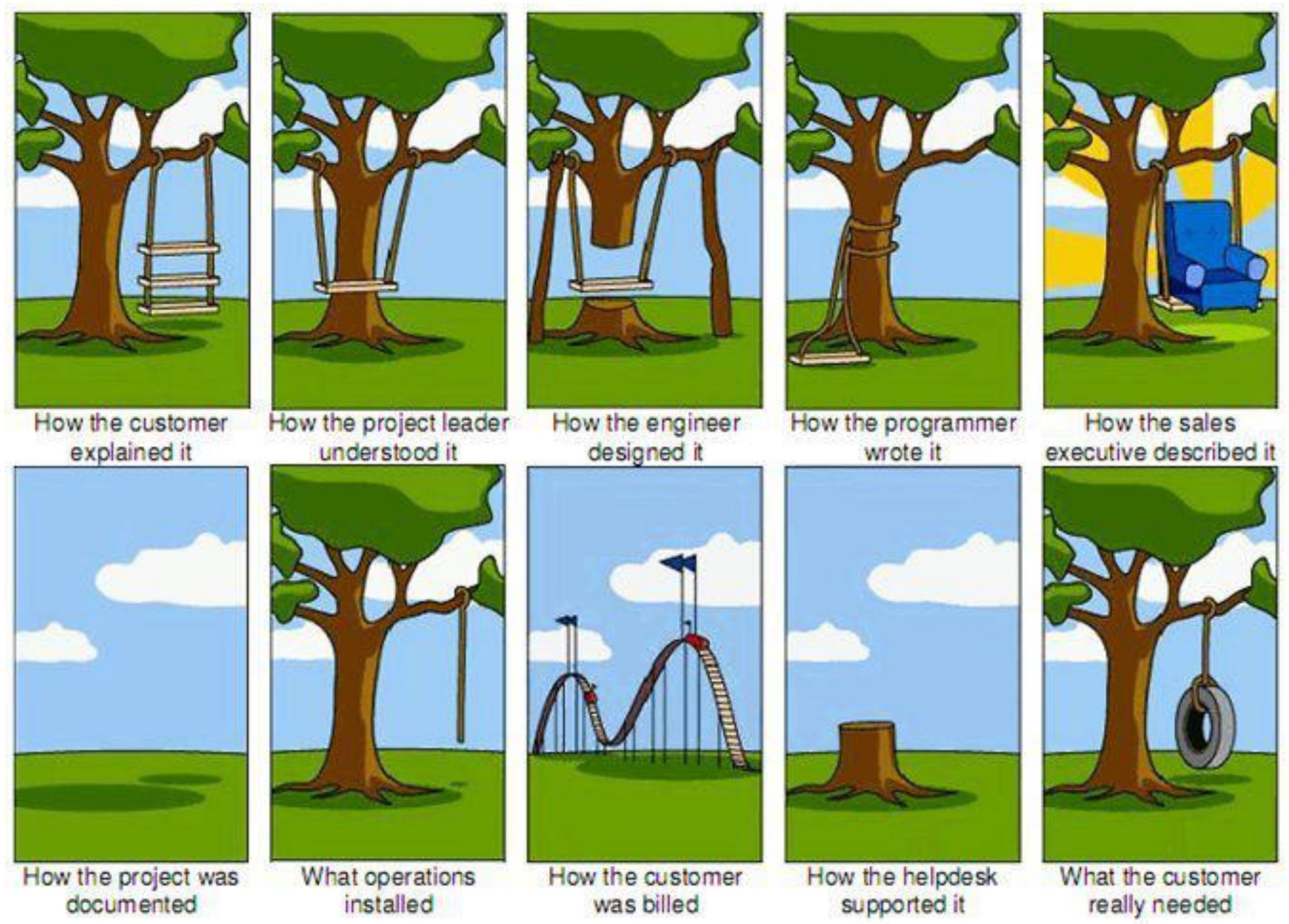
Software Requirements

With materials adapted from Software Engineering, 8th Ed. by Ian Sommerville

Objectives

- To describe functional and non-functional requirements
- To introduce the concepts of user and system requirements
- To explain how software requirements may be organized in a requirements document

