Requirements

Advanced Programming

Software lifecycle

- Stages:
  - **Requirement elicitation**: from customers
  - **Analysis**: purposes formalized in a consistent and coherent way
  - **Design:** a representation of entities and their relations and/or status (often graphical)
  - **Implementation**: code developed
  - **Testing**: system tested for correctness
  - **Maintenance**: bug fixes, new features, new versions

Requirement

- A requirement is a **documented need** of what a given product or service should be or perform
- Requirement are goals to achieve

Types of Requirements 1/4

- Traditionally, there are three major kinds of requirements:

  Functional Requirements

  Non-functional requirements

  Constraints

Types of Requirements 2/4

- **Functional requirements**: describe the interactions between the system and its environment independently from any implementation
  - Example: The watch system must display the time based on its location

Types of Requirements 3/4

- **Non-functional requirements:** User visible aspects of the system not directly related to its functional behavior
  - The response time must be less than 1 second
  - The accuracy must be within a second
  - The watch must be available 24 hours a day except from 2:00am-2:01am and 3:00am-3:01am

Types of Requirements 4/4

- **Constraints (“Pseudo requirements”)**: Imposed by the client or the environment in which the system will operate
  - The system must operate with Linux OS
  - The implementation language must be COBOL

What is usually not a requirement?

- It is desirable that none of these above are constrained by the client.
  - System structure, implementation technology
  - Development methodology
  - Development environment
  - Implementation language (in some cases it can be a pseudo requirement)

Good requirements are ... (1/6)

- Understandable
  - No confusion and misunderstanding
    - domain-specific language and terms confuse developers
    - Technical terms confuse external stakeholders
  - Using short, declarative statements
  - Examples, figures, and tables for clarification
- Non-prescriptive
  - Stating what customer wants, not how programmer will do it

Good requirements are ... (2/6)

- Correct and complete
  - Exhaustive list of requirements
- Concise
  - Facilitating customer's validation of requirements
  - Prevents developers from skimming through info
  - Use KISS (Keep It Simple, Stupid) principle

Good requirements are ... (3/6)

- Consistent language
  - "Shall" statement is a "contract" or mandatory
  - "Should"/"may" statement is desirable but optional
- Consistent
  - No contradiction between requirements

Good requirements are ... (4/6)

- Unambiguous & testable
  - Writing test cases during requirements elicitation
    - Involve customers early
  - Specify a quantitative description for each adverb and adjective
  - Replace pronouns with specific names of entities
  - Every noun is defined in exactly one place in the requirement document

Good requirements are ... (5/6)

- Traceable
  - Requirements assigned with unique identifiers
  - Easing the future reference to requirements
- Ranked for importance and stability
  - Should be decided together by team and stakeholders
  - Requirements negotiation process for determining:
    - Realistic priorities
    - How likely a requirement will change

Good requirements are ... (6/6)

- Feasible
  - Infeasible requirements found in elicitation phase
    - To be explained by stakeholder immediately
  - Infeasible requirements found in analysis phase
    - Stakeholder notified and requirements document updated

The term "specification"

- A specification is a solution to given requirements
  - Agreed with the user/customer/manufacturer/producer of a system
- The specification may also include both system's requirements and test requirements (e.g. acceptance test in eXtreme Programming)

Requirements major components

- What
- Effort
- Priority
- Risk

Requirements major components

- What?
  - Example: “Be able to configure all the variables, like user name, password, access level”
- Effort required?
  - Example: “1 week”

Requirements major components

- Priority?
  - Example: “Configuring the password is the most important thing, then access level, then user name”
- Risk?
  - Example: “Are the developers familiar with encryption technology?”

Key Principles

- Separate the “what” from the “how”
  - What: “Information on ordered books shall be persistently stored”
  - How: “Information on ordered books shall be stored using Database A”

Next slides

- Requirements analysis