THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Requirements Engineering

Slide Set - 4

Organized & Presented By:
Software Engineering Team CSED
TIET, Patiala

Software Requirements

Descriptions and specifications of a system

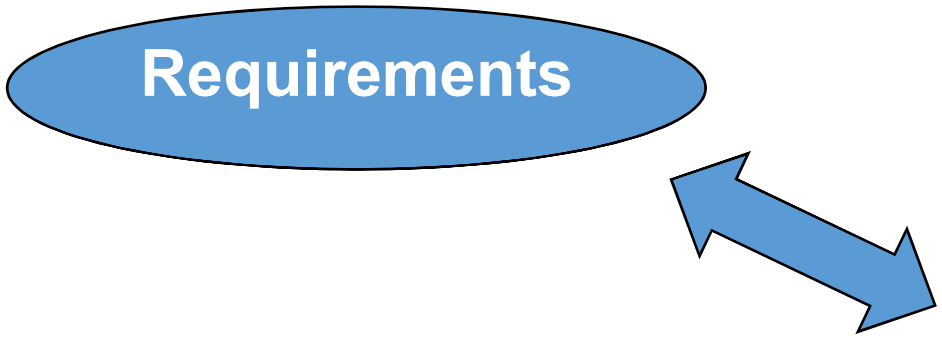
Objectives:

- To introduce the concepts of **user and system requirements**
- To describe **functional / non-functional requirements**
- To explain **two techniques** for describing system requirements
- To explain **how software requirements may be organised** in a requirements document

Requirements engineering

Requirements engineering is the process of establishing

- the services that the customer requires from a system
- the constraints under which it operates and is developed



The descriptions of the system services and constraints

that are generated during the requirements engineering process

Functional and non-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.
- **Non-functional requirements**
  - constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.
- **Domain requirements**
  - Requirements that come from the application domain of the system and that reflect characteristics of that domain

Functional Requirements

Describe functionality or system services

- **Depend on the type of software**, expected users and the type of system where the software is used
- **Functional user requirements may be high-level statements of what the system should do**

Examples of functional requirements

- The user shall be able to search either all of the initial set of databases or select a subset from it.
- The system shall provide appropriate viewers for the user to read documents in the document store.
- Every order shall be allocated a unique identifier (ORDER_ID) which the user shall be able to copy to the account’s permanent storage area.

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 very difficult or impossible to produce a complete and consistent requirements document

Non-functional requirements

Define system properties and constraints e.g. reliability, response time and storage requirements.

Constraints are I/O device capability, system representations, etc.

Constraints are limitations or restrictions that the software must operate within.
Examples include:
I/O Device Capability: This means the software can only use certain types of input and output devices (like specific printers or scanners).
System Representations: It might refer to how data or information is organized and presented within the software.

- **Process requirements** may also be specified mandating a particular CASE system, programming language or development method

Sometimes, the development process itself has requirements. This could include using a particular software development tool, programming language, or method.

- **Non-functional requirements** may be more critical than functional requirements. If these are not met, the system is useless

For example, if a banking app doesn't have good security (a non-functional requirement), it could put users' sensitive information at risk, making the app useless even if it can technically show account balances (a functional requirement).

Non Functional Requirements Measures

Property	**Measure**
Speed	Processed transactions/second User/Event response time Screen refresh time
Size	K Bytes Number of RAM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

User requirements

- **Should describe functional and non-functional requirements** so that they are understandable by system users who don’t have detailed technical domain knowledge

- **User requirements are defined using natural language**, **tables** and **diagrams**

Problems with natural language

- **Lack of clarity**
  - Precision is difficult without making the document difficult to read
- **Requirements confusion**
  - Functional and non-functional requirements tend to be mixed-up
- **Requirements amalgamation**
  - Several different requirements may be expressed together

Guidelines for writing requirements

- Invent a standard format and use it for all requirements
- Use language in a consistent way. Use
  shall for mandatory requirements,
  should for desirable requirements
- Use text highlighting to identify key parts of the requirement

Avoid the use of computer jargon !!!

Requirements and design

- **In principle**, requirements should state **what the system should** do and

  the design should describe **how it does this**

- **In practice**, requirements and design are inseparable
  - A system architecture may be designed to structure the requirements
  - The system may inter-operate with other systems that generate design requirements
  - The use of a specific design may be a domain requirement

Problems with NL specification

- **Ambiguity**
  - The readers and writers of the requirement must interpret the same words in the same way. NL is naturally ambiguous so this is very difficult
- **Over-flexibility**
  - The same thing may be said in a number of different ways in the specification
- **Lack of modularisation**
  - NL structures are inadequate to structure system requirements

Natural language is not well-suited for breaking down complex systems into smaller, more manageable parts (modularization).
Software systems are often complex and consist of many interconnected components. Natural language may not provide a structured way to represent these relationships and dependencies effectively.

Video Link – Functional vs Non-Functional Requirements

https://www.youtube.com/watch?v=NE1_cAWzQLM