Requirement Engineering

- The process of establishing the services that the customer requires from a system and the constraints under which it operates and is developed.
- The requirements themselves are the descriptions of the system services and constraints that are generated during the requirements engineering process.

What is a Requirement?

- It may range from a high-level abstract statement of a service or of a system constraint to a detailed mathematical functional specification.
- This is inevitable as requirements may serve a dual function
- May be the basis for a bid for a contract - therefore must be open to **interpretation;**
- May be the basis for the contract itself - therefore must be defined in **detail;**
- Both these statements may be called requirements.

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's functions, services and operational constraints.

▪ Defines what should be implemented so may be part of a contract between client and contractor.

Functional and non-functional requirements

- **Functional requirements**

▪ Statements of services the system should provide, how the system should react to some particular inputs and how the system should behave in a particular situation.

▪ May state what the system should not do.

- **Non-functional requirements (NFR)**

▪ Nonfunctional requirements relate to software usability. Nonfunctional software requirements define how the system must operate or perform. A system can meet its functional requirements and fail to meet its nonfunctional requirements.

▪ NFRs define the software's characteristics and expected user experience (UX). They cover- performance, usability, scalability, security, portability.

- **Domain requirements**

▪ Domain requirements are expectations related to a particular type of software, purpose or industry vertical. Domain requirements can be functional or nonfunctional.

▪ The common factor for domain requirements is that they meet established standards or widely accepted feature sets for that category of software project.

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.
* Functional system requirements should describe the system services in detail.

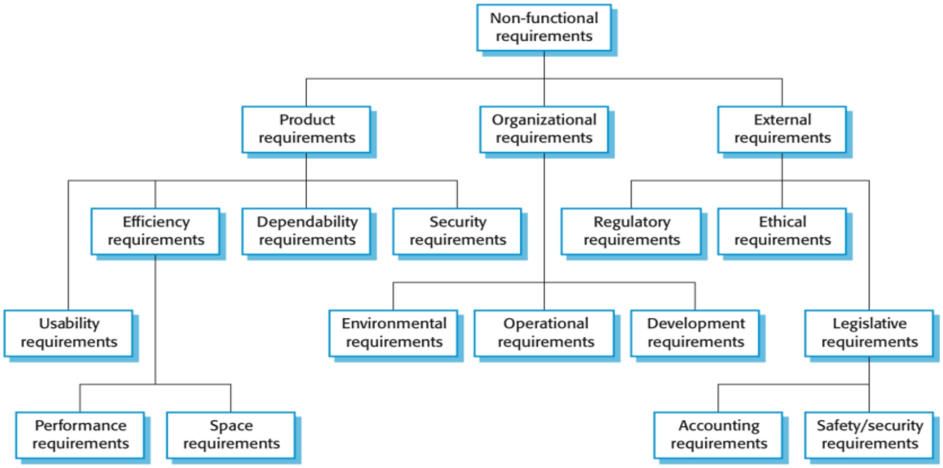
Requirements completeness and consistency

* 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

* These define system properties and constraints e.g. reliability, response time, and storage requirements. Constraints are I/O device capability, system representations, etc.
* Process requirements may also be specified mandating a particular IDE, programming language or development method.
* Non-functional requirements may be more critical than functional requirements. If these are not met, the system may be useless.

