this may exceed, restrict, or modify the expectations of the “Lastenheft”)

Lasten- and Pflichtenheft are mostly defined by their implications for legal affairs and negotiation

Document Structure
Types of Documents – Characterizing Developer vs. User Requirements

- Lastenheft
  - Should not overconstrain the possible solution
  - Should focus on prioritized expectations
- Pflichtenheft
  - Should also have a user focus in writing
  - Also addresses “User requirements”
  - Typically uses the same techniques for specification as in Lastenheft

Note: There are typically requirements that are not described in “Pflichtenheft”, e.g., internal to development organization, higher level of detail, technical aspects

Document Structure

Required Content – ToC

- Every requirement document should contain certain contents
  - Introduction
  - General Overview
  - Requirements
  - Appendices
  - Index

- Independent of the concrete structure

Document Structure

Required Content – Introduction

- Contains information about the entire document
- Provides an overview of the system
- Addresses the following issues
  - Purpose Why was the document created, who is the target audience
  - System coverage → Name, principle goals, and advantages of the system to be developed
  - Stakeholder → List of stakeholders and their relevant information

Document Structure

Required Content – Introduction

- Definitions, acronyms, and abbreviations
  - Terms used throughout the document are defined for consistent use
  - Can also be in the appendix
- References (might be part of the appendix) → List of all referenced documents
- Overview → Outline of the content and structure of the remainder of the document

Document Structure

Required Content – General Overview

- Information that increase the understandability of the requirements
- Operational information
  - No administrative, management of organizational aspects
- Addresses the following issues:
  - System environment
    - How is the system embedded into its environment
    - Defines the system boundary and the context boundary
  - Architecture description
    - Operational interfaces of the system
    - Additional information pertaining to the architecture → e.g., storage limitations

Document Structure

Required Content – General Overview

- System functionality
  - Tasks and coarse functionalities of the system → e.g., use cases
- User and target audience
  - Users of the system
  - State the target audience separately → Not all users are necessarily part of the target audience!
- Constraints
  - Everything that is not documented elsewhere that poses constraints on the system
- Assumptions
  - General assumptions about the system context → e.g., that a certain functionality is out of scope due to budgeting reasons

Document Structure

Required Content – Remainder

- Requirements
  - Contains the functional and non-functional requirements
- Appendices
  - Additional information that completes the document
- Standards, conventions, background information
- Index
  - Table of contents
  - Index directory

Document Structure

Glossary – Why?

- Some technical terms are ambiguous
- Example:
  - Interface
    - User Interface
    - Java Interfaces
    - API
    - ...
  - Service
    - Web service
    - Generic term to describe provided functionality

Document Structure

Glossary – Why?

- Some terms may not be known by all stakeholders
- Some terms may be specific to the project

- Example
  - Model Driven Architecture
  - Interoperability Testing Framework

Document Structure

Glossary – Purpose

- Definition of the meaning of terms
  - Increases the understandability
  - Avoids misunderstandings and different interpretations
- Unifies language between the stakeholders
- Glossary entries can (and should) be reused
  - Across projects
  - Some definitions are even universal → IEEE Std. 610.12-1990: Standard Glossary of Software Engineering Terminology

Document Structure

Glossary – Definition

Reference book for the terms in the requirements document

"A glossary is a collection of technical terms that are part of a language (terminology). A glossary defines the specific meaning of each of these terms. A glossary can additionally contain references to related terms as well as examples that explain the terms."

K. Pohl (2010) – Requirements Engineering – Fundamentals, Principles and Techniques

Document Structure

Glossary – Elements

- Context-specific technical terms
- Abbreviations and acronyms
- Everyday concepts with special meaning in the given context
- Synonyms
  - Same meaning, different terms
- Homonyms
  - Different meaning, same terms

Document Structure

Glossary – Rules

- Central Management
  - Only one valid glossary at a time
- Responsibilities must be clear
  - One individual must be responsible for the glossary
  - Avoids confusion
- Maintenance
  - The glossary is a living document
  - Must be maintained throughout the life cycle
- Commonly accessible
  - Available for all involved persons

Document Structure

Glossary – Rules

- Obligatory
  - Only the terms from the glossary are to be used
  - If additional synonyms are available, they are not to be used!
- Should include sources
  - If a source for a term is available, it should be part of a glossary
  - Avoids discussions about terms
- Agreed upon between stakeholders
  - The stakeholders should agree on the terminology that is used
- Consistent structure
  - All entries must have the same structure

Document Structure

Glossary – Structure

- Each entry in a glossary should have the same general structure
- A good glossary structure contains:
  - The name of the term
  - The definition of the term
  - Synonyms of the term
  - Related terms
  - Examples/counter-examples

Document Structure

Glossary – Structure (Example)

- Example of a (tabular) glossary entry

Term	Route
Definition	The specific distance of a direction from a starting point to a destination.
Synonyms	Itinerary
Related Terms	Alternative route (specialization)
Examples/Counter-examples	https://goo.gl/maps/3L82YanUoidbj1HJ6

Document Structure

Glossary – How to create a glossary?

