POLITECNICO
MILANO 1863

Software Engineering 2

Structure of a RASD document

RASD assignment

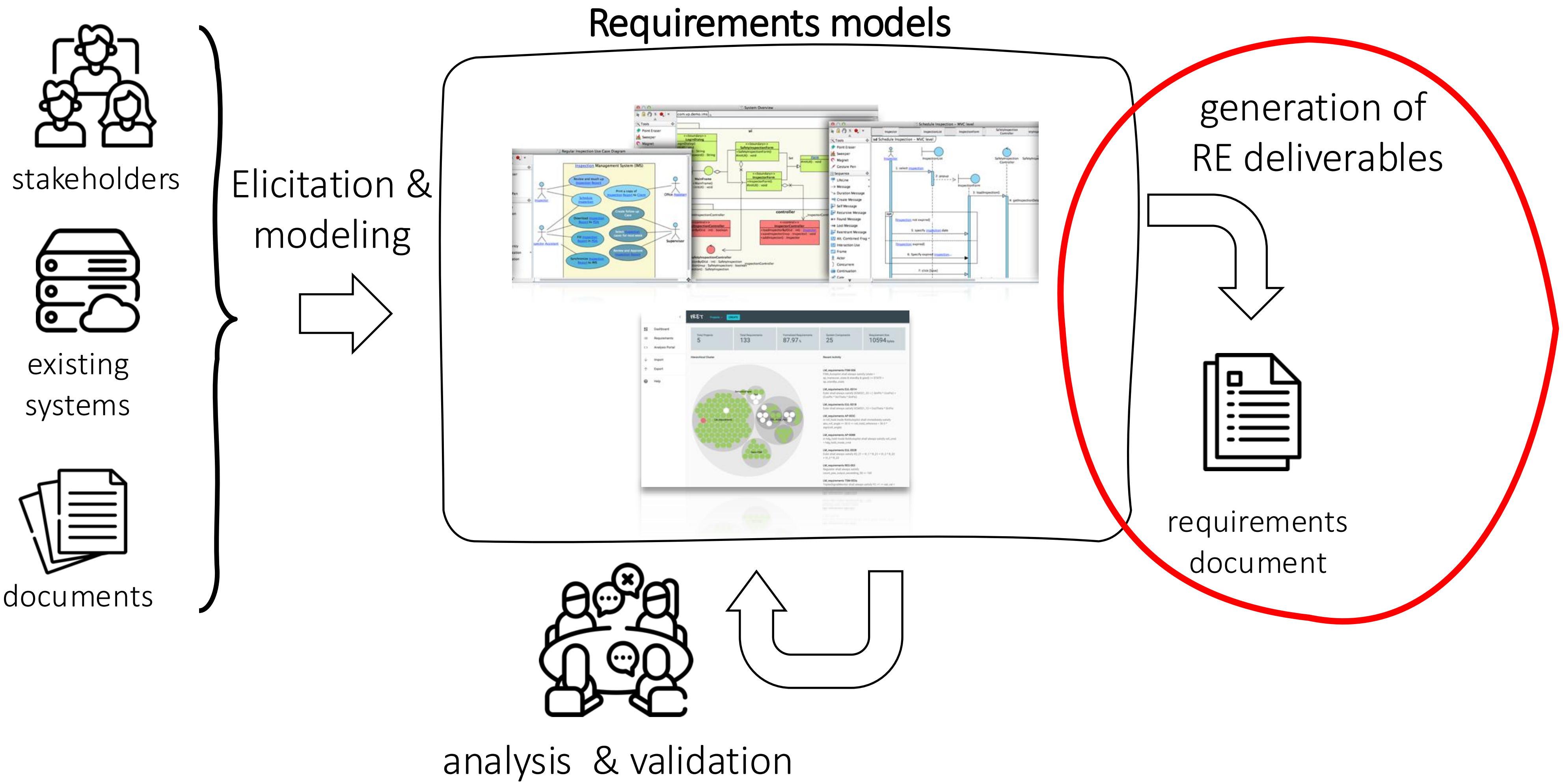
M Camilli, E Di Nitto, M Rossi

POLITECNICO
MILANO 1863

Requirements Engineering (RE)

Requirements Analysis and Specification Document (RASD)

