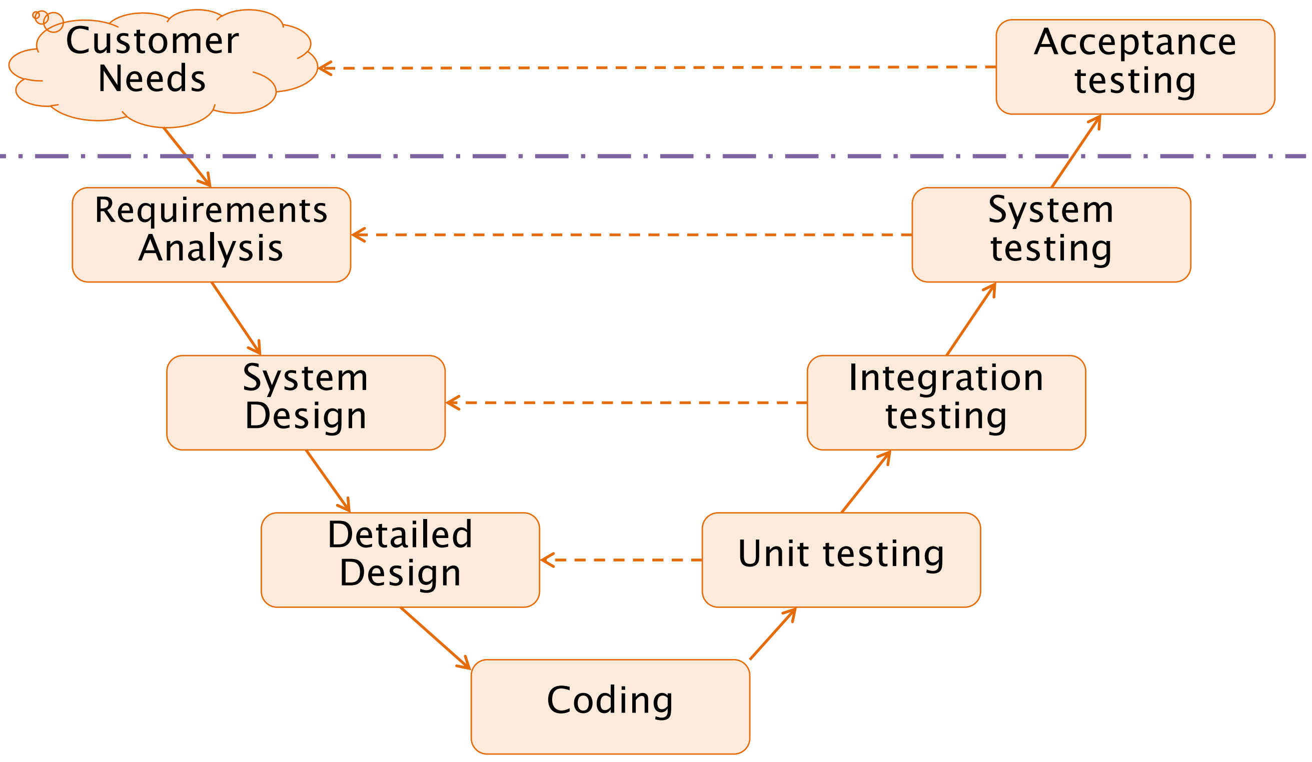
Requirements Engineering

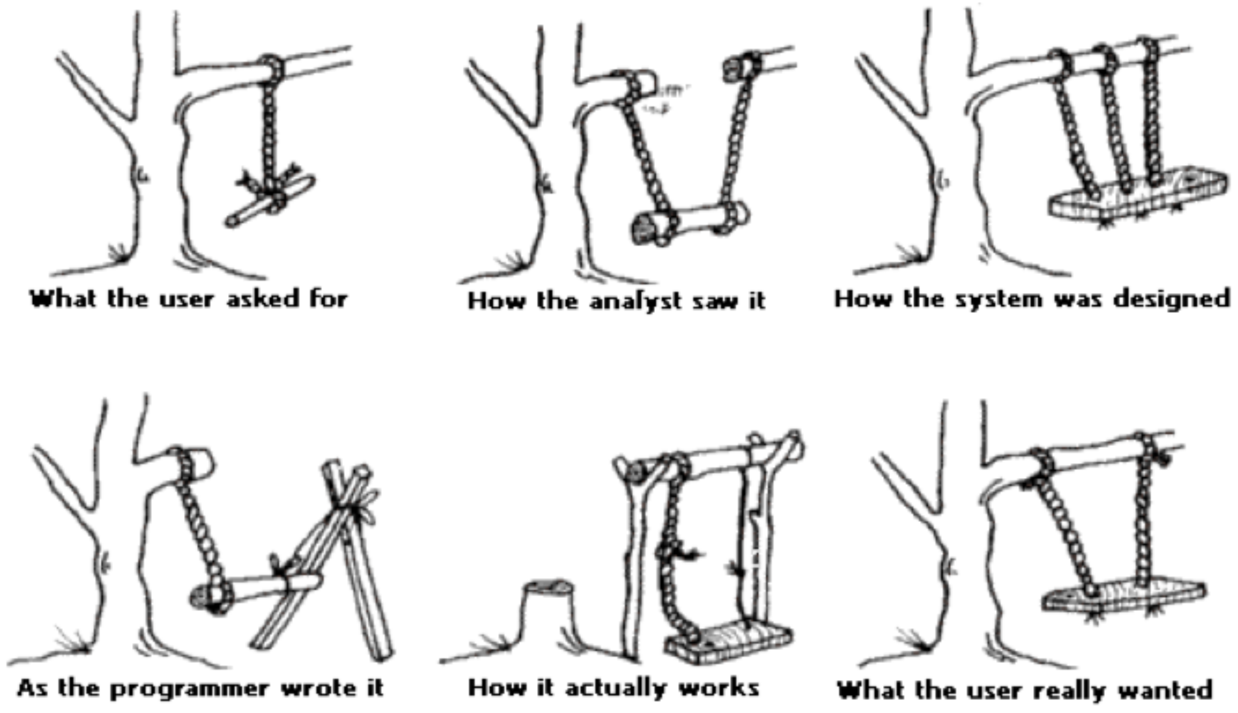
https://bit.ly/PolitoSIA

Version 1.3.1 - 22 October 2020

Software Development



Software Development Reality



What the user asked for
How the analyst saw it
How the system was designed
As the programmer wrote it
How it actually works
What the user really wanted

Requirements engineering

- The process of establishing the services that the customer requires from a system and the constraints under which it operates and is developed.
- The requirements themselves are the descriptions of the system services and constraints that are generated during the requirements engineering process.

Activities in req. engineering

- Elicitation
- Analysis
- Formalization
- V&V (verification and validation)

Elicitation

- Elicit
  1. : to call forth or draw out (something, such as information or a response)
  2. : to draw forth or bring out (something latent or potential)
     - From latin *elicere*

https://www.merriam-webster.com/dictionary/elicit

- Aim to understand stakeholder needs and identify potential solutions that may meet those needs

Elicitation

- Use different techniques
  - Brainstorming
  - Document analysis
  - Focus Group
  - Interface Analysis
  - Interviews
  - Observation (job shadowing)
  - Prototyping
  - Requirements workshops
  - Survey/questionnaire

Stakeholder

- A party that has an interest or concern in the system
- Includes:
  - Users of the system
  - Procurer of the system development
  - Anybody affected by the system usage
    - Ethical
    - Social
    - Etc…

Requirements vs. Design

- Requirements:

  What the system should do

- Design:

  How the system is done

What is a requirement?

- Ranges from a high-level abstract statement of a service or of a system constraint to a detailed mathematical functional specification.
- Requirements may serve dual function
  - Basis for a bid for a contract - therefore must be open to interpretation
  - Basis for the contract itself - therefore must be defined in detail

Types of requirement

- User requirements
  - Statements in natural language plus diagrams of the services the system provides and its operational constraints. Written for customers.
- System requirements (a.k.a. developer requirements)
  - A structured document setting out detailed descriptions of the system’s functions, services and operational constraints. Defines what should be implemented so may be part of a contract between client and contractor.

