Software Engineering

Software Requirements Engineering

- Process of establishing the services that the customer requires from a system and the constraints under which the system operates & is developed
- Descriptions & constraints
- Can be in the form of:
  - × High-level abstract statement
  - × Detailed mathematical function

Functional, Non-functional & Domain requirements

Functional requirements
- → Statements of services the system should provide, react to particular inputs, behave in particular situations
- → state what the system should not do

Non-functional requirements
- → Constraints on the services i.e. timing, development, standards
- → Apply to the system as a whole, not individual features

Domain requirements
- → Constraints on the system from the domain of operation

Functional Requirements

→ Describe functionality or system services

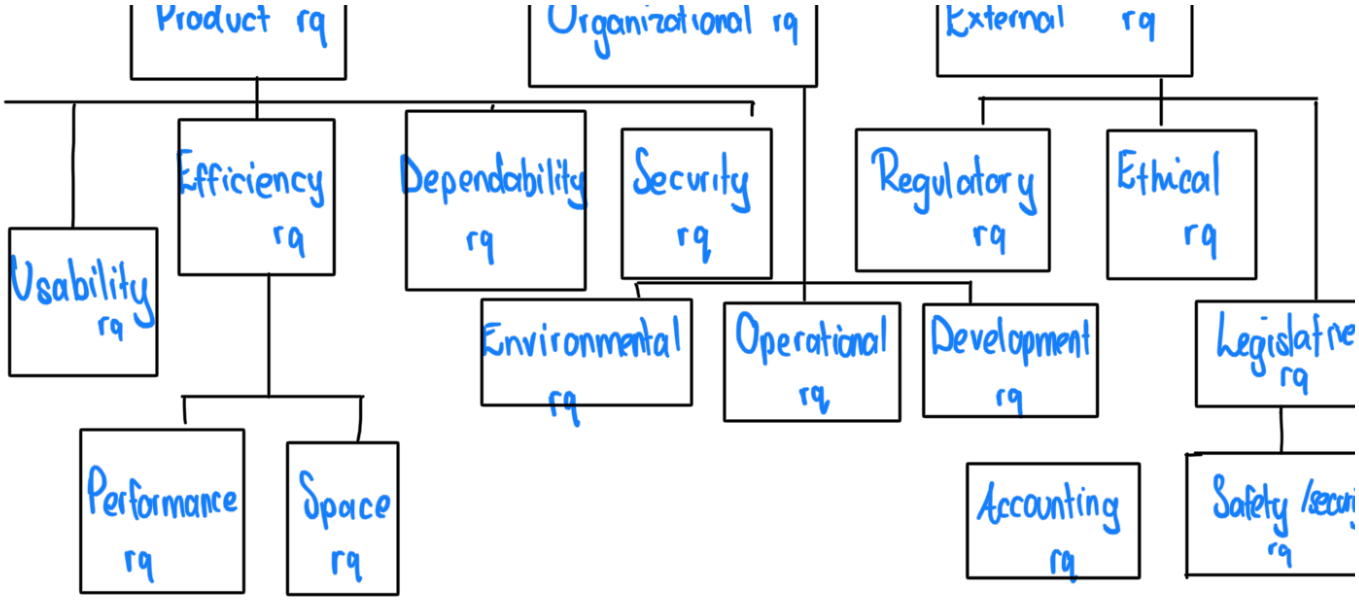
→ Depends on software type
expected users
type of system

requirements imprecision
→ when requirements are not precisely stated it can be interpreted in different ways

requirements completeness & consistency
→ should include descriptions of all facilities required
→ there should be no conflicts & contradictions

Non-Functional Requirements

- Define system properties & constraints
- Not the most important but a system is useless without them



Property	Measure
Speed	Processed transactions / second User / event response time Screen refresh time
Size	Mbytes No. of ROM chips
Ease of use	Training time No. of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurence Availability
Robustness	Time to restart after failure Percentage of events causing ↓ Probability of data corruption
Portability	% of target dependent statements Number of target systems

Domain Requirements

→ Imposes requirements on the system

Domain requirement problems

Understandability

→ Requirements expressed in the

Software Requirements Document

→ Official statement of what is requied

→ Not a design document

Requirements Specification

- Understandable
- Detailed requirements
- Part of the contract
- Writing specifications
  - × natural lang.
  - × structural natural lang
  - × design description lang.
  - × Graphical notations
  - × Mathematical Specifications

Requirements & Design

Requirements → what the system should do

Design → how the system does it

Problems with Natural Language

language of the application domain
→ ∴ often not understood by SE's

Implicitness
→ Domain specialists understand the area so well that they don't think of making domain requirements explicit

- ✓ Lack of clarity
- ✓ Requirements confusion
- ✓ Requirements amalgamation

Tabular Specification
✓ Used to supplement natural language

System Modeling

System Modeling → developing abstract models of a system with each model rep. a different view

Helps with
1. Understand functionality of app
2. Communicating with customers

UML uses a set of symbols to rep graphically the various components & r.ships with a system
- its universal

Importance of models
1. Clarity
2. Basis for discussing strengths and weaknesses
3. Lead to requirements

System Perspectives
1. External perspective → context/env.
2. Interaction perspective → interactions
3. Structural perspective → organization & structures
4. Behavioral perspective → dynamic behavior

Data Flow Diagrams (DFDs)

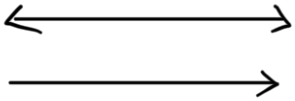
- Uses various symbols to show how the system transforms input data into useful info.
- Shows how data moves not logic or processing ∴ what not how

Four basic symbols
- ✓ Processes
- ✓ Data flows
- ✓ Data stores
- ✓ Entities

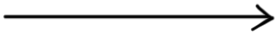
× always in capital

Process Symbol
✓ Receives data and produces output
✓ Rectangle w rounded corners

Data flow symbol
✓ Path for data to move from one part of the info system to another



if input & output is the same



Avoid

- Spontaneous generation } two outputs
- Black hole → two inputs
- Gray hole → irrelevant input

Data Store Symbols

✓ Represent data that the system stores because 1 or more processes need the data
✓ Flat rectangle open on one side
✓ MUST be connected to a process via a data flow

Entity Symbols

✓ External entities provide data to the system or receive output
Source entity → supply data
Sink entity → receive data

x Always connected to a process

Drawing Data Flow Diag.

Guidlines

1. Fit on one page
2. Name of info system is the process name of the contex diag.
3. Unique names
4. Don't cross lines
5. Provide as much data as possible

Context Diagram

✓ Place a single process in the center

Software Testing