Types of Non-Functional Requirement

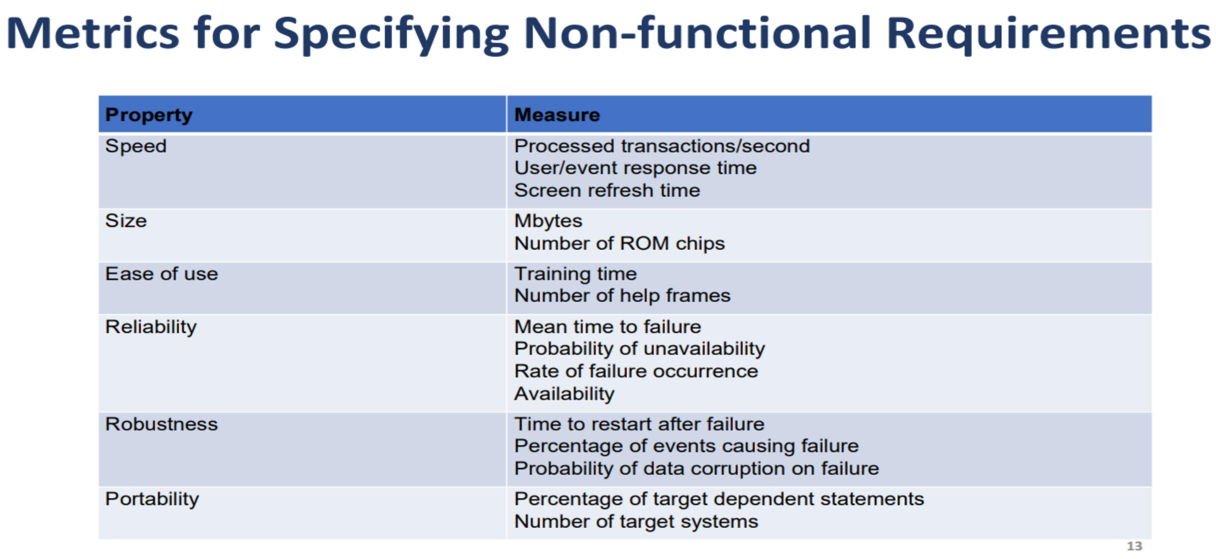


Non-Functional Requirements Implementation

* Non-functional requirements may affect the overall architecture of a system rather than the individual components.
* For example, to ensure that performance requirements are met, you may have to organize the system to minimize communications between components.
* A single non-functional requirement, such as a security requirement, may generate several related functional requirements that define system services that are required.
* It may also generate requirements that restrict existing requirements.

Goals and Requirements

* 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.



Metrics for Specifying Non-functional Requirements

Property	Measure
Speed	Processed transactions/second User/event response time Screen refresh time
Size	Mbytes 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

13

Usability Requirements

* The system should be easy to use by medical staff and should be organized in such a way that user errors are minimized. (Goal)

* Medical staff shall be able to use all the system functions after four hours of training. After this training, the average number of errors made by experienced users shall not exceed two per hour of system use. (Testable non-functional requirement).

Domain Requirements

* The system's operational domain imposes requirements on the system.

* For example, a train control system has to take into account the braking characteristics in different weather conditions.

* Domain requirements can be new functional requirements, constraints on existing requirements, or define specific computations.

* If domain requirements are not satisfied, the system may be unworkable.

Non-functional classifications

* Product requirements

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

* Organizational requirements

* Requirements which are a consequence of organizational 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.

Requirement Specification

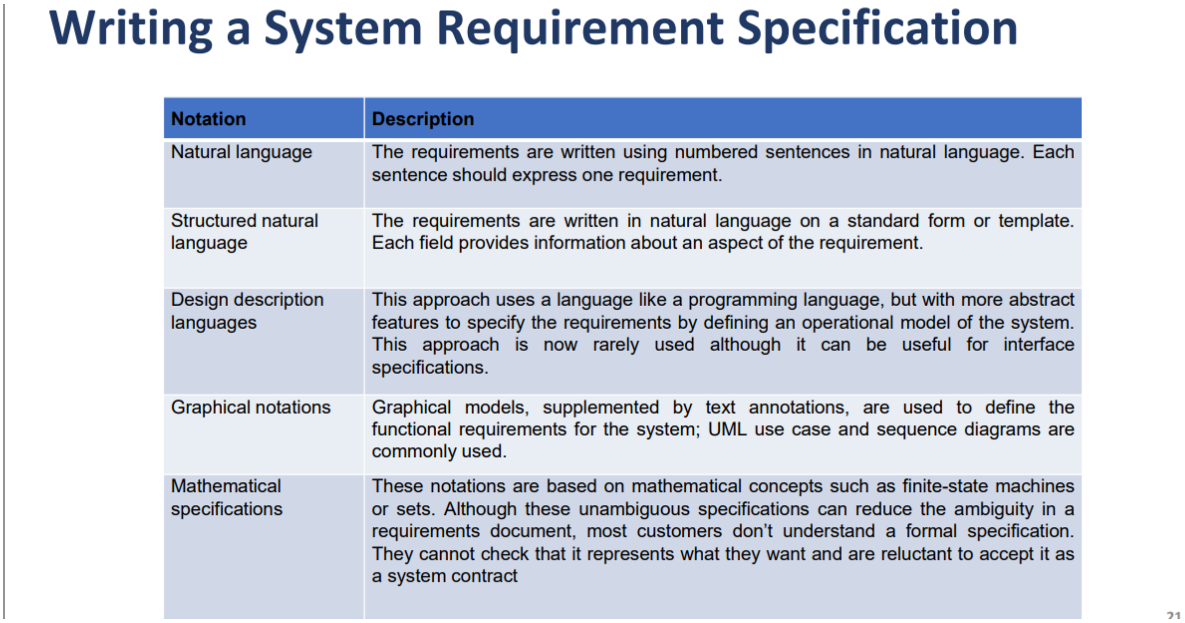
* The process of writing down the user and system requirements in a requirements document

* User requirements must be understandable by end-users and customers who do not have a technical background

* System requirements are more detailed requirements and may include more technical information

* The requirements may be part of a contract for the system development

* It is therefore important that these are as complete as possible.



