Introduction to Requirements Engineering

Software Requirements and Design
CITS 4401
Lecture 16

Lecture Overview

- Why is RE hard?
- 4 stages of RE
- Different types of requirements

Requirements Example 1

Write a program that will read in a list of 100 positive integers, sort them into ascending order, display the sorted list and display the average of those values

Requirements Example 2

Develop an automated system that will allow us to process orders at least 24 hours sooner, on the average, and will allow us to ship our products to customers at least 3 days sooner than currently
Requirements Example 3

Develop the SW that will allow the Z-676 airliner to land itself, without pilot intervention, at major airports

Requirements Example 4

Develop a new personal productivity product for small computers that will sell at least one million copies at a retail price of at least $200

Why is Requirements Engineering Difficult?

- SW Engineering is a creative, problem solving activity
- Real customers are not sure what they want
- Large SW systems have many different stakeholders with different needs and priorities
- Real developers are not sure how to build it
- Real requirements creep

Parnas & Clements 1986

- A system’s users seldom know exactly what they want and cannot articulate all they know
- Even if we could state all requirements, there are many details that we can only discover once we are well into implementation
- Even if we knew all the details, as humans we can master only so much complexity
- Even if we could master all this complexity, external forces lead to changes in requirements, some of which may invalidate earlier decisions
Why is Requirements Engineering Important?

- Poor requirements capture leads to SW developers solving the wrong problem or attempting an infeasible problem (=$$$)
- Misunderstanding the requirements leads to a chaotic development process (=$$$)
- see Denver Airport Baggage Handling system (lecture 1)

The 4 Major Activities of Requirements Engineering

- Elicitation
- Analysis
- Specification
- Validation

Requirements Engineering Desirable Outcomes

- SW developers are solving the RIGHT problem
- SW developers are solving a FEASIBLE problem
- Developers have a good knowledge of the system DOMAIN
- Customers and users have a good understanding of the implications of decisions they have made

Requirements Engineering Undesirable Outcomes

- SW developers are solving the WRONG problem
- SW developers are trying to solve an INFEASIBLE problem
- Poor understanding of the requirements leads to a chaotic development process
- Project is more likely to run over time and over budget
What is a SW Requirement?

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

[IEEE Std 610.12-1990]
Source: http://www.computer.org/certification/csdpprep/Glossary.htm

Functional Requirement (Defn)

An area of functionality the system must support.
The functional requirements describe the interactions between the actors and the system independent of the realization of the system.

[Bruegge & Dutoit, Glossary]

Non-functional Requirement (Defn)

A user-visible constraint on the system.
Non-functional requirements describe user-visible aspects of the system that are not directly related with the functionality of the system.

[Bruegge & Dutoit, Glossary]

Quality Attributes

- A class of non-functional requirements.
- See also design goals [B&D]
- Examples:
  - usability
  - reliability
  - security
  - safety
Project Requirements

- **Business Requirements** describe in business terms what must be delivered or accomplished to provide value.
- **Product Requirements** describe the system or product which is one of several possible ways to accomplish the business requirements.
- **Process Requirements** describe the processes the developing organization must follow and the constraints that they must obey.

Examples

- A maximum development cost requirement (a **Process requirement**) may be imposed to help achieve
- a maximum sales price requirement (a **Product requirement**)
- A requirement for the product to be maintainable (a **Product requirement**) often is traced to by
- requirements to follow particular development styles (e.g., OO), style-guides, or a review/inspection process (**Process requirements**)

References

  - Chapter 1, covering “The evolving Role of Software”, “The Changing Nature of Software”, “Software Myths”.
  - Section 1.1 “Introduction: Software Engineering Failures”