Requirements Engineering


How the customer explained it
How the project leader understood it
How the engineer designed it
How the programmer wrote it
How the sales executive described it
How the project was documented
What operations installed
How the customer was billed
How the helpdesk supported it
What the customer really needed

Topics covered

- Functional and non-functional requirements
- User requirements
- System requirements
- The software requirements document

Requirements engineering

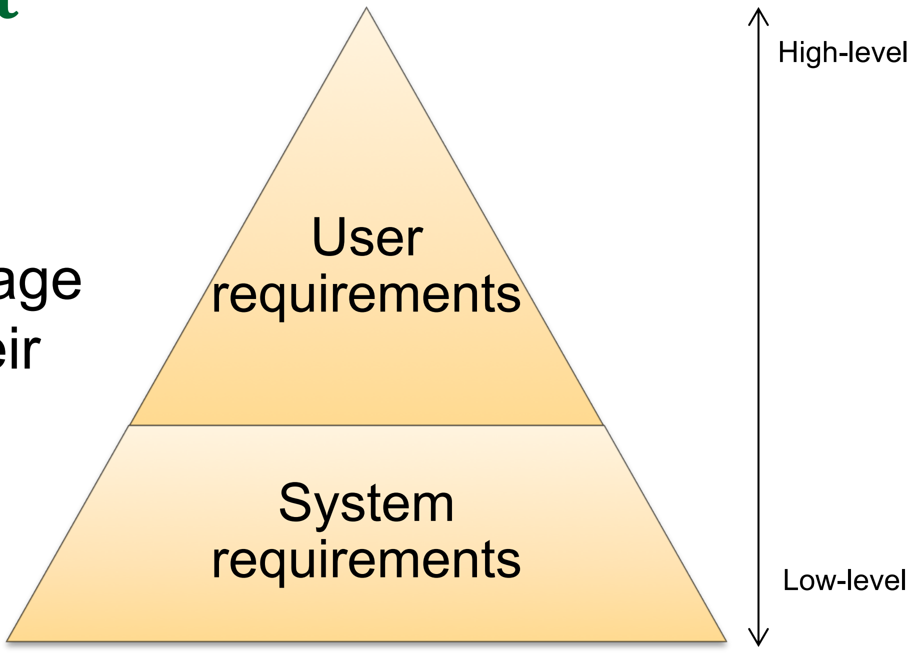
- Process of establishing the **services** that a system provide and the **constraints** for the system
  - Services and constraints are mainly from the customer
- Requirements
  - descriptions of the system services and constraints
  - generated during the requirements engineering process

What is a requirement?

- It may range from a high-level abstract statement of a service or a constraint to a detailed functional specification
  - Requirements specify **what** services a system provides and **how** the services are executed
- Requirements may serve a dual function
  - the basis for a bid or a contract – normally at high-level
  - the basis for the development team – detailed level

Types of requirement

- User requirements
  - Statements in natural language of system's services and their operational constraints
  - User point of view



- System requirements
  - A structured document detailing system's functions, services and operational constraints
  - Define what should be implemented
  - Technical point of view (designer, implementer, tester)