- Definition of the structure of the glossary
- Provide initial entries and definitions for all glossary entries
- Ask stakeholders to
  - Check the definitions
  - Provide their own definition in case of differences
  - Identify missing entries
- Update and align definitions in the glossary
- Make the glossary available to everyone
  - Ideally online

DOCUMENT TEMPLATES

Document Templates

Standardized Document Structures

- Standard outlines predefine the structure
  - Simplify incorporation of new staff members
  - Desired contents can be quickly found
  - Selective reading possibly
  - Automated verification of documents → e.g., their completeness (is every required section present)
  - Reuse of contents of other requirements documents

Structures must be tailored to project properties!

Document Templates

Standardized Document Structures – Overview

- Various document templates and guidelines

- Some selected (common) examples:
  - IEEE-830
  - V-Model
  - V-Model XT
  - Volere

Document Templates

IEEE Std. 830-1998

- *Recommended Practice for Software Requirements Specifications*
- Developed by IEEE
- Presents a template that might be used to specify developer requirements (some times it is partially used to describe user developer requirements as it contains parts that are on a higher level)
- In addition the template provides characteristics for a good software requirements specification document

Document Templates

IEEE Std. 830-1998 – Structure

- Suggest dividing the document into three main chapters

  1 Chapter with introductory information
  - System goal, system bounds, ...

  2 Chapter with general descriptions of the software
  - Perspective of the system, future users, constraints, ...

  3 Chapter with specific requirements
  - Functional and non-functional requirements

Document Templates

IEEE Std. 830-1998 – Example

- Introduction
  - Purpose
  - Definitions
  - System Overview
  - References

Document Templates

IEEE Std. 830-1998 – Example

- Overall Description
  - Product perspective
    - System Interfaces
    - User Interfaces
    - Hardware Interfaces
    - Software Interfaces
    - Communication Interfaces
    - Memory Constraints
    - Operations
    - Site Adaptation Requirements
  - Product functions
    - User Characteristics
    - Constraints, Assumptions and Dependencies

Document Templates

IEEE Std. 830-1998 – Example

- Specific Requirements
  - External Interface Requirements
  - Functional Requirements
  - Design Constraints
    - Standard Compliance
  - Logical Database Requirements
  - Software System Attributes
    - Reliability
    - Availability
    - Security
    - Maintainability
    - Portability
  - Other Requirements

Document Templates

V-Model

- Defines different structures, depending on the creator of the document
  - In Germany DIN 69905 defines the structure and terminology of the “Lastenheft” and “Pflichtenheft”
- Customer Requirements Specification (“Lastenheft”)
  - Describes demands on the contractor
    - Deliveries and services
    - Often includes demands of users
  - Includes constraints on the system, the development, ...

Document Templates

V-Model

- System Requirements Specification (“Pflichtenheft”)
  - Based on the Costumer Requirements Specification
    - Refinement that includes implementation suggestions

Document Templates

V-Model – Structure

- Structure of the Customer/System Requirements Specification
  - Objectives
  - Product Usage
  - Product Functions
  - Product Data
  - Product Requirements
  - Quality Requirements
  - Supplements
  - Glossary
- Difference in the level of detail