Writing a System Requirement Specification

Notation	Description
Natural language	The requirements are written using numbered sentences in natural language. Each sentence should express one requirement.
Structured natural language	The requirements are written in natural language on a standard form or template. Each field provides information about an aspect of the requirement.
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 now rarely used although it can be useful for interface specifications.
Graphical notations	Graphical models, supplemented by text annotations, are used to define the functional requirements for the system; UML use case and sequence diagrams are commonly used.
Mathematical specifications	These notations are based on mathematical concepts such as finite-state machines or sets. Although these unambiguous specifications can reduce the ambiguity in a requirements document, most customers don't understand a formal specification. They cannot check that it represents what they want and are reluctant to accept it as a system contract

Requirement 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 architecture to satisfy non-functional requirements

may be a domain requirement

* This may be the consequence of a regulatory requirement

Natural Language Specification

* Requirements are written as natural language sentences supplemented by diagrams and tables.

* This language is used for writing requirements because it is expressive, intuitive and universal. This means that the requirements can be understood by users and customers.

* 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.

* Include an explanation (rationale) of why a requirement is necessary.

Guidelines for Writing Requirements

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

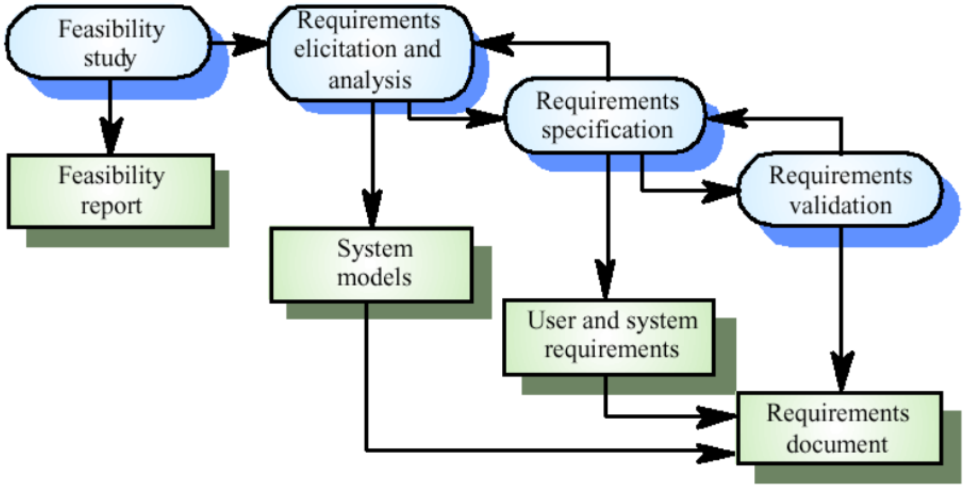
▪ Several different requirements may be expressed together.

Requirement Engineering (RE) Processes

- The processes used for RE vary widely depending on the application domain, the people involved and the organisation developing the requirements
- However, there are a number of generic activities common to all processes
- Requirements elicitation

- Requirements analysis
- Requirements validation
- Requirements management

The Requirement Engineering Process



Feasibility Study

- A feasibility study decides whether or not the proposed system is worthwhile
- A short focused study that checks
- If the system contributes to organisational objectives
- If the system can be engineered using current technology and within budget
- If the system can be integrated with other systems that are used

Feasibility Study Implementation

- Based on information assessment (what is required), information collection and report writing
- Questions for people in the organisation
- What if the system wasn't implemented?
- What are current process problems?

- How will the proposed system help?
- What will be the integration problems?
- Is new technology needed? What skills?
- What facilities must be supported by the proposed system?

