

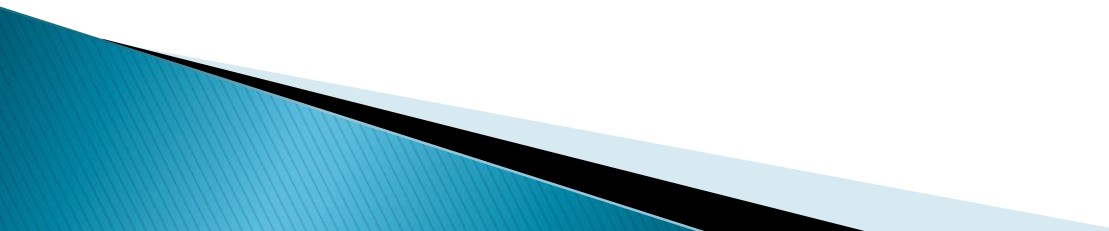
Requirement Engineering



Software Requirements

Descriptions and specifications of a system

Objectives:

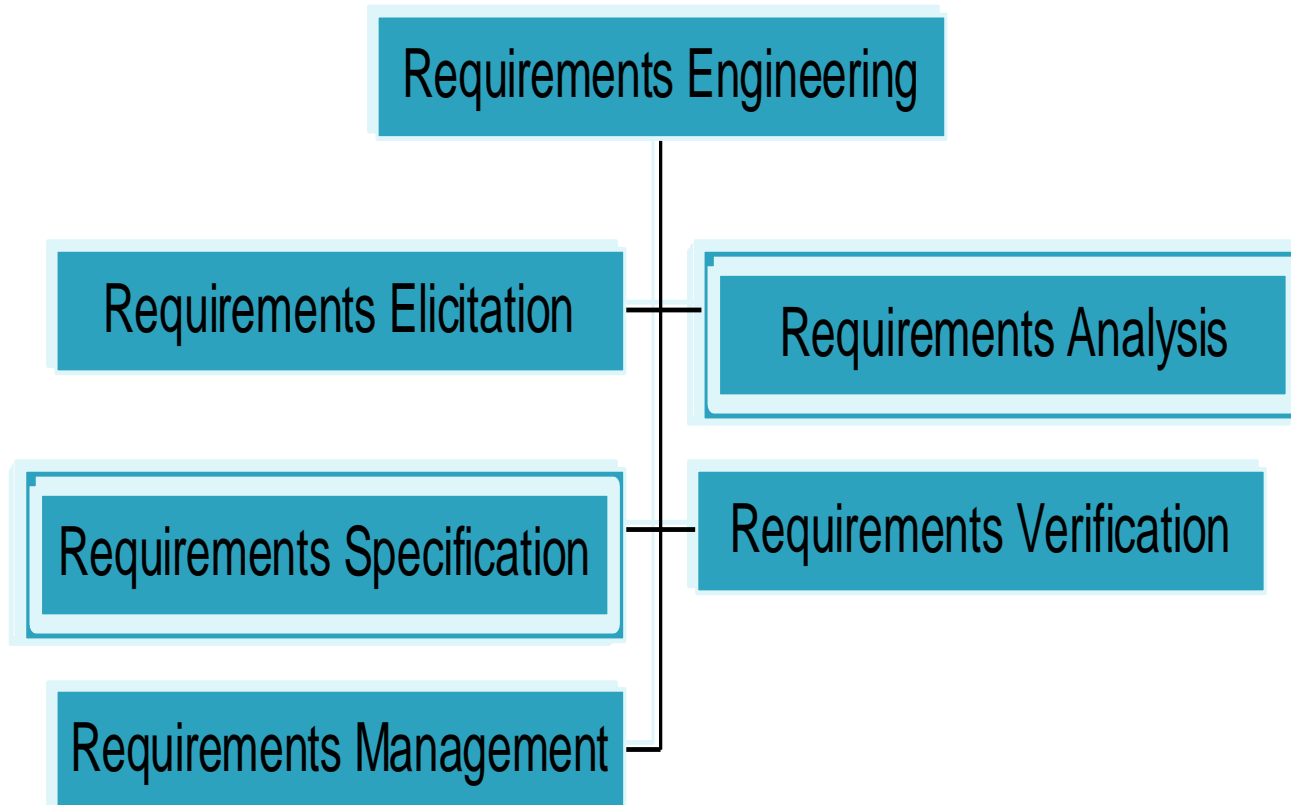
- ▶ To introduce the concepts of **user and system requirements**
 - ▶ To describe **functional / non-functional requirements**
 - ▶ To explain **two techniques** for describing system requirements
 - ▶ To explain **how software requirements may be organised** in a requirements document
- 