Definitions and specifications

User requirement definition

1. The software must provide a means of representing and accessing external files created by other tools

System requirements specification

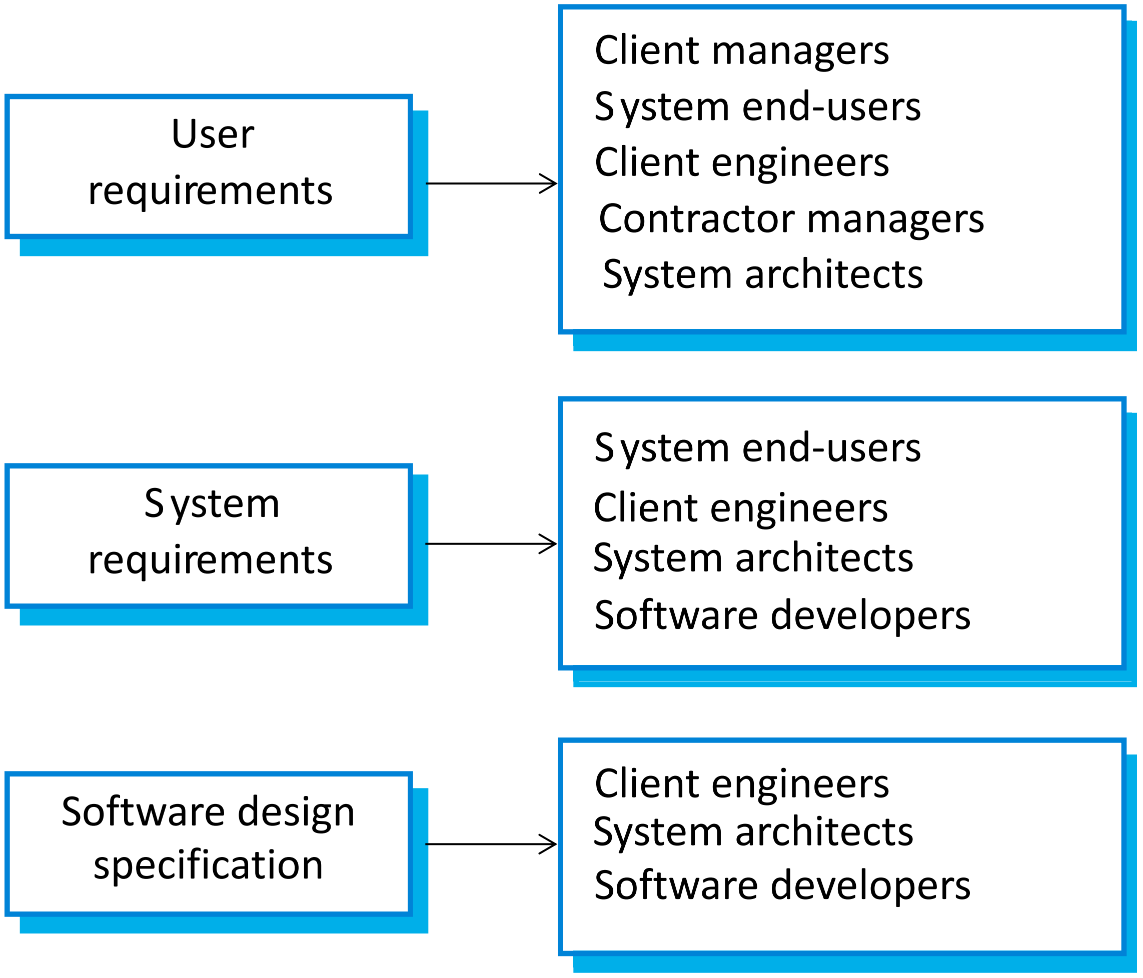
1.1 The user should be provided with facilities to define the type of external files

1.2 Each external file type may have an associated tool which may be applied to the file

1.3 Each external file type may be represented as a specific icon on the user's display

1.4 Facilities should be provided for the icon representing an external file type to be defined by the user

Requirements readers



Functional and non-functional requirements

- **Functional requirements**
  - State how the system should react to particular inputs and how the system should behave in particular situations
- **Non-functional requirements**
  - State **Constraints** on services or functions offered by the system
    - timing constraints, constraints on the development process, standards, etc.
- **Domain requirements**
  - come from the application domain of the system
  - reflect characteristics of that domain

Functional requirements

- Describe functionality or system services
- Functional user requirements
  - high-level statements of what the system should do
- Functional system requirements
  - describe the system services in detail

The LIBSYS system

- A library system that provides a single interface to a number of databases of articles in different libraries
- Users can search for, download and print these articles for personal study