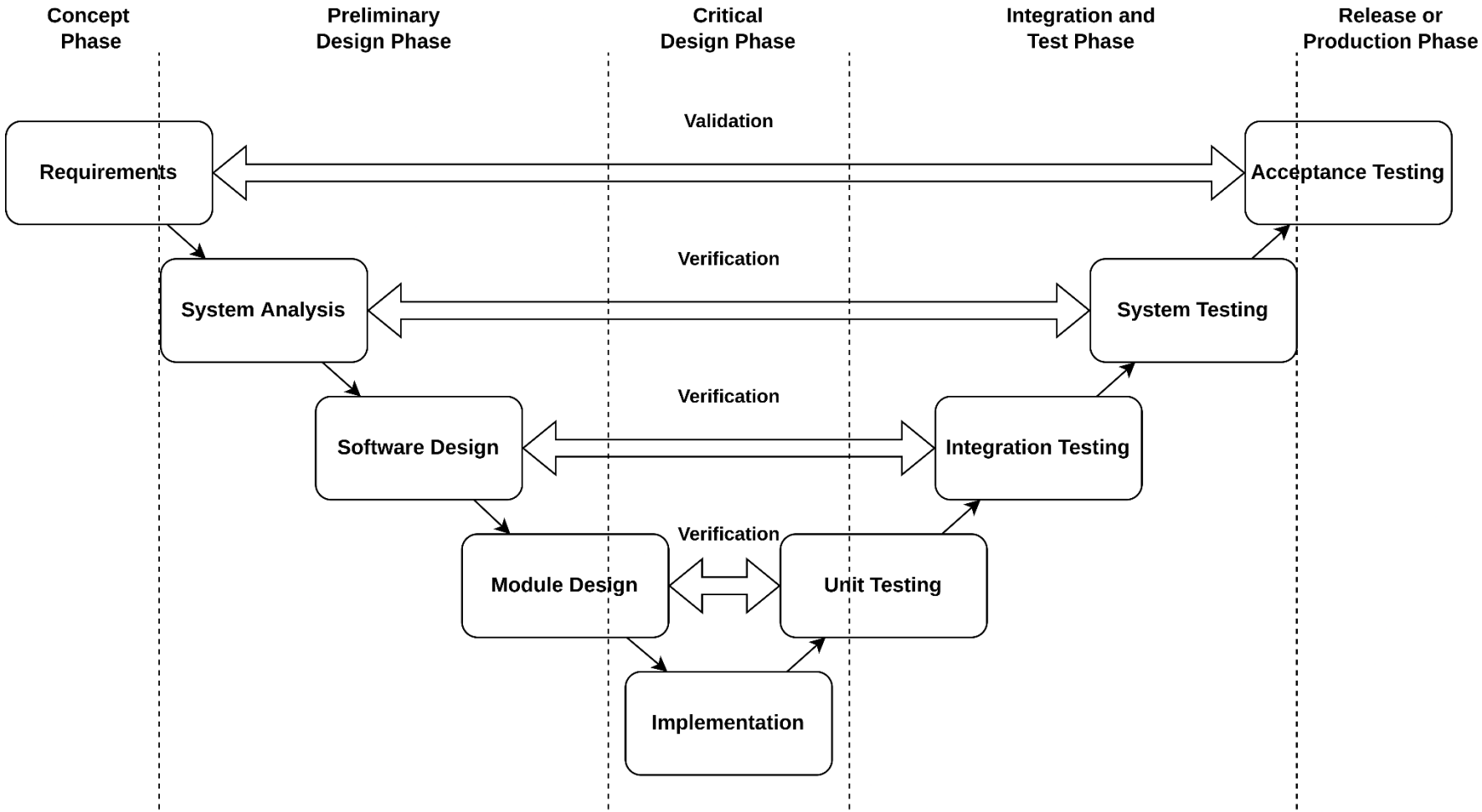
Document Templates

V-Model XT

- XT = *Extreme Tailoring*
- V-Model vs. V-Model XT:
  - Tailoring V-Model to specific needs, thereby avoiding unnecessary work by defining deletion conditions for small and medium-sized projects
  - Better suited for smaller and medium-sized projects
  - Involvement of the client: Up to now, the specifications were geared towards the contractor.
  - Greater modularization
  - Stronger orientation towards agile and incremental approaches
  - Available Online (German): Link

Document Templates

V-Model (XT) – Why V?



Document Templates

Volere Template

- Developed by James & Suzanne Robertson (The Atlantic Systems Guild)

- Presents a template that may be used to specify user requirements as well as developer requirements
  - some template sections describe very detailed information about the system while other sections are very high level (developer vs. user)
  - some template sections can be used for a developer audience as well as a user audience.
  - In these cases either the used notation is the key differentiator or the information contained in the user document is refined in the developer section
- Available online: Link

https://www.volere.org/

Document Templates

Volere Template – Structure

1. The Purpose of the Project
2. The Stakeholders

Project Constraints

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

Functional Requirements

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

https://www.volere.org/

Document Templates

Volere Template – Structure

Non-functional Requirements

10. Look and Feel Requirements
11. Usability and Humanity Requirements
12. Performance Requirements
13. Operational and Environmental Requirements
14. Maintainability and Support Requirements
15. Security Requirements
16. Cultural Requirements
17. Legal Requirements

https://www.volere.org/

Document Templates

Volere Template – Structure

Project Issues

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

https://www.volere.org/

SUMMARY

Summary

- Different types of documentation
  - Natural language, conceptual models, and hybrid approach
- Different perspectives
  - Data, functional, and behavioral
- Standardized structures available
  - Standards provide means to structure requirements documents
  - Indicate what should be the content of a requirements specification

Summary

- Standards do not:
  - indicate how to specify different parts or
  - indicate how to guarantee the characteristics of a good document
  - support to choose notation to specify a certain section
  - support in how to achieve for example completeness or traceability
- Quality of the documentation is important

Questions?