Topics covered

Functional and non-functional requirements

- User requirements
- System requirements
- The software requirements document

Requirements Engineering

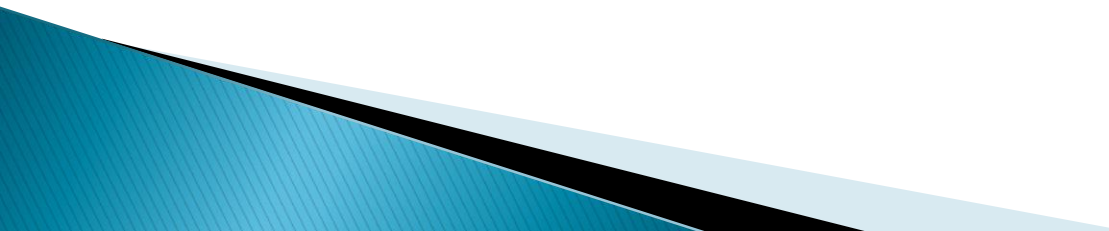


Requirements Analysis & Specification Definitions

Requirements Analysis

- The process of studying and analyzing the customer and the user needs to arrive at a definition of software requirements.

Requirements Specification

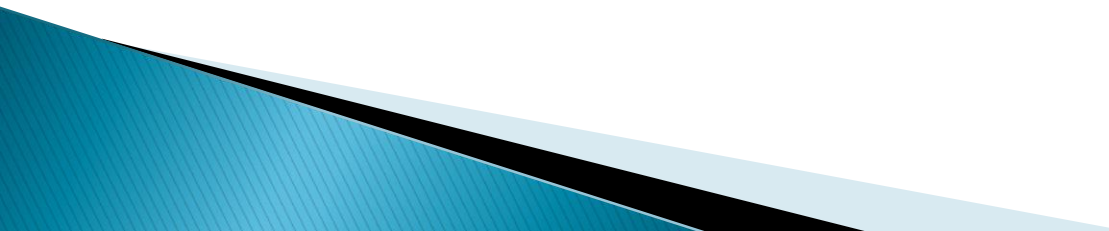
- A document that clearly and precisely describes, each of the essential requirements (functions, performance, design constraint, and quality attributes) of the software and the external interfaces. Each requirement being defined in such a way that its achievement is capable of being objectively verified by a prescribed method; for example inspection, demonstration, analysis, or test.
- 

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.

What is Requirement?

- ▶ **Requirement** is a singular documented physical and functional need that a particular design, product or process must be able to perform.
 - ▶ It is a statement that identifies a necessary attribute, capability, characteristic, or quality of a system for it to have value and utility to a customer, organization, internal user, or other stakeholder
- 

