Content

- Requirements engineering
  - What is a requirement?
  - How to work with them?
  - What is a requirements specification?
  - Thoughts about requirements in the project
- About the base system (Rasmus Ros)

Definition

IEEE defines a **requirement** as

(1) A condition or capability **needed** by a user to solve a problem or achieve an objective

(2) A condition or capability that **must** be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document

(3) A documented **representation** of a condition or capability as in (1) or (2)

Abstraction level and target group

- User Requirements
- System Requirements
- Software Design Specification

Client engineers
System end-users
System architects
…

System end-users
system architects
Developers
…

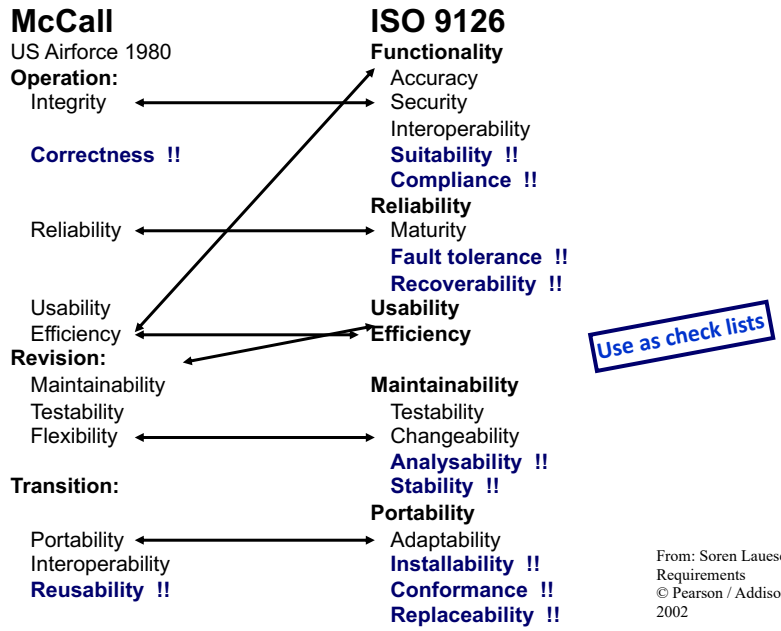
System architects
Software developers
…

- E.g. [*Sommerville,Software Engineering, **6th Ed**., 2001*]

Different types of requirements

- Functional requirements
- Quality requirements
  - ”limitations” for the product
  - Often affect the whole product
  - Often affect the architecture
  - Different types: product, organization, …

Quality factors



From: Soren Lauesen: Software Requirements

May be conflicting

For example

- Increased performance
  - May mean decreased maintainability
- Increased security
  - May mean decreased usability

Requirements engineering

- methods and techniques for
  - identification
  - documenting
  - validating
  - evolving
  - tracing
- requirements throughout the project!

What is Requirements Engineering?

- "Requirements engineering is a process that involves all the activities required to **create and maintain** a requirements **document**."
  "There are four generic [...] process activities [...] feasibility study, [...] elicitation and analysis, [...] specification, [...] validation"

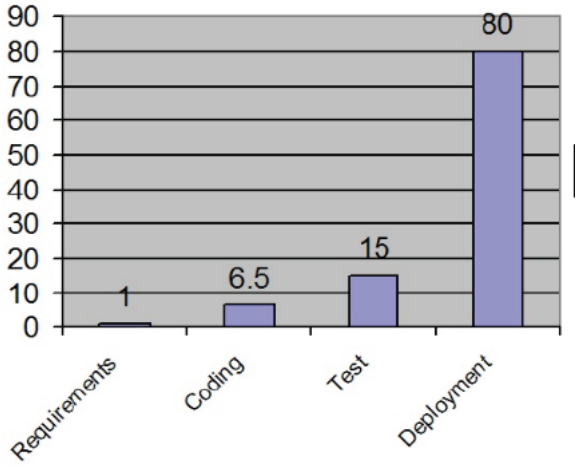
[*Sommerville, Software Engineering, 6th Ed., 2001*]

Why are requirements engineering important?

Standish Group Survey
Top 10 Challenges
(% of Responses)

1. Lack of User Input	12.8%
2. Incomplete Requirements	12.3%
3. Changing Requirements	11.8%
4. Lack of Executive Support	7.5%
5. Technology Incompetence	7.0%
6. Lack of Resources	6.4%
7. Unrealistic Expectations	5.9%
8. Unclear Objectives	5.3%
9. Unrealistic Time Frames	4.3%
10. New Technology	3.7%

Cost of errors



Pressman & Grady – one of many similar estimations

Why is RE difficult?

