Software Engineering in Practice
Software requirements

Diomidis Spinellis
Department of Management Science and Technology
Athens University of Economics and Business

dds@aueb.gr
http://www.dmst.aueb.gr/dds
@CoolSWEng

2023-03-13

Team presentations

- Extraction of requirements for an important-popular OSS project.
- Specific objectives:
- Identification of stakeholders
- Identification of functional and non-functional requirements
- Study of the impact of change in requirements
- Teams

Attributes of requirements

- Verifiable
- Priority
- State
- Identifier

Separation of requirements

- Axis:
- Requirements from the product
- Requirements from the development process
- Axis:
- Functional requirements
- Non-functional requirements
- Axis:
- System requirements
- Software requirements
- Emerging requirements (system-wide)

The requirements elicitation process

- Continues throughout the whole software life cycle
- Places the requirements under configuration control

- Adapts to the software and product

Who have a say in requirements

- Users
- Customers
- Market analysts
- Regulation authorities
- Software engineers (very important)

Sources of requirements

- Project objectives
- Domain knowledge
- Those who have a say
- Business rules
- Operational environment (e.g. non-functional requirements)
- Organizational environment

Ways to elicit requirements

- Interviews
- Use case scenarios
- Prototypes
- Facilitated meeting
- Observation
- User stories
- Others
- Analysis of competition
- Issue databases, telemetry
- User comments
- Analysis of domain knowledge sources — ## Why analyze requirements
- To identify conflicting ones
- To specify the boundaries between software and its environment
- To develop system specifications into software specifications

Analysis: categorization

- Functional / Non-functional
- Origin
- From high-level specification
- From emerging requirement
- From direct source
- Process or product
- Width

- Stability or instability
- (why is this important?)

Analysis: abstracted modeling

- E.g. UML
- Case diagram
- Data flow model
- State model
- Object model
- Data model
- …
- Model selection
- Problem type
- Specialization of implementation engineer
- Requirements from software development process — ## Requirements analysis: other elements
- Architecture design
- Requirements allocation
- Requirements negotiation (and prioritization)
- Formal analysis

Determination of specifications

- System determination
- System specifications determination
- Software specifications determination

Validation of requirements

- Review
- Prototype implementation
- Model validation
- Acceptance tests

Practical issues

- The requirements specification process is iterative
- Requirements should be placed under version control
- The most important element of a requirement is its identifier
- The provenance information of a requirement should be documented

Assignment (Architecture)

- Select an (important-popular) OSS project and examine elements of its design:
  - look for principles of good software design
  - identify topics - design factors that become particularly important in the project
  - select a design strategy and matching notation to represent a static view of the software design. You can use abstraction, one of the principles of good software design, to communicate those entities that you consider more important.

Advice for the assignment:

- Consult the documentation and the source code of the OSS project (for OSS project hosted at GitHub you can download their source code in zip format).
- Combine high-level information (what the system offers to users) with structural elements (packages, components) to achieve your goal.
- Select OSS projects that use technologies familiar to you (e.g. Java).

Distribution License