REQUIREMENTS ENGINEERING

JIMMA UNIVERSITY
JIMMA INSTITUTE OF TECHNOLOGY
FACULTY OF COMPUTING AND INFORMATICS

CHAPTER TWO

PURPOSE AND NATURE OF REQUIREMENTS

Topics we will cover

- Basics of Requirements
- Functional Requirements
- Requirement’s imprecision
- Requirements completeness and consistency
- Non-functional requirements
- Domain requirements

Basics of Requirements

"The beginning is the most important part of the work"

- Requirements capture what a proposed system should do, and the constraints they should follow.
- Poor requirements are the source of all evil.
- **Poor requirements:**
  - **The Engineering Argument:** Engineering is about developing solutions to problems. A good solution can only be developed if engineers understand the problem.
  - **The Economic Argument:** Errors cost more to correct the longer they go undetected.
  - **The Empirical Argument:** Failure to understand and manage requirements is the biggest single cause of cost and schedule over-runs

Basics of Requirements

1. View university wide events.
2. Easy to use with no training or user manuals.
3. Adoption rate>=75% within one year
4. Web based, Mobile app(Android, iOS)
5. View event conflicts, if any
6. Auto-populate course time tables
7. Secure access through centralized authentication

Jimma University Event Management App

Types of Requirements

- Requirements fall into three categories:

1. **User requirements:** Statements in natural language plus diagrams of the services the system provides and its operational constraints. Written for **customers.**
2. **System requirements:** A structured document setting out detailed descriptions of the system services. Written as a contract between client and contractor.
3. **Software specification requirements:** A detailed software description which can serve as a basis for a design or implementation. Written for developers

Types of Requirements

- **User**, **system** and **software specification** requirements can be:
  - Functional requirements
  - Non-functional requirements
  - Domain requirements

Functional Requirements

- **Functional Requirements:** Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
- Describe functionality or system services.
- **Example:** Some of the functional requirements for the events management App.
  - The application must send an even notification whenever a new even is available.
  - The application must allow users to send feedback about the event.
  - The application must allow users to share the event to their friends.
  - etc.

Requirement's Imprecision

"A major reason to project failure is the failure to spend the time at the beginning of the project to clearly define the product requirements before beginning product development i.e. gathering imprecise requirements"

- **Requirements imprecision:** problems arise when functional requirements are not precisely stated.
- **Ambiguous requirements** may be interpreted in different ways by developers and users.
- User intention ≠ Developer interpretation
- **Example:** Consider the term "**search events**" requirement in the event management application.
    - **User intention**: Search for coming events in Jimma University.
    - **Developer interpretation**: Search for events in Jimma University.

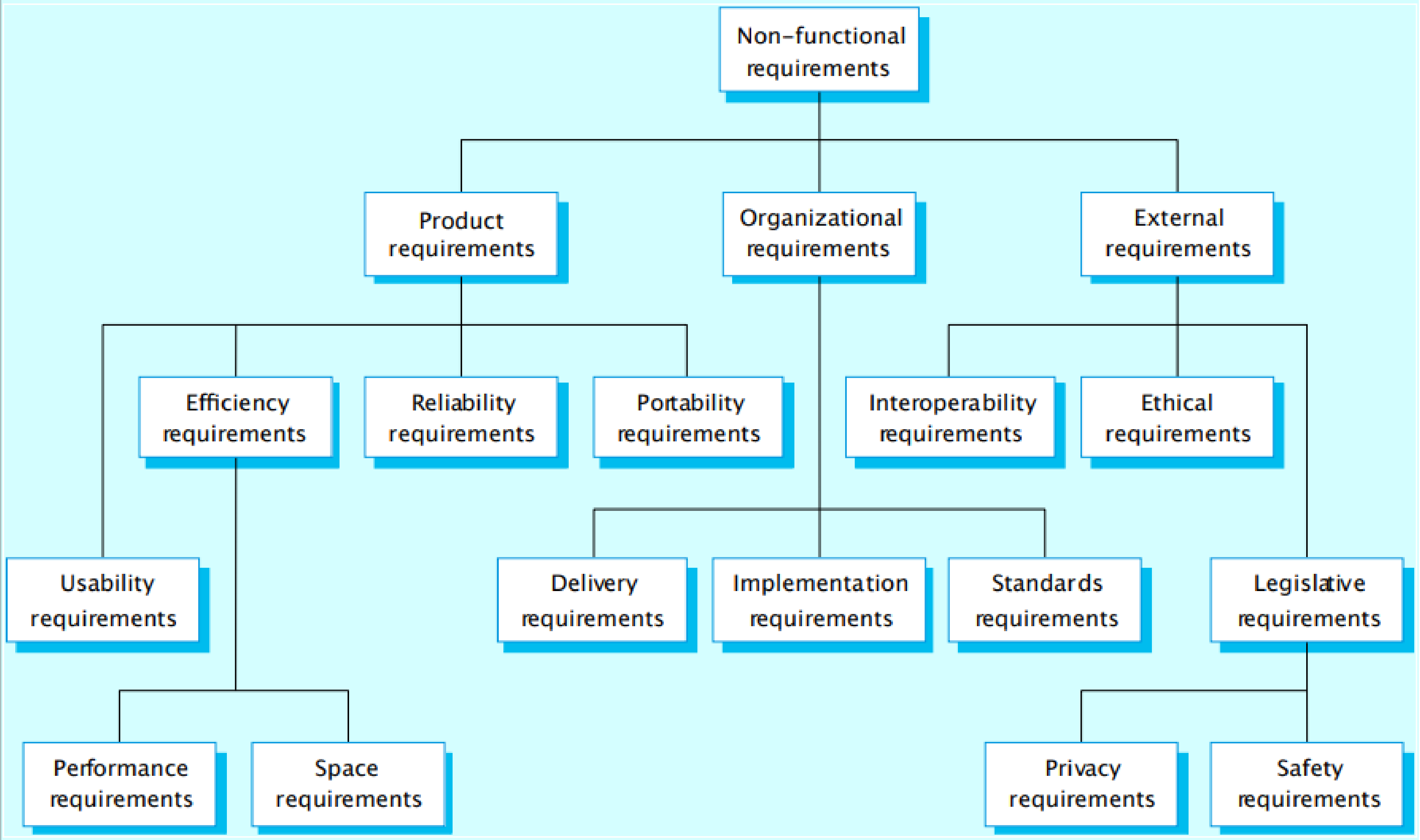
Requirements Completeness and Consistency