RASD



stakeholders

existing systems

documents

Elicitation & modeling

Requirements models

analysis & validation

generation of RE deliverables

requirements document

Purposes of the RASD

- Communicates an understanding of the requirements
    - explains both the application domain and the system to be developed
- Contractual
    - may be legally binding!
- Baseline for other activities
    - Project planning and estimation (size, cost, schedule)
    - Software V&V
        - supports system testing, verification and validation activities
        - should contain enough information to verify whether the delivered system meets requirements
    - Change control
        - requirements change, software evolves

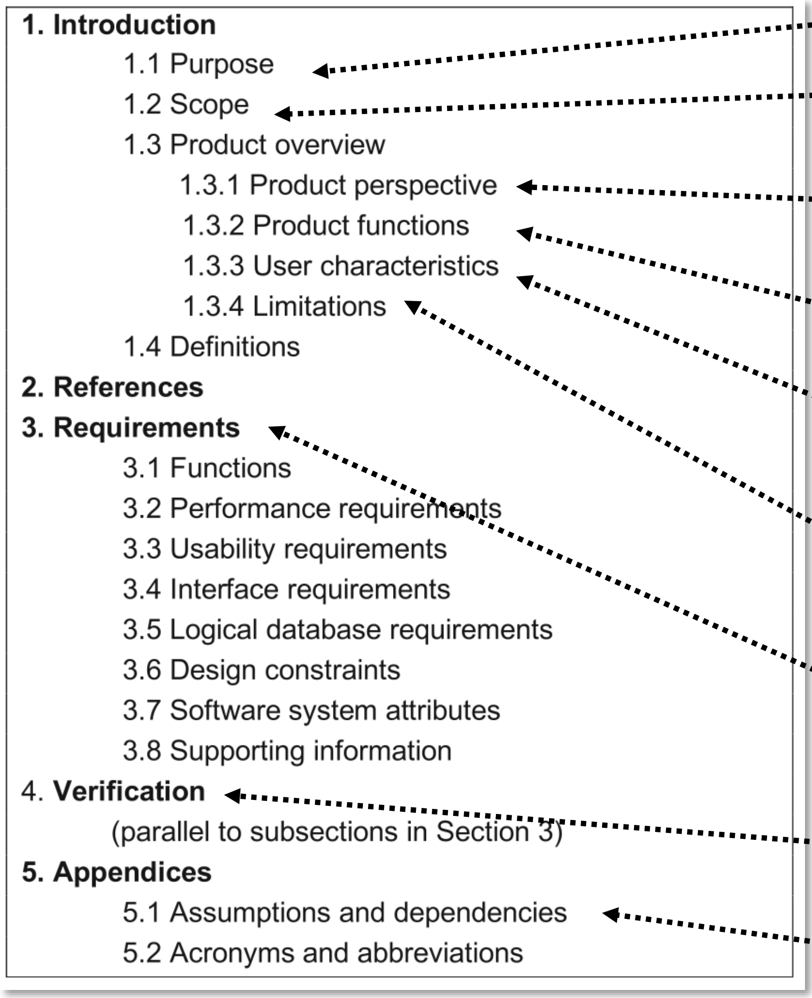
Audience of the RASD

- Customers & Users
  - most interested in validating system goals and high-level description of functions
  - not generally interested in detailed software requirements
- Systems Analysts, Requirements Analysts
  - write various specifications of other systems that inter-relate
- Developers, Programmers
  - Eventually implement the requirements
- Quality Assurance teams
  - determine that the requirements have been met
- Project Managers
  - measure and control the analysis and development processes

IEEE Standard for RASD – example of possible structure

Source: ISO/IEC/IEEE 29148 (Nov 2018)

https://doi.org/10.1109/IEEESTD.2018.8559686



1. Introduction
- 1.1 Purpose
- 1.2 Scope
- 1.3 Product overview
  - 1.3.1 Product perspective
  - 1.3.2 Product functions
  - 1.3.3 User characteristics
  - 1.3.4 Limitations
- 1.4 Definitions

2. References

3. Requirements
- 3.1 Functions
- 3.2 Performance requirements
- 3.3 Usability requirements
- 3.4 Interface requirements
- 3.5 Logical database requirements
- 3.6 Design constraints
- 3.7 Software system attributes
- 3.8 Supporting information

4. **Verification**
(parallel to subsections in Section 3)

5. Appendices
- 5.1 Assumptions and dependencies
- 5.2 Acronyms and abbreviations

Reasons motivating the existence of the product

Identifies the product and application domain

Defines system's relationship to other products. Describes external interfaces: system, user, hardware, software

Summary of major functions

General characteristics of the intended groups of users, including those that may influence usability (e.g., disabilities, expertise)

Anything that will limit the developer's options (e.g., regulations, reliability, criticality, hardware limitations, interfaces, etc.)