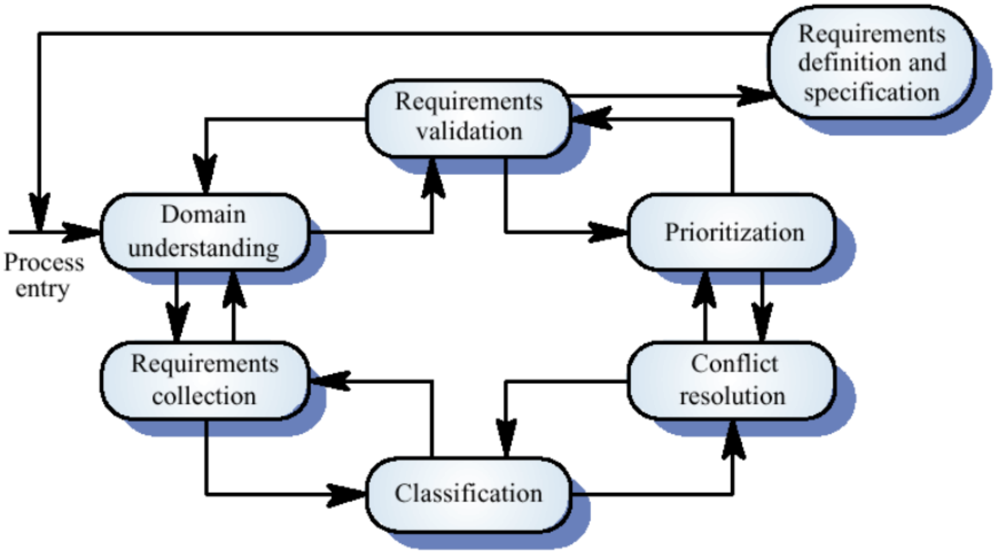
Elicitation and analysis

- Sometimes called requirements elicitation or requirements discovery
- Involves technical staff working with customers to find out about the application domain, the services that the system should provide and the system's operational constraints
- May involve end-users, managers, engineers involved in maintenance, domain experts, trade unions, etc. These are called stakeholders

Problems of Requirement Analysis

- Stakeholders don't know what they really want
- Stakeholders express requirements in their own terms
- Different stakeholders may have conflicting requirements
- Organisational and political factors may influence the system requirements
- The requirements change during the analysis process. New stakeholders may emerge and the business environment change

The Requirement Analysis Process



Use Case

* Use-cases are a scenario based technique in the UML which identify the actors in an interaction and which describe the interaction itself.

* A set of use cases should describe all possible interactions with the system.

* High-level graphical model supplemented by more detailed tabular description.

* Sequence diagrams may be used to add detail to use-cases by showing the sequence of event processing in the system.

Requirement Validation

Requirement Checking

* Validity. Does the system provide the functions which best support the customer’s needs?

* Consistency. Are there any requirements conflicts?

* Completeness. Are all functions required by the customer included?

* Realism. Can the requirements be implemented given available

budget and technology
* Verifiability. Can the requirements be checked?

Requirements reviews
▪ Systematic manual analysis of the requirements.
* Prototyping
▪ Using an executable model of the system to check requirements.
* Test-case generation
▪ Developing tests for requirements to check testability.

Requirement Validation Techniques

Requirement Review

* Regular reviews should be held while the requirements definition is being formulated.
* Both client and contractor staff should be involved in reviews.
* Reviews may be formal (with completed documents) or informal.
Good communications between developers, customers and users can resolve problems at an early stage.

Review Checks
 Verifiability
▪ Is the requirement realistically testable?
 Comprehensibility
▪ Is the requirement properly understood?
 Traceability
▪ Is the origin of the requirement clearly stated?
 Adaptability
▪ Can the requirement be changed without a large impact on other requirements?

Key Points

# Requirements for a software system set out what the system should

do and define constraints on its operation and implementation.

# Functional requirements are statements of the services that the system must provide or are descriptions of how some computations must be carried out.

# Non-functional requirements often constrain the system being developed and the development process being used.

# They often relate to the emergent properties of the system and therefore apply to the system as a whole.

# The software requirement document is an agreed statement of the system requirements. It should be organized so that both system customers and software developers can use it.

# The requirement engineering process is an iterative process including requirements elicitation, specification and validation.

# Requirement elicitation and analysis is an iterative process that can be represented as a spiral of activities – requirements discovery, requirements classification and organization, requirements negotiation and requirements documentation.

# You can use a range of techniques for requirements elicitation including interviews, scenarios, use-cases.

# Requirements validation is the process of checking the requirements for validity, consistency, completeness, realism and verifiability.

# Business, organizational and technical changes inevitably lead to changes to the requirements for a software system. Requirements management is the process of managing and controlling these changes.