Types of requirement

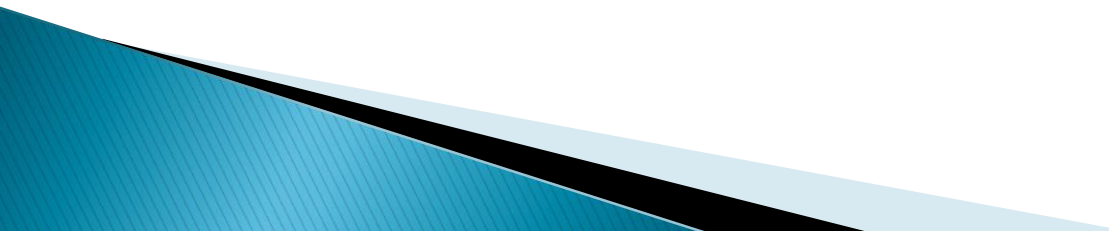
User requirements

- Statements in natural language plus diagrams of the services the system provides and its operational constraints. Written for customers.

System requirements

- A structured document setting out detailed descriptions of the system services. Written as a contract between client and contractor.

Software specification

- A detailed software description which can serve as a basis for a design or implementation. Written for developers.
- 

Types of requirement

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

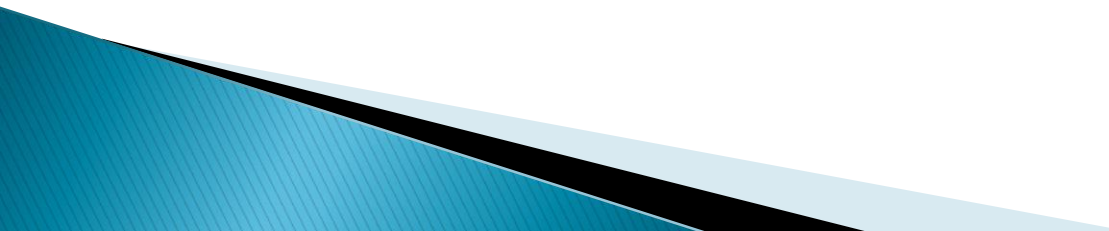
- Requirements that come from the application domain of the system and that reflect characteristics of that domain

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** BUT functional system requirements should describe the system services in detail

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 imprecision

- ▶ Problems arise when requirements are not precisely stated
- ▶ Ambiguous requirements may be interpreted in different ways by developers and users
- ▶ Consider the term ‘appropriate viewers’
 - **User intention** – special purpose viewer for each different document type
 - **Developer interpretation** – Provide a text viewer that shows the contents of the document

Requirements completeness and consistency

- ▶ In principle requirements should be both complete and consistent

Complete

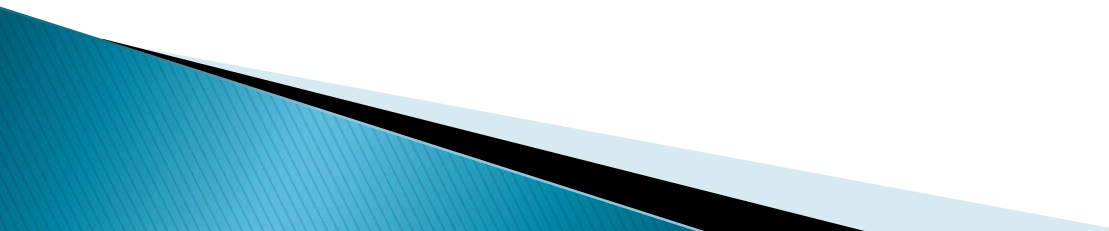
- They should include descriptions of all facilities required

Consistent

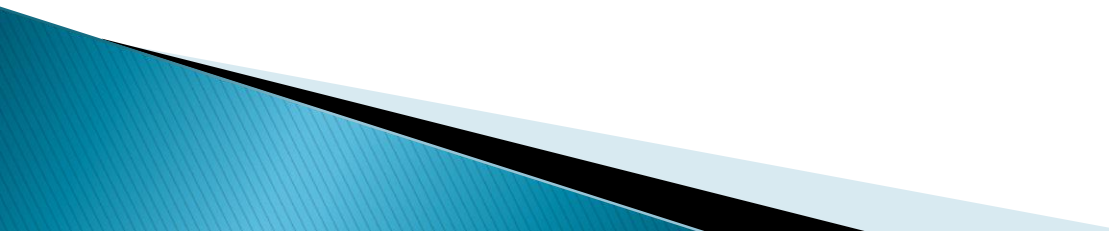
- There should be no conflicts or contradictions in the descriptions of the system facilities