- In principle, requirements should be both **complete** and **consistent**.
  - **Complete:** They should include descriptions of all facilities required.
  - **Consistent:** There should be no conflicts or contradictions in the descriptions of the system facilities.
- In practice, it is impossible to produce a complete and consistent requirements document.

Non-Functional Requirements

- **Non-functional requirements** are the constraints or the requirements imposed on the system.
- **Non-functional requirements** deal with issues like scalability, maintainability, performance, usability, and many more.
- **Non-functional requirements** classified in to three major categories.
  1. **Product requirements**: Requirements which specify that the delivered product must behave in a particular way, e.g. execution speed, reliability etc.
  2. **Organizational requirements:** Requirements which are a consequence of organizational policies and procedures, e.g. process standards used, implementation requirements etc.
  3. **External requirements:** Requirements which arise from factors which are external to the system and its development process, e.g. interoperability requirements, legislative requirements etc.

Non-Functional Requirements



Product Requirements

- **Usability:** Understand the application/system in context of user, technology, task/activity and environment.
  - **User:** Setting up user profiles . Who are the users of the application?
  - **Tasks**: Tasks are the activities undertaken to achieve a goal using the application. Descriptions of the activities and steps involved in performing the task should be related to the goals that are to be achieved.
  - **Technology:** Technology refers to the capabilities and constraints of technology platform needed for running the application. This factor determines user interface related capabilities and constraints of the hardware and software platform for the application.
  - **Environment:** Environment refers to the Internet capability and availability needed for running the application.

Product Requirements

- **Efficiency:** Efficiency refers to the capability of the application to provide appropriate performance, relative to the number of resources used, under stated conditions.
  - **Performance requirements:** the user should not spend a lot of time in doing specific task.
  - **Space requirements**: the system should not take a lot of space while doing a specific task.
- **Reliability:** Reliability refers to error handing of the application and the extent to which the action history is dependable, trusted and reach user mind stability.
- **Portability:** Portability is the ease with which a software system can be transferred from its current hardware or software environment to another environment.

Organization Requirements

- **Delivery requirements:** Delivery Requirements are the deadlines given by the customer to the software team for the completion of the given system.
- **Implementation requirements:** implementation requirements are the requirements dealing with software installation and software demonstrations with the actual users of the software.
- **Standard requirements:** Describe the process standards to be used during software development. For example, the software should be developed using standards specified by the ISO and IEEE standards.

External Requirements

- **Interoperability requirements:** Interoperability requirements define how business processes are to be shared.
- **Ethical requirements:** Specify the rules and regulations of the software so that they are acceptable to users.
- **Legislative requirements:** Ensure that the software operates within the legal jurisdiction. For example, pirated software should not be sold.

Domain Requirements

- **Domain requirements:** Requirements which are derived from the application domain of the system instead of the needs of the users.
- Domain requirements are expectations related to a particular type of software, purpose or industry vertical.
- **Domain requirements problems:**
  - **Understandability:** Requirements are expressed in the language of the application domain. This is often not understood by software engineers developing the system.
  - **Implicitness:** Domain specialists understand the area so well that they do not think of making the domain requirements explicit.

Key Points

- Basics of Requirements
- Functional Requirements
- Requirement’s imprecision
- Requirements completeness and consistency
- Non-functional requirements
- Domain requirements

Any Question?