All the requirements go in here (i.e., this is the body of the document); the standard provides different strategies for the organization

Verification methods planned to qualify the software

Factors not under the control of the software that may affect requirements

Section 3 — Requirements

Adapted from ISO/IEC/IEEE 29148 (Nov 2018)

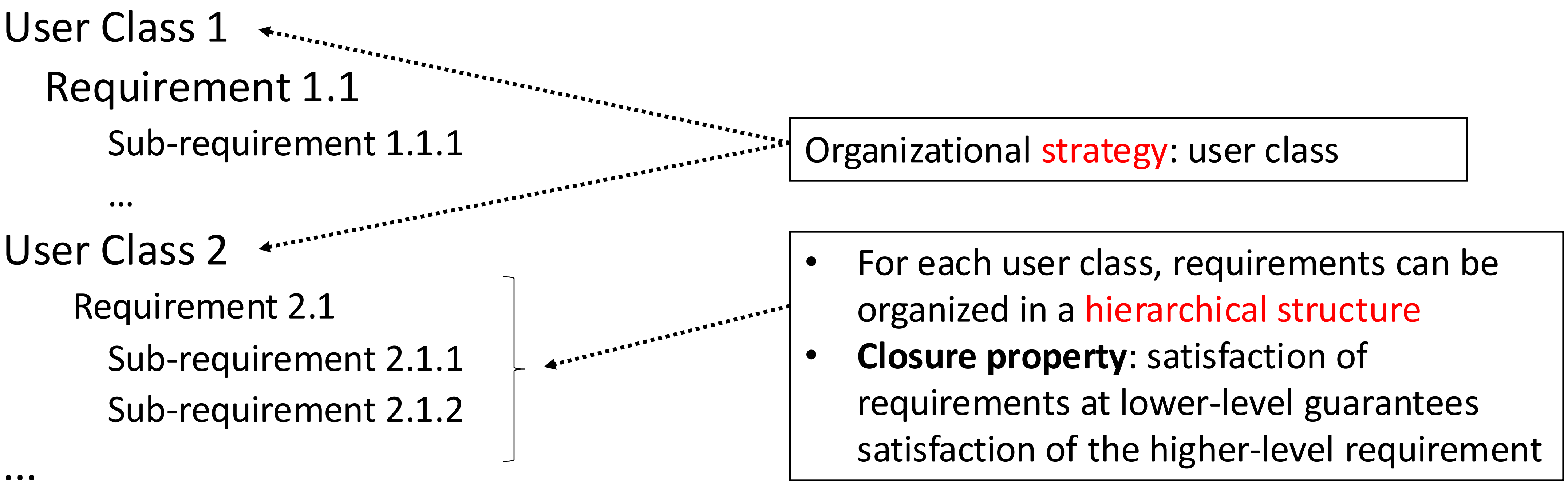
3.1 Functions (functional requirements)

- Fundamental actions that must occur in the software in accepting/processing the inputs and in generating the outputs
  - Validity checks
  - Sequence of operations
  - Responses to abnormal situations (e.g., overflow, failures, error handling)
  - Effect of configuration parameters
  - Input/output relationships
- Functional requirements can be organized by mode, user class, feature, etc.
- Functional requirements can be hierarchical (partition into sub-requirements)

Section 3 — Requirements

Adapted from ISO/IEC/IEEE 29148 (Nov 2018)

3.1 Functions (example)



User Class 1

Requirement 1.1

Sub-requirement 1.1.1

…

User Class 2

Requirement 2.1

Sub-requirement 2.1.1

Sub-requirement 2.1.2

…

Organizational strategy: user class

- For each user class, requirements can be organized in a hierarchical structure
- **Closure property**: satisfaction of requirements at lower-level guarantees satisfaction of the higher-level requirement

Section 3 — Requirements

Adapted from ISO/IEC/IEEE 29148 (Nov 2018)

3.2 Performance requirements

3.3 Usability requirements

… etc.

