

Software Requirements (utvalgte foiler fra Kap 6 og 7 i Sommerville)

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.

Definitions and specifications

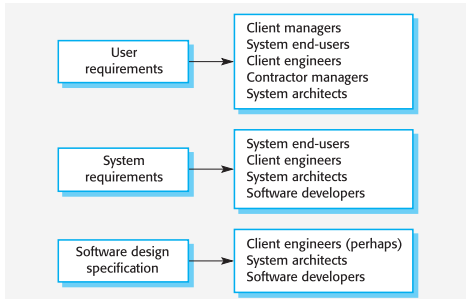
User requirement definition

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.
- 1.5 When a user selects an icon representing an external file, the effect of that selection is to apply the tool associated with the type of the external file to the file represented by the selected icon.

Requirements readers



©Ian Sommerville 2004

Software Engineering, 7th edition, Chapter 6

Slide 4

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.

©Ian Sommerville 2004

Software Engineering, 7th edition, Chapter 6

Slide 5

Non-functional requirements

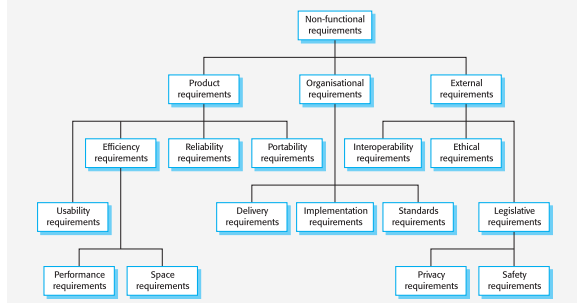
- These define system properties and constraints e.g. reliability, response time and storage requirements. Constraints are I/O device capability, system representations, etc.
- Process requirements may also be specified mandating a particular CASE system, programming language or development method.
-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

©Ian Sommerville 2004

Software Engineering, 7th edition, Chapter 6

Slide 6

Non-functional requirement types



©Ian Sommerville 2004

Software Engineering, 7th edition, Chapter 6

Slide 7

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

©Ian Sommerville 2004

Software Engineering, 7th edition, Chapter 6

Slide 8

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.

©Ian Sommerville 2004

Software Engineering, 7th edition, Chapter 6

Slide 9

Guidelines for writing requirements

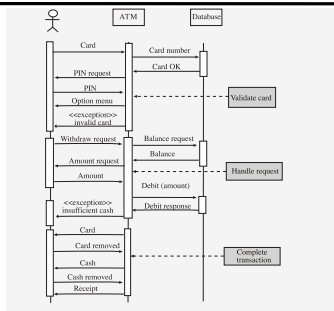
- Invent a standard format and use it for all requirements.
- Use language in a consistent way. Use shall for mandatory requirements, should for desirable requirements.
- Use text highlighting to identify key parts of the requirement.
- Avoid the use of computer jargon.

Form-based node specification

Insulin Pump Control Software/SRS/3.3.2

Function Compute insulin dose: Safe sugar level
Description Computes the dose of insulin to be delivered when the current measured sugar level is in the safe zone between 3 and 7 units.
Inputs Current sugar reading (r2), the previous two readings (r0 and r1)
Source Current sugar reading from sensor. Other readings from memory.
Outputs CompDose \$ the dose in insulin to be delivered
Destination Main control loop
Action: CompDose is zero if the sugar level is stable or falling or if the level is increasing but the rate of increase is decreasing. If the level is increasing and the rate of increase is increasing, then CompDose is computed by dividing the difference between the current sugar level and the previous level by 4 and rounding the result. If the result, is rounded to zero then CompDose is set to the minimum dose that can be delivered.
Requires Two previous readings so that the rate of change of sugar level can be computed.
Pre-condition The insulin reservoir contains at least the maximum allowed single dose of insulin.
Post-condition r0 is replaced by r1 then r1 is replaced by r2
Side-effects None

Sequence diagram of ATM withdrawal



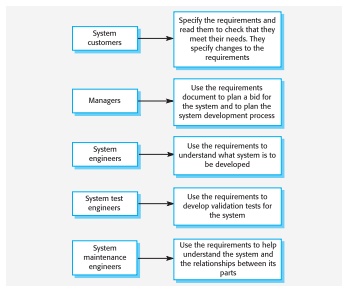
Interface specification

- Most systems must operate with other systems and the operating interfaces must be specified as part of the requirements.
- Three types of interface may have to be defined
 - Procedural interfaces;
 - Data structures that are exchanged;
 - Data representations.
- Formal notations are an effective technique for interface specification.

