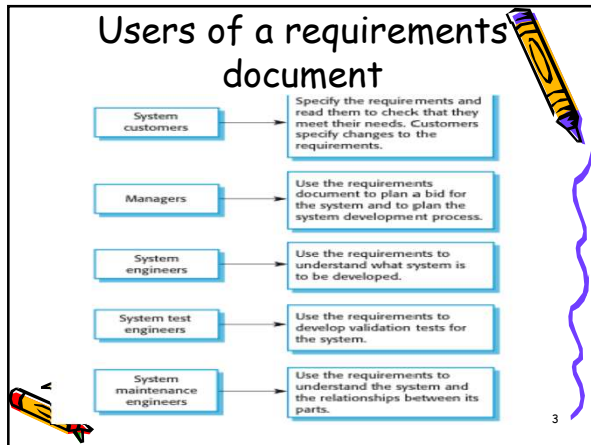


Requirements Engineering II

The software requirements document

- The software 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 be a set of WHAT the system should do rather than HOW it should do it.



Requirements document variability

- Information in requirements document depends on type of system and the approach to development used.
- Systems developed incrementally will, typically, have less detail in the requirements document.
- Requirements documents standards have been designed e.g. IEEE standard.
 - These are mostly applicable to the requirements for large systems engineering projects.

The structure of a requirements document


Chapter	Description
Preface	This should define the expected readership of the document and describe its version history, including a rationale for the creation of a new version and a summary of the changes made in each version.
Introduction	This should describe the need for the system. It should briefly describe the system's functions and explain how it will work with other systems. It should also describe how the system fits into the overall business or strategic objectives of the organization commissioning the software.
Glossary	This should define the technical terms used in the document. You should not make assumptions about the experience or expertise of the reader.
User requirements definition	Here, you describe the services provided for the user. The nonfunctional system requirements should also be described in this section. This description may use natural language, diagrams, or other notations that are understandable to customers. Product and process standards that must be followed should be specified.
System architecture	This chapter should present a high-level overview of the anticipated system architecture, showing the distribution of functions across system modules. Architectural components that are reused should be highlighted.

The structure of a requirements document

Chapter	Description
System requirements specification	This should describe the functional and nonfunctional requirements in more detail. If necessary, further detail may be added to the nonfunctional requirements. Interfaces to other systems may be defined.
System models	This might include graphical system models showing the relationships between the system components and the system and its environment. Examples of possible models are object models, data-flow models, or semantic data models.
System evolution	This should describe the fundamental assumptions on which the system is based and any anticipated changes due to hardware evolution, changing user needs, and so on. This section is useful for system designers as it may help them avoid design decisions that would constrain likely future changes to the system.
Appendices	These should provide detailed, specific information that is related to the application being developed; for example, hardware and database descriptions. Hardware requirements define the minimal and optimal configurations for the system. Database requirements define the logical organization of the data used by the system and the relationships between data.
Index	Several indexes to the document may be included. As well as a normal alphabetic index, there may be an index of diagrams, an index of functions, and so on.

Requirements specification


- The process of writing down the user and system requirements in a requirements document.
- User requirements have to be understandable by end-users and customers who do not have a technical background.



7

Requirements specification


- 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.



8

Ways of writing a system requirements 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.



30

Natural language specification


- Requirements are written as natural language sentences supplemented by diagrams and tables.
- Used for writing requirements because it is expressive, intuitive and universal. This means that the requirements can be understood by users and customers.



10

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.



11

Example requirements

- 3.2 The system shall measure the blood sugar and deliver insulin, if required, every 10 minutes. *(Changes in blood sugar are relatively slow so more frequent measurement is unnecessary; less frequent measurement could lead to unnecessarily high sugar levels.)*
- 3.6 The system shall run a self-test routine every minute with the conditions to be tested and the associated actions defined in Table 1. *(A self-test routine can discover hardware and software problems and alert the user to the fact the normal operation may be impossible.)*



12

Requirements engineering 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

13

Requirements engineering processes

- Requirements elicitation;
- Requirements analysis;
- Requirements validation;
- Requirements management.
- In practice, RE is an iterative activity in which these processes are interleaved.

14

Requirements 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.