Essentially, all NFR we consider high priority for our system, grouped by type

Section 3 — Requirements
Adapted from ISO/IEC/IEEE 29148 (Nov 2018)

3.6 Design constraints

- Constraints on design decisions imposed by:
  - Domain-specific standards, regulatory documents
  - Other project limitations (as specified in Sec. 2)

3.7 Software system attributes

- Includes the required quality attributes of the product (system-level):
  - Reliability — required reliability level at the time of delivery
  - Availability — required availability level at the time of delivery
  - … etc.

Section 3 — Requirements

Adapted from ISO/IEC/IEEE 29148 (Nov 2018)

3.9 Supporting information

- Additional supporting information to be considered includes:
  - Sample input/output formats
  - Background information that can help the readers
  - Description of the problem(s) to be solved

Target qualities for a RASD (1)

- Completeness
  - **w.r.t. goals**: the requirements are sufficient to satisfy the goals under given domain assumptions

    $$\text{Req and Dom} \models \text{Goals}$$

    - all Goals have been correctly identified, including all relevant quality goals
    - Dom represent valid assumptions; incidental and malicious behaviours have been anticipated
  - **w.r.t. inputs**: the required behavior is specified for all possible types of inputs
  - **w.r.t. structure**: document does not contain TBDs

Target qualities for a RASD (2)

- Precision
  - Requirements should have a level of detail sufficient for software design, development, and verification of the software release
- Pertinence
  - each requirement or domain assumption is needed for the satisfaction of some goal
  - each goal is truly needed by the stakeholders
  - the RASD does not contain items that are unrelated to the definition of requirements (e.g., design or implementation decisions)
- Consistency
  - no contradiction in formulation of goals, requirements, and assumptions

Target qualities for a RASD (3)

- Unambiguity
  - unambiguous **vocabulary**: every term is defined and used consistently
  - unambiguous **assertions**: goals, requirements and assumptions must be stated clearly in a way that precludes different interpretations
  - unambiguous **responsibilities**: the split of responsibilities between the software-to-be and its environment must be clearly indicated
  - **verifiability**: a process exists to test satisfaction of each requirement

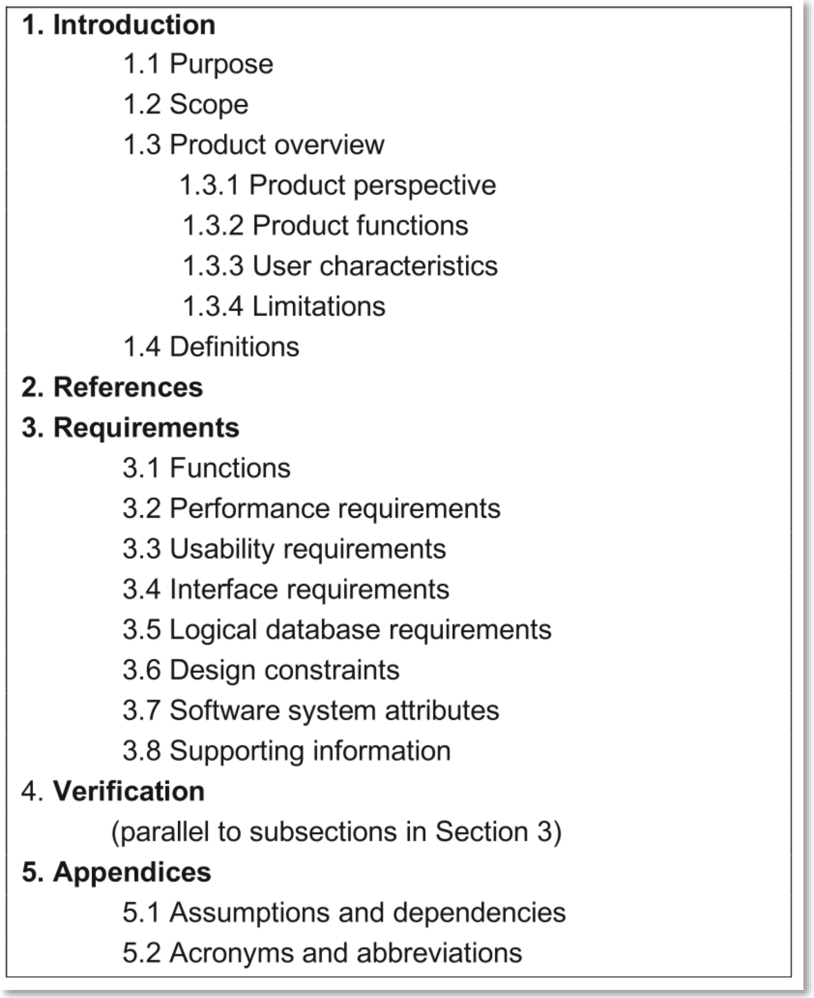
Target qualities for a RASD (4)

