Lecture 12 – Defining System Requirements

- Good Requirements Definition
- Small Example Case
- Examples of Inadequate Requirements

**Learning outcomes:**
- Understand the importance of defining adequate requirements
- Understand good practices for requirements definition
- Practice defining requirements: discuss weaknesses of inadequate requirements

---

**Good Requirements Definition**

A **requirement** is an attribute of the system and should be described with a simple statement of what the system should do or what characteristics it should have.

**Functional** requirements: processes and information

**Non-functional** requirements: behaviour of the system
- Operational: physical/technical environment
- Performance: speed, capacity, reliability
- Security: authorized access
- Cultural and Political: factors affecting the system

---

**Good Practices for Defining Requirements**

- Different types of requirements
- Customer involvement
- Unambiguous and clear
- Prioritised list
- Required functionality
- Closure of analysis
- Observe system scope
- Manage changes
- Single requirements
- Wishful thinking

**Good Requirements**

Necessary, feasible, correct, concise, unambiguous, complete, consistent, verifiable, traceable, allocated, design independent, non-redundant, clearly written, well identified, no escape clauses.
Small Example Case

Pretend that you are going to build a new system that automates or improves the interview process for the Career Services Department of your university.

Develop a requirements definition for the new system. Include both functional and non-functional system requirements. Pretend you will release the system in three different versions. Prioritize the requirements accordingly.

Functional Requirements

Version 1
- Career Services personnel post each company’s interview schedule on the system
- Students may reserve one and only one interview slot per company
- Students may change their interview reservation until the day before the interview
- Confirmation of interview reservation is sent to student by email 24 hours prior to interview

Version 2
- If an interview schedule is full, student may register on a waiting list
- If openings appear on an interview schedule, students on waiting list are notified by email

Version 3
- Students may register their specific qualifications and when a company seeking those qualifications opens an interview schedule, the student is notified by email

Non-functional Requirements (for the 3 versions)

Performance
- System is accessible via a web browser
- System integrates with the resume-posting system and the job listings system

Operational
- System is real-time: interview reservations are immediately reflected in the interview schedules
- System is available 24 hours a day whenever university is in session
Security

- Only students registered with Career Services have access to system
- Students can choose to be identified by name on the schedule or by some other identifier

Cultural and Political

- Students can make only two changes to their interview reservation per company
- If a student fails to appear for a scheduled interview, he/she will be dropped from any other scheduled interviews they may have

Examples of Inadequate Requirements

The activity-and-precedence table should first be translated into a notation expression format before it is directly made into a PERT diagram. The notation expression should be capable of being transformed into a PERT diagram or back into an activity-and-precedence table.

The PERT diagrams produced should not overlap at all unless they are not planar. The generation process of the PERT diagram should implement this: diagrams should be “made not to contain overlaps” not “made then overlaps removed”. An algorithm should be used for the diagram generation process.

Avoid program confusion.

A ‘help’ section should be available, to assist users when usage difficulties are faced using the program. Additional assistance should be displayed where necessary.

The program should have a clean object oriented design allowing good maintainability.

Functional Requirement: store space and building information as a "model".

Handle paradoxes (both solvable and unsolvable) gracefully.

System should be platform and browser independent client.

Designed to replace the old mailing system of game play.

Additional Reading


Chapter 4 of (Dennis et al., 2006)

Chapter 6 of (Hoffer et al., 2005)