15

Requirements elicitation and analysis

- May involve end-users, managers, engineers involved in maintenance, domain experts, trade unions, etc. These are called *stakeholders*.

16

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.

17

Problems of requirements analysis

- Organisational and political factors may influence the system requirements.
- The requirements change during the analysis process.
 - New stakeholders may emerge and the business environment may change.

18

Requirements validation

- Concerned with demonstrating that the requirements define the system that the customer really wants.
- Requirements error costs are high so validation is very important
 - Fixing a requirements error after delivery may cost up to 100 times the cost of fixing an implementation error.



19

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?



20

Requirements checking

- Realism.
 - Can the requirements be implemented given available budget and technology
- Verifiability.
 - Can the requirements be checked?



21

Requirements validation techniques

- 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.



22

Requirements reviews

- 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



23

Review checks

- Verifiability
 - Is the requirement realistically testable?
- Comprehensibility
 - Is the requirement properly understood?
- Traceability
 - Is the origin of the requirement clearly stated?



24

Review checks

- Adaptability
 - Can the requirement be changed without a large impact on other requirements?

25

Requirements management

- Requirements management is the process of managing changing requirements during the requirements engineering process and system development.
- New requirements emerge as a system is being developed and after it has gone into use.

26

Requirements management

- You need to keep track of individual requirements and maintain links between dependent requirements so that you can assess the impact of requirements changes.
 - You need to establish a formal process for making change proposals and linking these to system requirements

27

Changing requirements

- The business and technical environment of the system always changes after installation.
 - New hardware may be introduced, it may be necessary to interface the system with other systems, business priorities may change (with consequent changes in the system support required), and new legislation and regulations may be introduced that the system must necessarily abide by.

28

Changing requirements

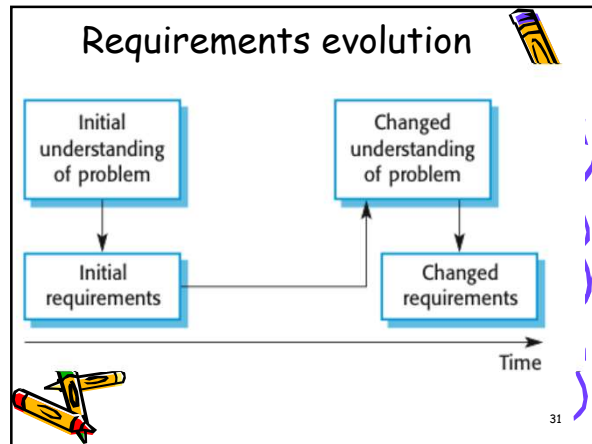
- The people who pay for a system and the users of that system are rarely the same people.
 - System customers impose requirements because of organizational and budgetary constraints.
 - These may conflict with end-user requirements and, after delivery, new features may have to be added for user support if the system is to meet its goals.

29

Changing requirements

- Large systems usually have a diverse user community, with many users having different requirements and priorities that may be conflicting or contradictory.
 - The final system requirements are inevitably a compromise between them and, with experience, it is often discovered that the balance of support given to different users has to be changed.

30



Requirements management planning

- Establishes the level of requirements management detail that is required.
- Requirements management decisions:
 - *Requirements identification*: Each requirement must be uniquely identified so that it can be cross-referenced with other requirements.
 - *A change management process*: This is the set of activities that assess the impact and cost of changes.

32

Requirements management planning

- *Traceability policies*: These policies define the relationships between each requirement and between the requirements and the system design that should be recorded.
- *Tool support*: Tools that may be used range from specialist requirements management systems to spreadsheets and simple database systems.

33

Requirements change management

- Deciding if a requirements change should be accepted
 - *Problem analysis and change specification*
 - During this stage, the problem or the change proposal is analyzed to check that it is valid. This analysis is fed back to the change requestor who may respond with a more specific requirements change proposal, or decide to withdraw the request.

34

Requirements change management

- *Change analysis and costing*
 - The effect of the proposed change is assessed using traceability information and general knowledge of the system requirements.
 - Once this analysis is completed, a decision is made whether or not to proceed with the requirements change.

35

Requirements change management

- *Change implementation*
 - The requirements document and, where necessary, the system design and implementation, are modified.
 - Ideally, the document should be organized so that changes can be easily implemented

36

