Lecture 1, Part 2: What are Requirements?

Jennifer Campbell
CSC340 - Winter 2007

Exercise: Are these requirements?

1. The elevator will only change directions when stopped at a floor.
2. When a lift is within 20 cm of the sensor's position the sensor sends a high signal; otherwise a low signal.
3. The new system shall occupy no more than 10 Mbytes of RAM.
4. We would like a new email system that is easier to use than the old one. [Bra02]

Definition of Requirement

A requirement is: “A condition or capability that must be met by the system to solve a problem or achieve an objective.”

(IEEE 1984 quoted in [Bra02])

What vs. How

- Traditionally, should specify 'what' without specifying 'how'
  - But this is not always easy to distinguish
  - The 'how' at one level of abstraction forms the 'what' for the next level
- Also misses:
  - 'Why' questions:
    - Why is this system needed?
    - Why should it behave that way?
  - 'Who' questions:
    - Whose problem is it?

Software Engineering Problems

“Software Engineering problems are located in the world, and their analysis and structuring is primarily an analysis and structuring of the world, not of the machine.”

[Jac99]

Separate the problem from the solution

- A separate problem description is useful
- Need to check:
  - Solution correctly solves the stated problem
  - Problem statement corresponds to the needs of the stakeholders

[Blu92]
But systems change the world…

Some observations about RE

- **RE is not necessarily a sequential process**
  - RE activities continue throughout the development process
- **The problem statement will be imperfect**
  - RE models are approximations of the world
    - will contain inaccuracies and inconsistencies
    - will omit some information
- **Perfecting a specification may not be cost-effective**
  - Requirements analysis has a cost
  - For different projects, the cost-benefit balance will be different
- **Problem statement should never be treated as fixed**
  - Change is inevitable, and therefore must be planned for

What are Requirements?

- **Domain Properties**: things in the application domain that are true whether or not we ever build the proposed system
- **Requirements**: things in the application domain that we wish to be made true by delivering the proposed system
- **A Specification**: is a description of the behaviours that the program must have in order to meet the requirements
  - Can only be written in terms of shared phenomena!

Example: Prevent unauthorized access to a network

Verification and Validation

- Two correctness (verification) criteria:
  - The Specification, in the context of the given domain properties, satisfies the requirements (S, D entail R)
  - The Program running on a particular Computer satisfies the Specification
- Two completeness (validation) criteria:
  - We discovered all the important requirements
  - We discovered all the relevant domain properties

Example: Network

- **Requirement R**:
  - “The network shall only be accessible by authorized personnel”
- **Domain Properties D**:
  - Authorized personnel have passwords
  - Passwords are never shared with non-authorized personnel
- **Specification S**:
  - Access to the network shall only be granted after the user types an authorized password
Example: Network

- **Requirement R:**
  - "The network shall only be accessible by authorized personnel"
- **Domain Properties D:**
  - Authorized personnel have passwords
  - Passwords are never shared with non-authorized personnel
- **Specification S:**
  - Access to the network shall only be granted after the user types an authorized password
- **Verification:** S, D entail R

Why did an unauthorized person access the database?

Example: Aircraft

- **Requirement R:**
  - "Reverse thrust shall only be enabled when the aircraft is moving on the runway"
- **Domain Properties D:**
  - Wheel pulse sensors on if and only if wheels turning
  - Wheels turning if and only if moving on runway
- **Specification S:**
  - Reverse thrust enabled if and only if wheel pulse sensors on
- **Verification:** S, D entail R

Why did the plane crash?

Moving the boundaries...

- E.g. Elevator control system:
  - People waiting
  - People in the elevator
  - People wanting to go to a particular floor
  - Elevator call buttons
  - Floor request buttons
  - Button lights
  - Current floor indicators
  - Motor on/off
  - Door open/close
  - Scheduling algorithm
  - Control program
  - Safety rules

We can shift things around:
  - E.g. Add some sensors to detect when people are waiting

  This changes the nature of the problem to be solved

Summary

- Requirements Engineering is about describing problems
  - It is useful to separate the problem from the solution
  - Problems evolve continuously
- Key distinctions:
  - Application Domains vs. Machine Domains
  - Verification vs. Validation
- Next lecture:
  - Science, Engineering, and Systems;
  - Software Requirements Specification

References