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

Requirements imprecision

- Problems arise when requirements are not precisely stated
- Ambiguous requirements may be interpreted in different ways by developers and users

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

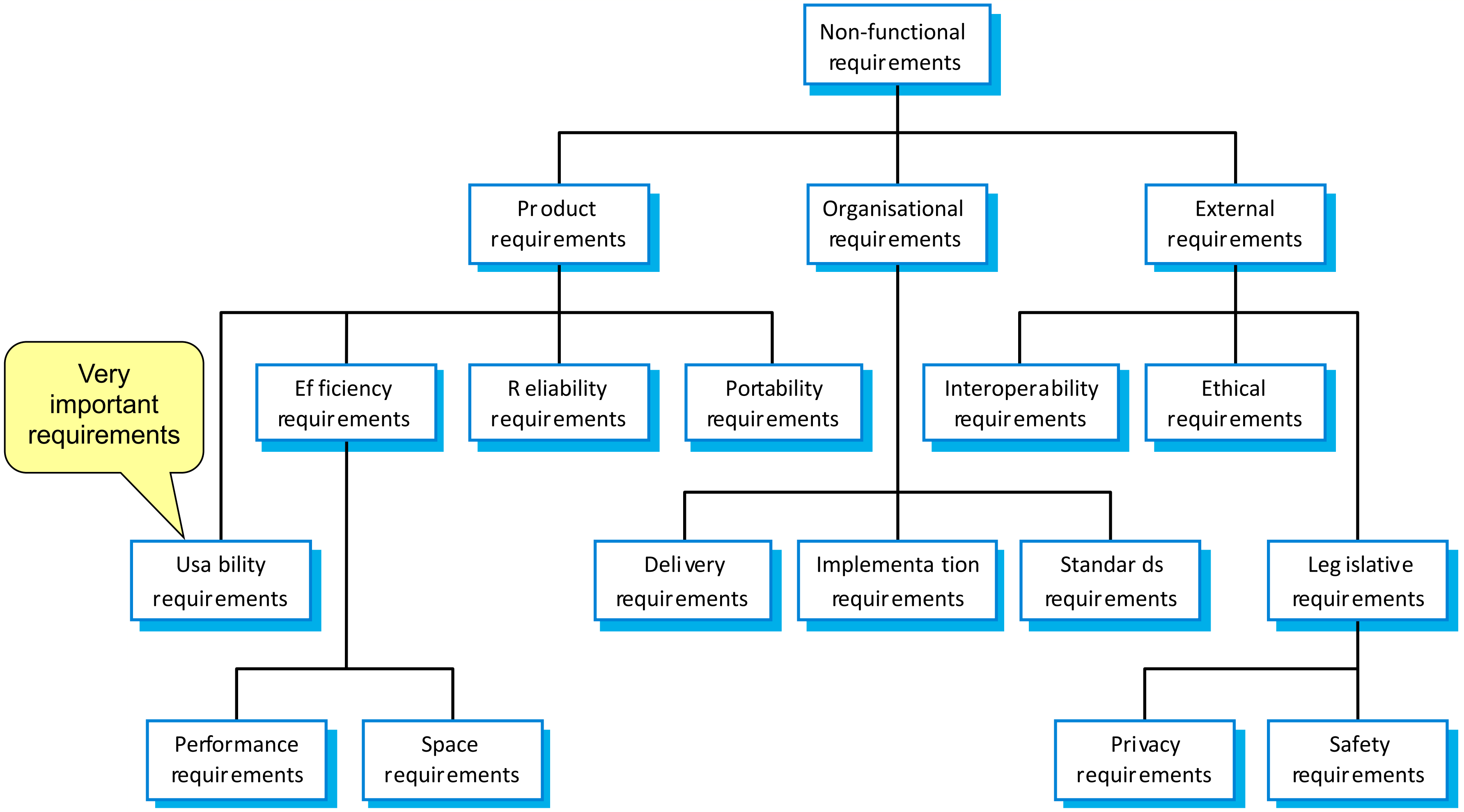
Functional and non-functional requirements