Definitions and specifications

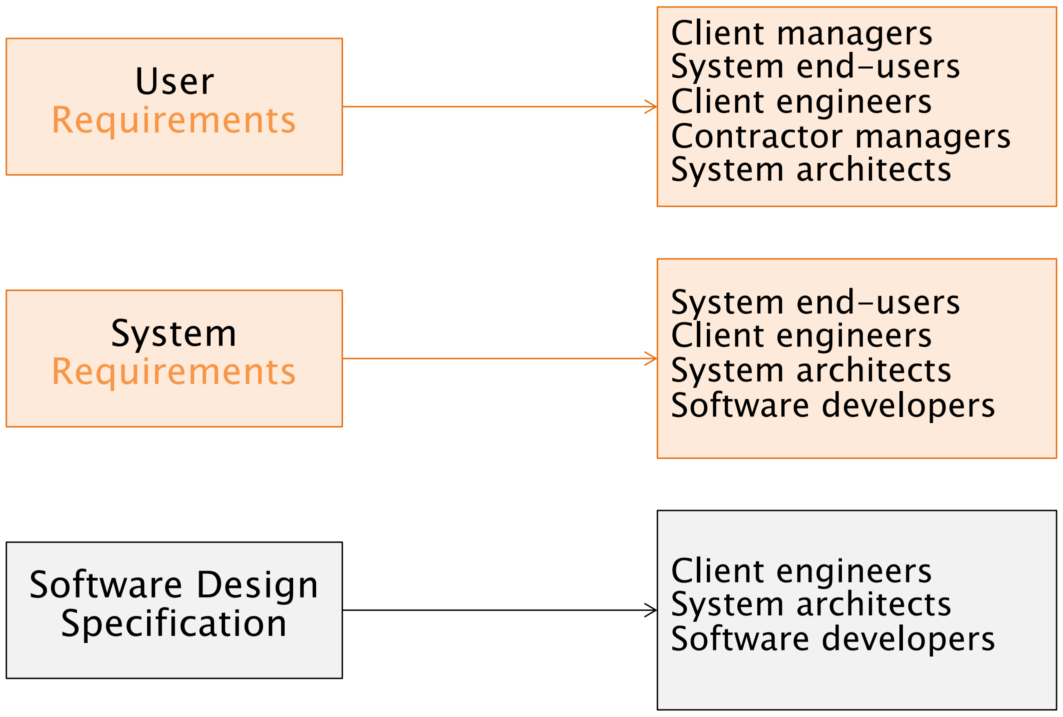
User requirement definition

The software must provide a means of representing and accessing external files edited by other tools

System requirements specification

1.1 The user should be provided with facilities to define the type of external files
1.2 Each external file type may have an associated tool which may be applied to the file
1.3 Each external file type may be represented as a specific icon on the user's display
1.4 Facilities should be provided for the icon representing an external file type to be defined by the user
1.5 When a user selects an icon representing an external file the effect of that selection is to apply the tool associated with the external file type to the file represented by the selected icon

Requirements readers



Requirements features

- Correct
- Unambiguous
- Complete
- Consistent
- Ranked for importance and/or stability
- Verifiable
- Modifiable
- Traceable

Correct

- Every requirement stated is one that the software shall meet
- Customer or users can determine if the requirement correctly reflects their actual needs
  - Traceability makes this easier

Unambiguous

- Every requirement has only one interpretation
- Each characteristic of the final product must be described using a single unique term
- Both to those who create it and to those who use it.

Complete

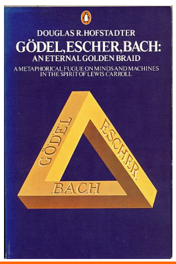
- Include all significant requirements
  - Address external requirements imposed by system specification
- Define response to all realizable inputs
  - Both correct or incorrect
- Define all terms and unit of measure

Internally Consistent

- No subset of requirements is in conflict
  - Characteristics of real-world objects (e.g. GUI
  - Logical or temporal
  - Different terms for the same object

Completeness vs. consistency

- In principle, requirements should be both complete and consistent.
  - Complete: they should include descriptions of all facilities required.
  - Consistent: there should be no conflicts or contradictions in the descriptions of the system facilities.
  - In practice, it is impossible to produce a document that is both complete and consistent
    - See: **Gödel's incompleteness theorems**



Ranked

By:

- Value
- Necessity
  - Essential
  - Conditional
  - Optional
- Stability (in the future)

Verifiable

- There exists some finite cost-effective process with which a person or machine can check that the software product meets the requirement.
  - Ambiguous requirements are not verifiable

Modifiable

- structure and style such that any changes can be made easily, completely, and consistently while retaining the structure and style
  - Well structured
  - Non redundant
  - Separate requirements

Traceable

- Backward
  - explicitly referencing source in earlier documents
- Forward
  - unique name or reference number

Defects in requirements

- Omission (incompleteness)
- Inconsistency (contradiction)
- Ambiguity
- Incorrect Fact
- Extraneous Information
  - Over-specification (design)
- Lack of structure
- Redundancy