- ▶ **In practice**, it is very difficult or impossible to produce a complete and consistent requirements document

Non-functional requirements

- ▶ Non-functional requirements define system properties and constraints.
 - ▶ Non-functional requirements describe how the system works, while functional requirements describe what the system should do.
 - ▶ Non-functional requirement is that it essentially specifies **how the system should behave.**
- 

Non-functional requirements

- ▶ Some typical non-functional requirements are:
 - ▶ Performance – for example Response Time, Throughput, Utilization.
 - ▶ Scalability
 - ▶ Capacity
 - ▶ Availability
 - ▶ Reliability
 - ▶ Maintainability
 - ▶ Security
 - ▶ Regulatory
 - ▶ Manageability
 - ▶ Data Integrity
 - ▶ Usability
- 

Non-functional classifications

▶ Product requirements

- Requirements which specify that the **delivered product must behave in a particular way** e.g. execution speed, reliability, etc.

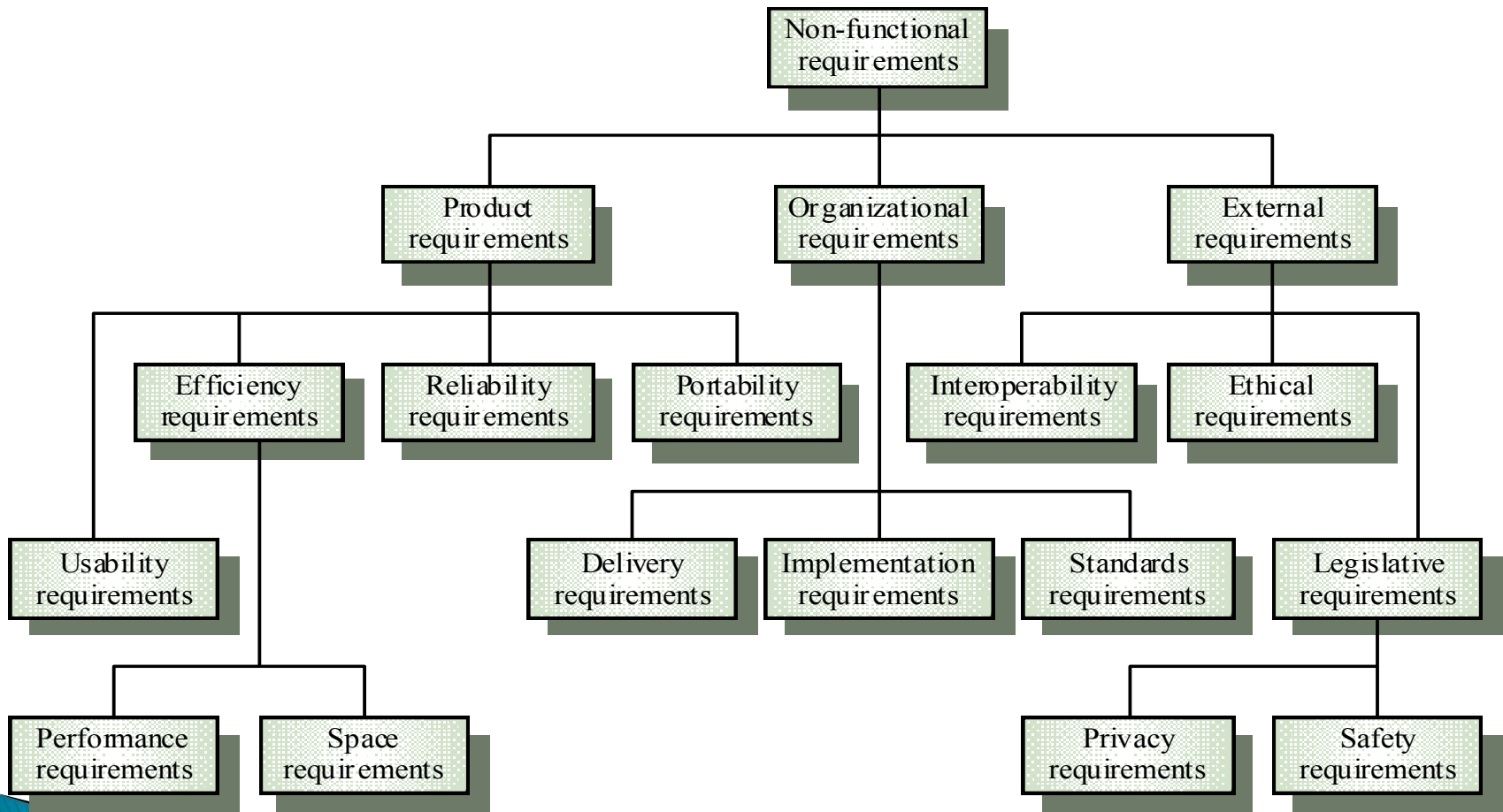
▶ Organisational requirements

- Requirements which are a **consequence of organisational policies and procedures** e.g. process standards used, implementation requirements, etc.

▶ 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 classifications



Non-functional requirements examples

Product Requirements

- ▶ The System service X shall have an availability of 999/1000 or 99%. This is a reliability requirement which means that out of every 1000 requests for this service, 999 must be satisfied.
- ▶ System Y shall process a minimum of 8 transactions per second. This is a performance requirement.

Organisational requirement

- ▶ The development process to be used must be explicitly defined and must be conformant with ISO 9000 standards