- Stakeholders don't know what they really want
- Stakeholders express requirements in their own terms
- Different stakeholders have conflicting requirements
- Organisational and political factors may influence
- The requirements change during the process

+ hard to know the result is "good enough"

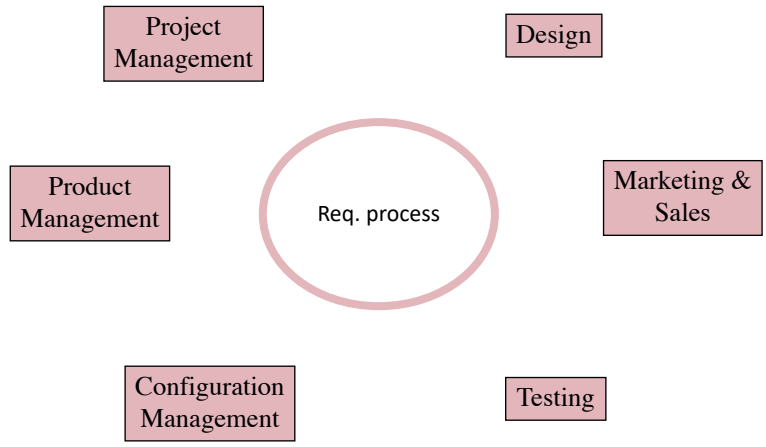
+ hard to find right level of abstraction

...

The RE process

- **Requirements elicitation**
- **Requirements analysis**
- **Requirements documentation**
- **Requirements validation**

Requirements-related processes



Elicit Requirements

- To discover the needs of the stakeholders
- The requirements are elicited through
  - Consultation with stakeholders
  - System documents
  - Domain knowledge
  - Market surveys

Validate Requirements

- To check that we have elicited and documented the right requirements
- Are the requirements ”sufficiently”
  - Correct?
  - Complete?
  - Consistent?
  - Unambiguous?
  - Realistic?
  - Verifiable?
  - ...

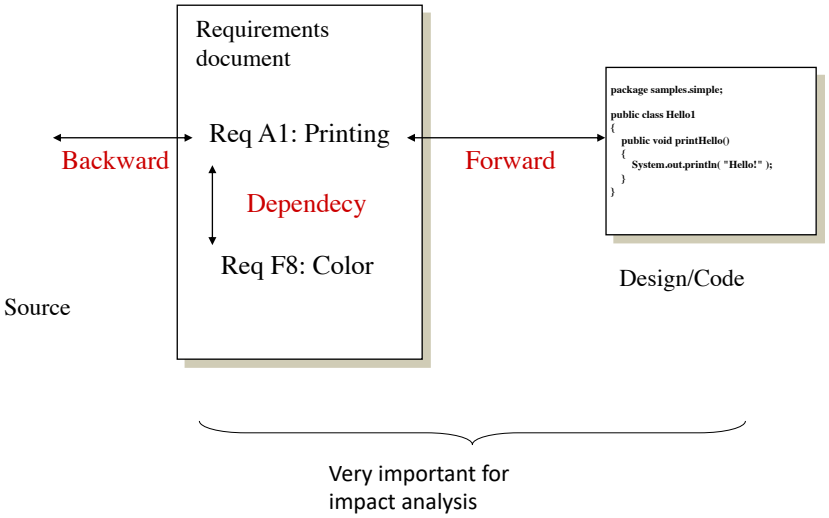
Requirements validation techniques

- Requirements reviews (inspections)
  - Systematic manual analysis of the requirements
- Prototyping
  - Using an executable model of the system to check requirements.
- Test-case generation
  - Developing tests for requirements to check testability
- Automated consistency analysis
  - Checking the consistency of a structured requirements description

Requirements change management

- Should apply to all proposed changes to the requirements
- Principal stages
  - Problem analysis. Discuss requirements problem and propose change
  - Change analysis and costing. Assess effects of change on other requirements
  - Change implementation. Modify requirements document and other documents to reflect change

Traceability



Requirements specification

- goal
  - documents the customer's requirements to the system, such that developers know exactly what is required from the system
- quality in requirements specification
  - correctness
  - completeness
  - verifiable
  - unambiguous
  - consistent
  - traceable
  - organized
  - motivated

Problems with natural language requirements

- 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

Requirements division

- Structure of GS:SRS
  - background and goals
  - terminology
  - system requirements
    - functional requirements, divided in:
      - overall requirements valid for all services
      - divided up by services => one section in SRS per service
      - interface/interaction requirements in special section
    - interface towards hardware
    - quality requirements (non-functional requirements)
      - usability, extendability, trustable, ...
  - project requirements
    - development environment, testing, releasing, ...

Requirements numbering

- to allow things to be traceable to requirements:
  - running numbers pre-fixed by section number
  - why is this a good idea?
- for example same structure as example SRS
- (do **NOT** change requirements in older documents - remove the old one and add a new one)

	Req 1	Req 2	Req 3	Req 4	Req 5	
Test 1		x	x			
Test 2			x			
Test 3	x			x		
Test 4		x				
Test 5						
…						

All requirements should be tested
All tests should test at least one requirement

Working with the SRS (proposed way of working)

- SG develops a ”template” and gives instructions on how requirements should be written
- project groups analyze and develop requirements (distribution of work)
- SG puts requirements together and fixes terminology
- Make an ”interaction table” and check that ”all” interactions (sufficently) are specified
- TG acts as informal reviewers
- SVVS depends on SRS => synchronization problems?

Some thoughts about the SRS

- Try to think about all requirements – i.e. try to be complete
  - Normal ways of using the system
  - Non-normal ways of using the system
  - Wrong ways of using the system
  - “Interections” between different main functions of the system
    - both “normal” and “non normal”
- Develop requirements in several steps
  - involve many group members
  - …but get a consistent set of requirements
  - Inspections and other ways of getting feedback important