- Feasibility
  - the goals and requirements must be technically realizable within the assigned budget and schedules
- Comprehensibility
  - must be comprehensible by all in the target audience
- Good Structuring
  - e.g., highlights links between goals, requirements and assumptions
  - every item must be defined before it is used
- Modifiability
  - must be easy to adapt, extend or contract through local modifications
  - impact of modifying an item should be easy to assess

Target qualities for a RASD (5)

- Traceability
  - must indicate sources of goals, requirements and assumptions
  - must link requirements and assumptions to underlying goals
  - facilitates referencing of requirements in future documentation (design, test cases, etc.)

IEEE structure vs our RASD template



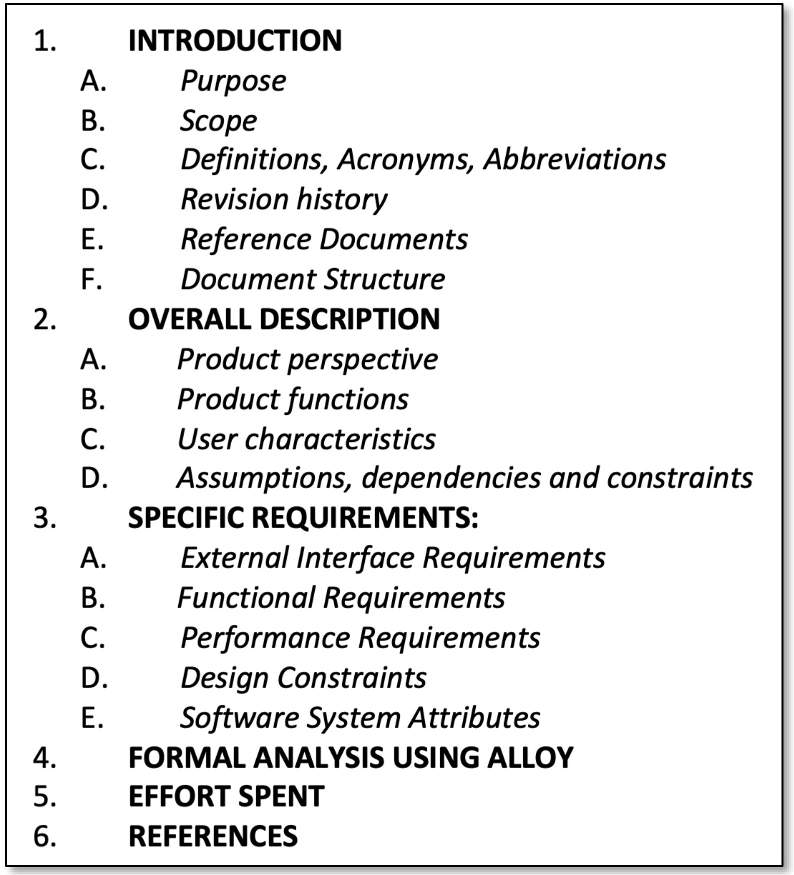
1. Introduction
- 1.1 Purpose
- 1.2 Scope
- 1.3 Product overview
  - 1.3.1 Product perspective
  - 1.3.2 Product functions
  - 1.3.3 User characteristics
  - 1.3.4 Limitations
- 1.4 Definitions

2. References

3. Requirements
- 3.1 Functions
- 3.2 Performance requirements
- 3.3 Usability requirements
- 3.4 Interface requirements
- 3.5 Logical database requirements
- 3.6 Design constraints
- 3.7 Software system attributes
- 3.8 Supporting information

4. **Verification**
- (parallel to subsections in Section 3)

5. Appendices
- 5.1 Assumptions and dependencies
- 5.2 Acronyms and abbreviations



1. **INTRODUCTION**
   - A. *Purpose*
   - B. *Scope*
   - C. *Definitions, Acronyms, Abbreviations*
   - D. *Revision history*
   - E. *Reference Documents*
   - F. *Document Structure*
2. **OVERALL DESCRIPTION**
   - A. *Product perspective*
   - B. *Product functions*
   - C. *User characteristics*
   - D. *Assumptions, dependencies and constraints*
3. **SPECIFIC REQUIREMENTS:**
   - A. *External Interface Requirements*
   - B. *Functional Requirements*
   - C. *Performance Requirements*
   - D. *Design Constraints*
   - E. *Software System Attributes*
4. **FORMAL ANALYSIS USING ALLOY**
5. **EFFORT SPENT**
6. **REFERENCES**

In which sections do we include all we have learnt about requirements?