- ▶ The system must be developed using the XYZ suite of CASE tools

Non-functional requirements examples

External requirement

- The system shall not disclose any personal information about customers apart from their name and reference number to the operators of the system

Non-functional requirements examples

- ▶ **Non-functional requirements** may be very difficult to state precisely and imprecise requirements may be difficult to verify.
- ▶ **Goal**
 - A general intention of the user such as ease of use
- ▶ **Verifiable non-functional requirement**
 - A statement using some measure that can be objectively tested
- ▶ **Goals are helpful to developers** as they convey the intentions of the system users

Examples

▶ A system goal

- The system should be easy to use by experienced controllers and should be organised in such a way that user errors are minimised.

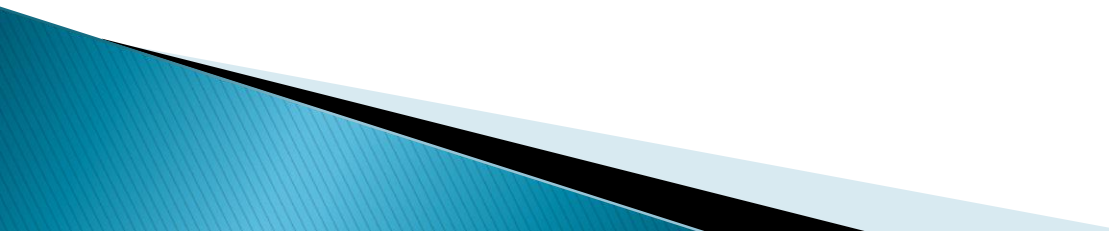
▶ A verifiable non-functional requirement

- Experienced controllers shall be able to use all the system functions after a total of two hours training. After this training, the average number of errors made by experienced users shall not exceed two per day.

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

Domain requirements

- ▶ Derived from the application domain and describe system characteristics and features that reflect the domain
 - ▶ May be new functional requirements, constraints on existing requirements or define specific computations
 - ▶ If domain requirements are not satisfied, the system may be unworkable
- 

Domain requirements problems

▶ Understand ability

- 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