- **Functional requirements**
  - State how the system should react to particular inputs and how the system should behave in particular situations
- **Non-functional requirements**
  - State **Constraints** on the services or functions offered by the system
    - timing constraints, constraints on the development process, standards, etc.
- **Domain requirements**
  - come from the application domain of the system
  - reflect characteristics of that domain

Non-functional requirements

- Define system properties and constraints
  - E.g., reliability, performance, storage requirements, usability
- Process requirements may also be specified to mandate
  - programming language
  - or development method
- Non-functional requirements may be more critical than functional requirements

Non-functional requirement types



Goals and requirements

- Non-functional requirements may be very difficult to state precisely
  - often stated as goals
- 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 organized 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	M Bytes Number of ROM 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

Functional and non-functional requirements

- **Functional requirements**
  - State how the system should react to particular inputs and how the system should behave in particular situations
- **Non-functional requirements**
  - State **Constraints** on the services or functions offered by the system
    - timing constraints, constraints on the development process, standards, etc.
- **Domain requirements**
  - come from the application domain of the system
  - reflect characteristics of that domain

Domain requirements

- Derived from the application domain and describe system characteristics and features that reflect the domain
- Specify new functional requirements, constraints on existing requirements or define specific computations

Library system domain requirements

- There shall be a standard user interface to all databases which shall be based on the Z39.50 standard

- Because of copyright restrictions, some documents must be deleted immediately on arrival.

Domain requirements problems

- Understandability
  - Requirements are expressed in the language of the application domain
  - This is often not easily 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

Topics covered

- Functional and non-functional requirements
- User requirements
- System requirements
- The software requirements document

User requirements

- Should describe functional and non-functional requirements in a way understandable by system users
- User requirements are often 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

- Create 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

Topics covered

- Functional and non-functional requirements
- User requirements
- System requirements
- The software requirements document

System requirements

- More detailed specifications of system functions, services and constraints than user requirements
- They are intended to be a basis for designing the system
- System requirements may be defined or illustrated using system models

Requirements and design

- In principle
  - requirements should state what the system should do
  - design should describe how the system does
- 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 Natural Language (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

Alternatives to NL specification

Notation	Description
Structured natural language	This approach depends on defining standard forms or templates to express the requirements specification.
Design description languages	This approach uses a language like a programming language but with more abstract features to specify the requirements by defining an operational model of the system. This approach is not now widely used although it can be useful for interface specifications.
Graphical notations	A graphical language, supplemented by text annotations is used to define the functional requirements for the system. An early example of such a graphical language was SADT. Now, use-case descriptions and sequence diagrams are commonly used .
Mathematical specifications	These are notations based on mathematical concepts such as finite-state machines or sets. These unambiguous specifications reduce the arguments between customer and contractor about system functionality. However, most customers don't understand formal specifications and are reluctant to accept it as a system contract.

Topics covered

- Functional and non-functional requirements
- User requirements
- System requirements
- ➡ The software requirements document

Software requirements document

- The requirements document is the official statement of what is required of the system developers

- Should include both a definition of user requirements and a specification of the system requirements

- It should set of WHAT the system should do rather than HOW it should do it

IEEE requirements standard

- Defines a generic structure for a requirements document that must be instantiated for each specific system
  - Introduction
  - General description
  - Specific requirements
  - Appendices
  - Index

Requirements document structure

- Preface
- Introduction
- Glossary
- User requirements definition
- System architecture
- System requirements specification
- System models
- System evolution
- Appendices
- Index

Requirements Documents in Practice

- RUP: Two main types of requirements documents
  - Vision document
  - Use-case specifications
- Agile methods: Scrum and XP
  - Vision or objectives document
  - User stories

Key points

- Requirements set out what the system should do and define constraints on its operation and implementation
- Functional requirements set out services the system should provide
- Non-functional requirements constrain the system being developed or the development process.
- User requirements are high-level statements of what the system should do
- User requirements should be written using natural language, tables and diagrams