1. INTRODUCTION

- A. **Purpose**: here we include the goals of the project
- B. **Scope**: here we include the analysis of the world and of the shared phenomena
- C. **Definitions, Acronyms, Abbreviations**
- D. **Revision history**
- E. **Reference Documents**
- F. **Document Structure**

In which sections do we include all we have learnt about requirements?

2. OVERALL DESCRIPTION

A. **Product perspective**: here we include scenarios and further details on the shared phenomena and a domain model, possibly expressed through class diagrams and state diagrams
B. **Product functions**: here we include the most important requirements/categories of use cases
C. **User characteristics**: here we include anything that is relevant to clarify their needs
D. **Assumptions, dependencies and constraints**: here we include domain assumptions

In which sections do we include all we have learnt about requirements?

3. SPECIFIC REQUIREMENTS: Here we include more details on all aspects in Section 2 if they can be useful for the development team.

A. External Interface Requirements

A.1 User Interfaces

A.2 Hardware Interfaces

A.3 Software Interfaces

A.4 Communication Interfaces

B. Functional Requirements: use case diagrams, use cases and related sequence and activity diagrams, and mapping on requirements

C. Performance Requirements

In which sections do we include all we have learnt about requirements?

D. Design Constraints

- D.1 Standards compliance
- D.2 Hardware limitations
- D.3 Any other constraint

E. Software System Attributes

- E.1 Reliability
- E.2 Availability
- E.3 Security
- E.4 Maintainability
- E.5 Portability

In which sections do we include all we have learnt about requirements?

4. FORMAL ANALYSIS USING ALLOY:

- Brief presentation of the main objectives driving the formal modeling activity
- the model itself
- what can be verified with it, why what is verified is important given the problem at hand
- To show the soundness and correctness of the model, this section can show some worlds obtained by running it, and/or the results of the checks performed on meaningful assertions

5. EFFORT SPENT: In this section you will include information about the number of hours each group member has worked for this document

6. REFERENCES

A note on traceability

- Use cases are related to some requirements
- Keep track of this relationship through proper identifiers
    - E.g., RE.3 is associated with UC.3.1 and UC.3.2
- We may also have use cases that refer to multiple requirements
    - E.g., UC.3.1 may refer also to RE.2
        - …even though the main relationship is with RE.3
    - Make this explicit in the presentation
        - E.g., you could build a traceability matrix

Traceability matrix

Raw ID	Goal ID	Req ID	Use Case ID	Comments
r1	G.1	RE.3	UC.3.1	
r2	G.1	RE.2	UC.3.1	

- This may grow during the development process, example:

Raw ID	Goal ID	Req ID	Use Case ID	Test case ID	Comments
r1	G.1	RE.3	UC.3.1	TC.3.1.1	
r2	G.1	RE.2	UC.3.1		

Homework

- Review the RASD available on Webeep, direct link
  - https://webeep.polimi.it/pluginfile.php/1302807/mod_folder/content/0/ProjectToBeReviewed/RASD.pdf
  - It refers to the assignment described in this document: https://webeep.polimi.it/pluginfile.php/1302807/mod_folder/content/0/ProjectToBeReviewed/Assignment_RDD_2023-2024.pdf
- Answer to the questionnaire here (one set of answers per group)
  - https://forms.office.com/e/fXseHm1Wjj
  - if you are doing the R&DD project , keep the **same groups as for R&DD** projects
    - If you are not doing the R&DD project, you can create a new group (even cross-class), but you will have to keep the same group also for the DD homework
  - We will assign up to 1 point to clear and convincing answers
- **Deadline**: October 30th at 23.59 (Rome time)
- Answers will be used as basis for discussion during the lab of October 31st

Homework — important notes

- Focus more on content rather than structure
- Your critical review should identify weaknesses and strengths especially considering our “target qualities for a RASD”
- Pure AI-generated content will not be considered acceptable.
  - Value your reasoning and expressive capabilities! You are the ones who will build machines, not vice versa!