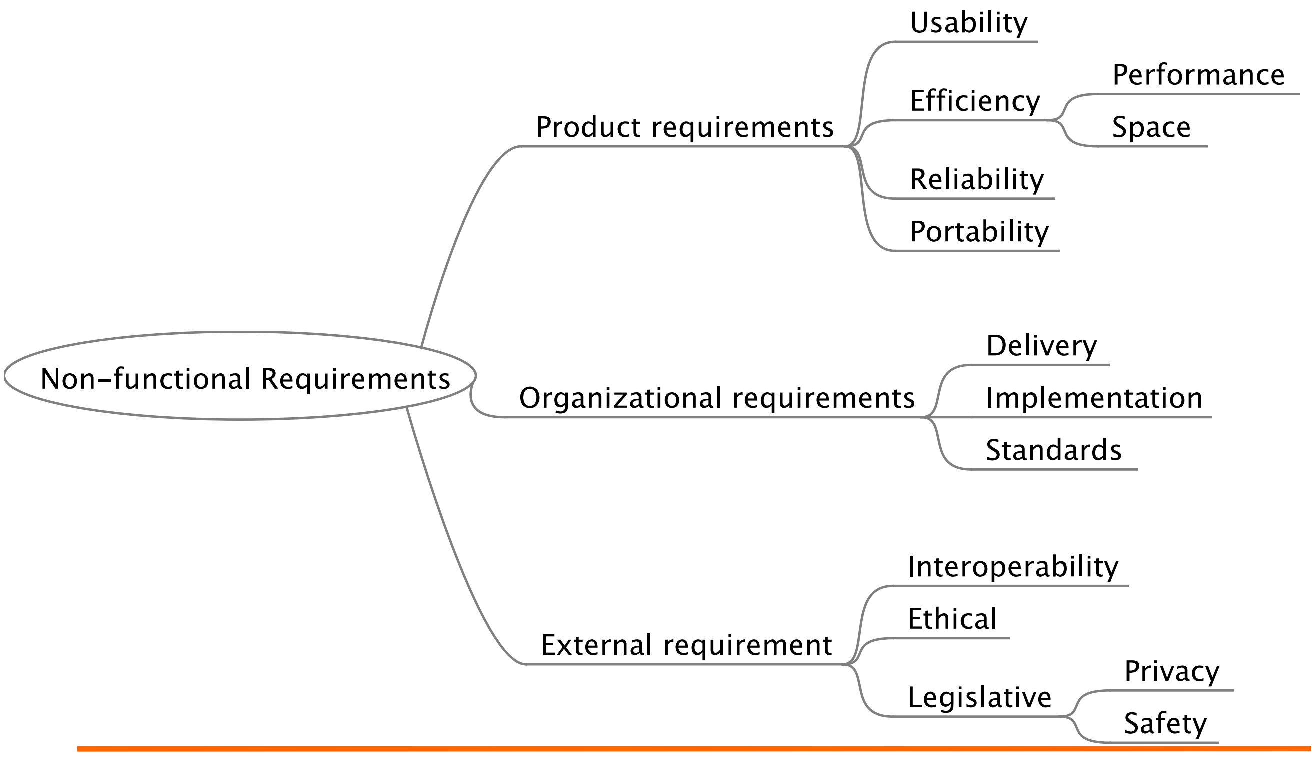
Functional vs. Non-Functional

- Functional requirements (FR)
  - Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
- Non-functional requirements (NFR)
  - A.k.a. Quality requirements
  - constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.
- Domain requirements
    - Requirements that come from the application domain of the system and that reflect characteristics of that domain.

Non-functional requirements

- They define system properties and constraints e.g. reliability, response time and storage requirements.
  - Constraints are I/O device capability, system representations, etc.
- Process requirements may also be specified mandating a particular tool, programming language, or development method.
- Non-functional requirements may be more critical than functional requirements: if they are not met, the system is useless.