The 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 is NOT a design document. As far as possible, it should set of WHAT the system should do rather than HOW it should do it

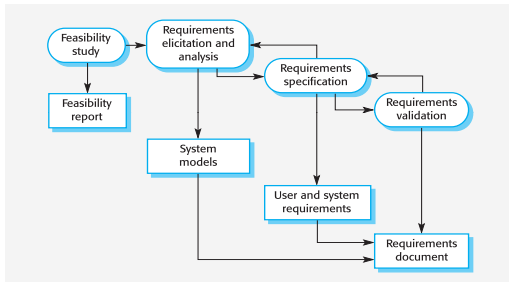
Users of a requirements document



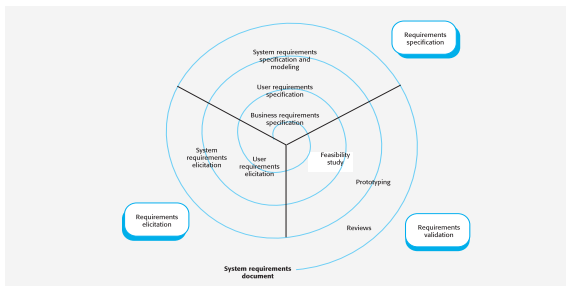
IEEE requirements standard

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

The requirements engineering process



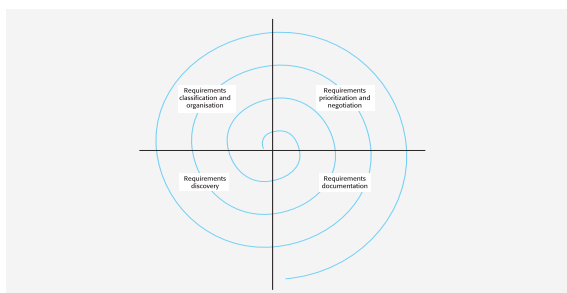
Requirements engineering



Problems of requirements 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 requirements spiral



ATM stakeholders

- Bank customers
- Representatives of other banks
- Bank managers
- Counter staff
- Database administrators
- Security managers
- Marketing department
- Hardware and software maintenance engineers
- Banking regulators

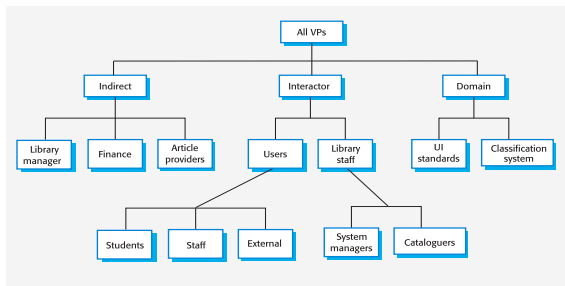
Viewpoints

- Viewpoints are a way of structuring the requirements to represent the perspectives of different stakeholders. Stakeholders may be classified under different viewpoints.
- This multi-perspective analysis is important as there is no single correct way to analyse system requirements.

Types of viewpoint

- **Interactor viewpoints**
 - People or other systems that interact directly with the system. In an ATM, the customer's and the account database are interactor VPs.
- **Indirect viewpoints**
 - Stakeholders who do not use the system themselves but who influence the requirements. In an ATM, management and security staff are indirect viewpoints.
- **Domain viewpoints**
 - Domain characteristics and constraints that influence the requirements. In an ATM, an example would be standards for inter-bank communications.

LIBSYS viewpoint hierarchy



Scenarios

- Scenarios are real-life examples of how a system can be used.
- They should include
 - A description of the starting situation;
 - A description of the normal flow of events;
 - A description of what can go wrong;
 - Information about other concurrent activities;
 - A description of the state when the scenario finishes.

Requirements 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?
