

Requirements Engineering I

1

Requirements engineering


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

Requirements

↔

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 requirements

- 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




User and system requirements

User requirement definition

1. The MHC-PMS shall generate monthly management reports showing the cost of drugs prescribed by each clinic during that month.


System requirements specification

1.1 On the last working day of each month, a summary of the drugs prescribed, their cost and the prescribing clinics shall be generated.
 1.2 The system shall automatically generate the report for printing after 17.30 on the last working day of the month.
 1.3 A report shall be created for each clinic and shall list the individual drug names, the total number of prescriptions, the number of doses prescribed and the total cost of the prescribed drugs.
 1.4 If drugs are available in different dose units (e.g. 10mg, 20 mg, etc.) separate reports shall be created for each dose unit.
 1.5 Access to all cost reports shall be restricted to authorized users listed on a management access control list.



Requirements readers

User requirements	→	Client managers System end-users Client engineers Contractor managers System architects
System requirements	→	System end-users Client engineers System architects Software developers
Software design specification	→	Client engineers (perhaps) System architects Software developers



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.

Functional and non-functional requirements

- **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

Example of functional requirements

- A user shall be able to search the appointments lists for all clinics.
- The system shall generate each day, for each clinic, a list of patients who are expected to attend appointments that day.
- Each staff member using the system shall be uniquely identified by his or her 8-digit employee number.

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


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



Non-functional requirements


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



14


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.

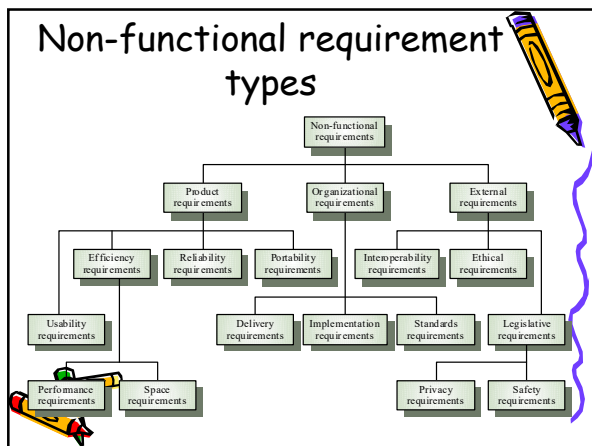


Non-functional classifications

- **External requirements**
 - Requirements which arise from factors which are **external to the system and its development process** e.g. interoperability requirements, legislative requirements, etc.




16



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.



18

Non-functional requirements implementation

- A single non-functional requirement, such as a security requirement, may generate a number of related functional requirements that define system services that are required.
 - It may also generate requirements that restrict existing requirements.



19

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.



20

Non-functional classifications

- **External requirements**
 - Requirements which arise from factors which are external to the system and its development process e.g. interoperability requirements, legislative requirements, etc.



21

Examples of nonfunctional requirements

- **Product requirement**
 - The System shall be available to all clinics during normal working hours (Mon-Fri, 0830-17.30). Downtime within normal working hours shall not exceed five seconds in any one day.
- **Organizational requirement**
 - Users of the system shall authenticate themselves using their health authority identity card.



22

Examples of nonfunctional requirements

- **External requirement**
 - The system shall implement patient privacy provisions as set out in HStan-03-2006-priv.



23


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



Goals and requirements


- **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**



25


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.



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)



27


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


- 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 be new functional requirements, constraints on existing requirements or define specific computations.



29

Domain requirements

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



30

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



User requirements

- Should describe functional and non-functional requirements so that they are understandable by system users who don't have detailed technical knowledge
- User requirements are defined using natural language, tables and diagrams



Guidelines for writing requirements

- Invent a standard format and use it for 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 !!!

System requirements

- More detailed specifications of user requirements
- Serve as a basis for designing the system
- May be used as part of the system contract
- System requirements may be expressed using system models



Requirements and design

- **In principle, requirements** should state **what the system should** do and the **design** should describe **how it does this**
 - 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



Questions