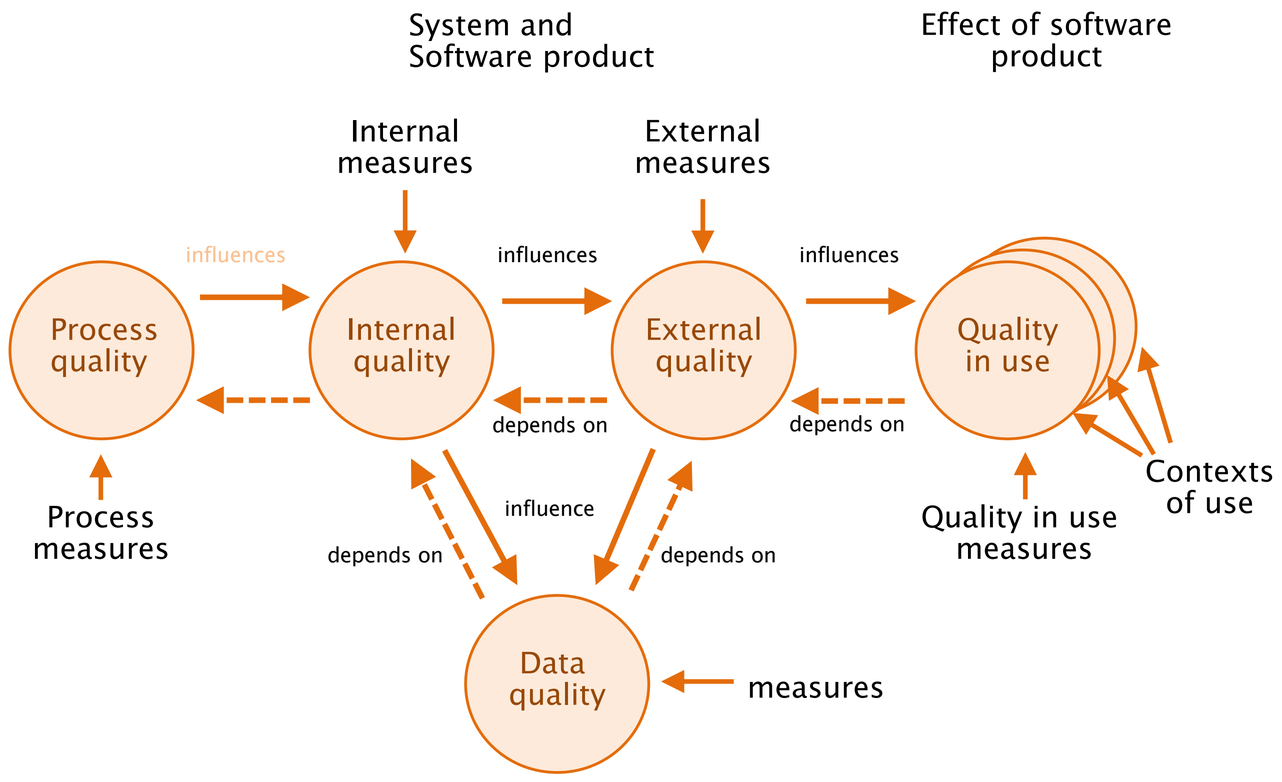
Non-functional requirements



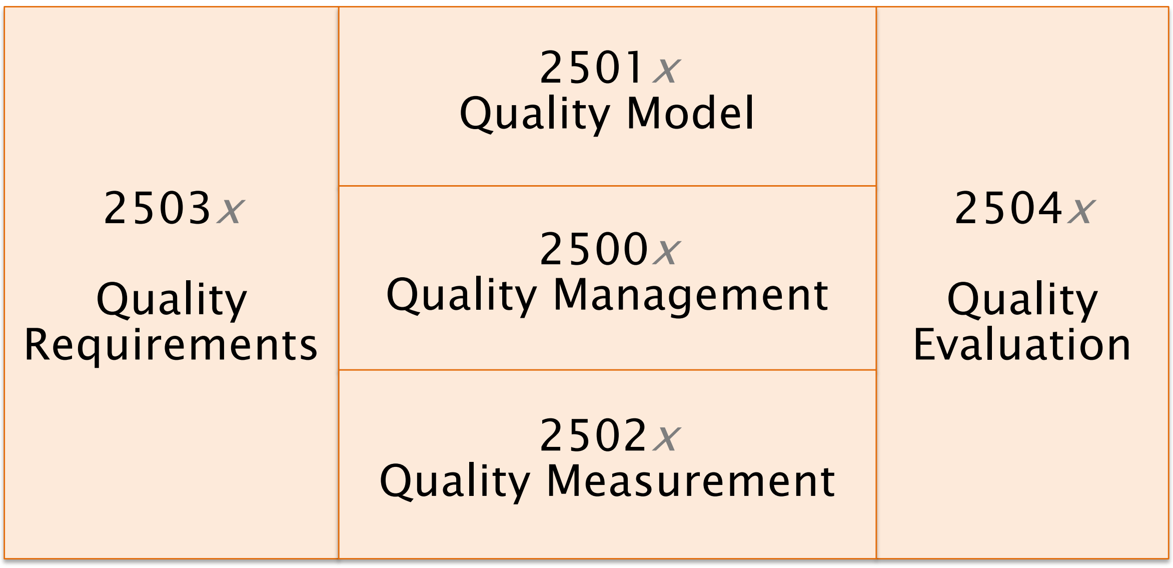
Non-functional req.: examples

- Product requirement
  - 8.1 The user interface for LIBSYS shall be implemented as simple HTML without frames or Java applets.
- Organisational requirement
  - 9.3.2 The system development process and deliverable documents shall conform to the process and deliverables defined in XYZCo-SP-STAN-95.
- External requirement
  - 7.6.5 The system shall not disclose any personal information about customers apart from their name and reference number to the operators of the system.

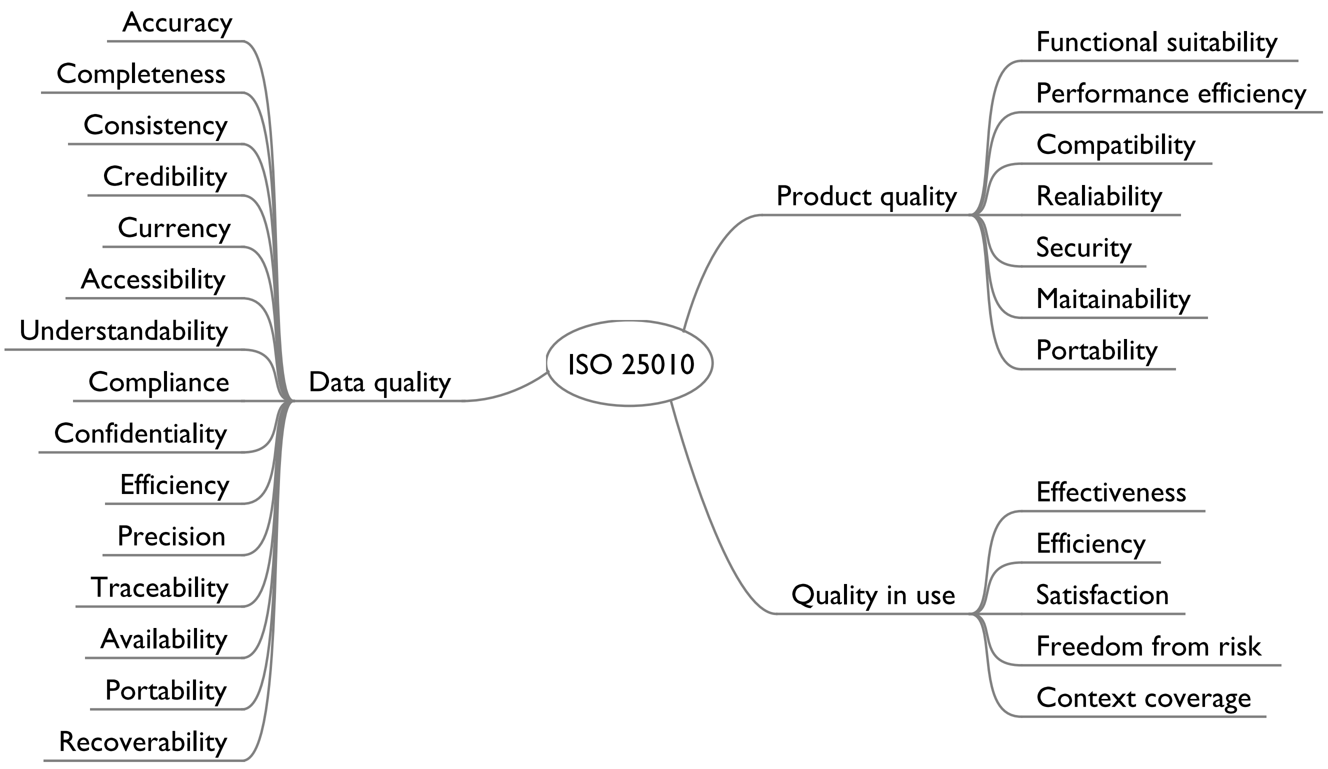
Software Qualities

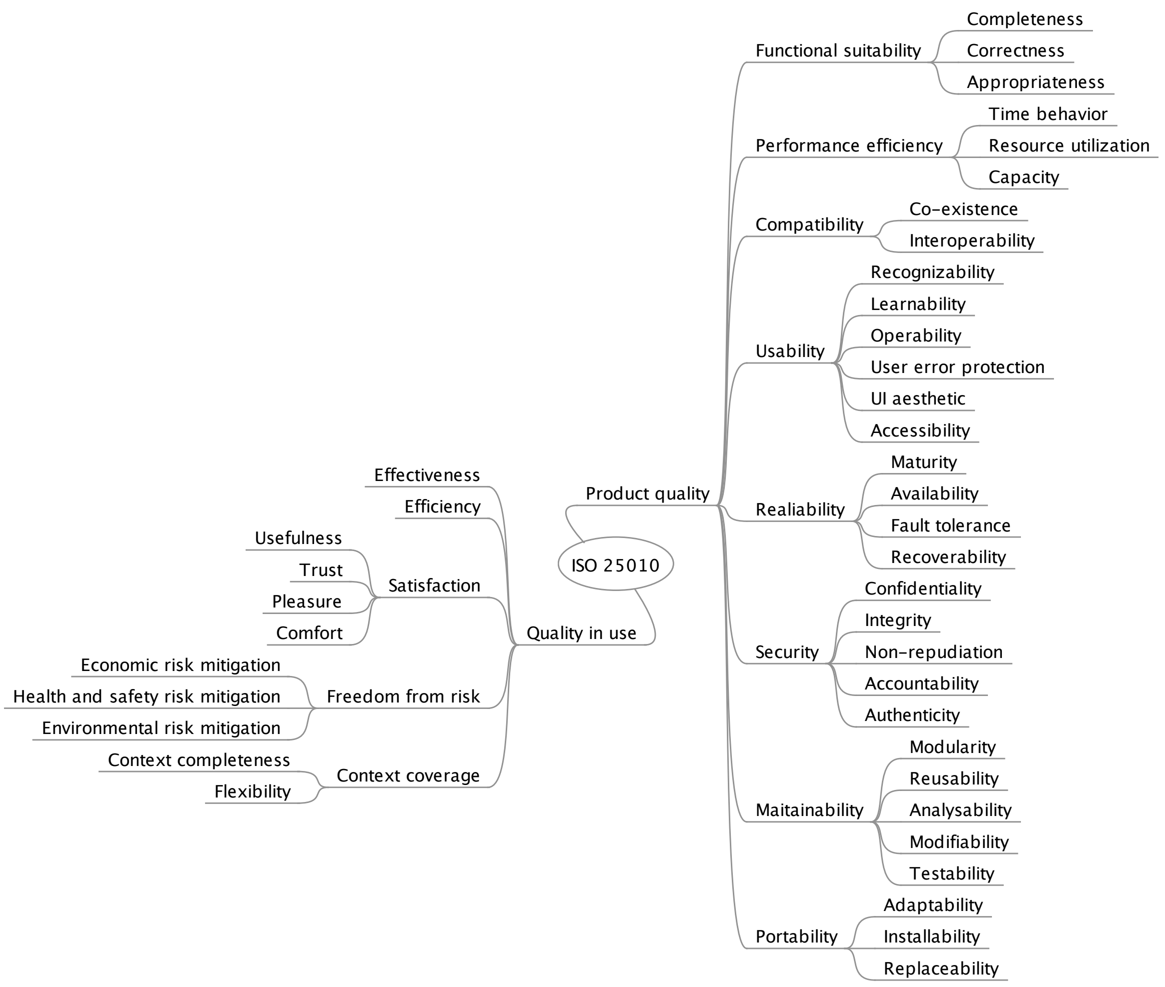


ISO SQuaRE – Standard Family



ISO 25010 – Quality model





Internal vs. External

- Internal features concern the static attributes of a software product
  - verified by review, inspection, simulation and/or automated tools
- External features concern the behavior of a system
  - verified and/or validated by executing the software product during testing and operation

Goals and requirements

- Non-functional requirements may be very difficult to state precisely and imprecise requirements may be difficult to verify.
- Goal
  - A general intention of the user such as ease of use.
- Verifiable non-functional requirement
  - A statement with a measure that can be objectively tested.
- Goals are helpful to developers as they convey the intentions of the system users.

Examples

- System goal
  - The system should be easy to use by experienced controllers and should be organised in such a way that user errors are minimised.
- Verifiable non-functional requirement
  - Experienced controllers shall be able to use all the system functions after a total of two hours training. After this training, the average number of errors made by experienced users shall not exceed two per day.

NFR measures

Property	Measure
Speed	Processed transactions/second
	User/Event response time
	Screen refresh time
Size	M Bytes
	Number of ROM chips
Ease of use	Training time
	Number of help frames
Reliability	Mean time to failure
	Probability of unavailability
	Rate of failure occurrence
	Availability
Robustness	Time to restart after failure
	Percentage of events causing failure
	Probability of data corruption on failure
Portability	Percentage of target dependent statements
	Number of target systems

Natural language

- Lack of clarity
  - Precision is difficult without making the document difficult to read.
- Requirements confusion
  - Functional and non-functional requirements tend to be mixed-up.
- Requirements amalgamation
  - Several different requirements may be expressed together.

The requirements document

- The requirements document is the official statement of what is required of the system developers.
- Should include both a definition of user requirements and a specification of the system/developer requirements.
- It is NOT a design document. As far as possible, it should set of WHAT the system should do rather than HOW it should do it

Users of requirements

System customers	Specify the requirements and read them to check that they meet their needs. They specify changes to the requirements
Managers	Use the requirements document to plan a bid for the system and to plan the system development process
System engineers	Use the requirements to understand what system is to be developed
System test engineers	Use the requirements to develop validation tests for the system
System maintenance engineers	Use the requirements to help understand the system and the relationship between its parts

IEEE requirements standard

- IEEE Std 830:1998
  - Superseded by ISO/IEC/IEEE 29148:2011
- Defines a generic structure for a requirements document that must be instantiated for each specific system.
  - Introduction.
  - Overall description.
  - Specific requirements.
  - Appendixes.
  - Index.

Req. document structure

- Preface
- Introduction
- Glossary
- User requirements definition
- System architecture
- System requirements specification
- System models
- System evolution
- Appendices
- Index

Organizing requirements

- Mode
- User class
- Object
- Feature
- Stimulus
- Functional hierarchy

Requirements Document lightweight

1. Purpose and scope
2. The terms used / Glossary
3. The use cases
4. The technology to be used
5. Other various requirements
6. Human backup, legal, political, organizational issues

Requirements Document

1. Purpose and scope

- What is the overall scope and goal?
- Stakeholders (who cares?)
- What is in scope, what is out of scope

organizational issues

Requirements Document

1. Purpose and scope
2. The terms used / Glossary
3. The use cases
4. The technology to be used
5. Other various requirements
6. Human backup, legal, political, organizational issues

Requirements Document

1. Purpose and scope
2. The terms used / Glossary
3. The use cases

- The primary actors and their general goals
- The business use cases (operations concepts)
- The system use cases

Requirements Document

1. Purpose and scope
2. The terms used / Glossary
3. The use cases
4. The technology to be used

- What technology requirements are there for this system?
- What systems will this system interface with, with what requirements?

R

- Development process
- Business rules
- Performance
- Operations, security, documentation
- Use and usability
- Maintenance and portability
- Unresolved or deferred

5. Other various requirements
6. Human backup, legal, political, organizational issues

R

- What is the human backup to system operation?
- What legal, what political requirements are there?
- What are the human consequences of completing this system?
- What are the training requirements?
- What assumptions, dependencies are there on the human environment?

6. Human backup, legal, political, organizational issues

Guidelines for requirements

- Define a standard format and use it for all requirements.
- Use language in a consistent way. Use shall for mandatory requirements, should for desirable requirements.
- Use text highlighting to identify key parts of the requirement.
- Avoid the use of computer jargon.

V&V of requirements

- Natural language, UML
    - Inspection, reading
        - By user, by developer
- UML
    - Some syntactic check by tools
- Formal language
    - Model checking

Tools

- RequisitePro, Doors, Serena RM
- Word, Excel
- UML tools
  - Powerpoint, Visio, specialized tools (StarUML)

References

- IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1998, Revision of IEEE Std 830-1993)
- ISO/IEC 250xx:2011 – Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE)

Key points

- Requirements set out what the system should do and define constraints on its operation and implementation.
- User requirements are high-level statements of what the system should do. User requirements should be written using natural language, tables and diagrams.
- System requirements are intended to communicate the functions that the system should provide.

Key points

- Functional requirements set out services the system should provide.
- Non-functional requirements constrain the system being developed or the development process.
- A software requirements document is an agreed statement of the system requirements.
- The IEEE standard is a useful starting point for defining more detailed specific requirements standards.