Requirements Engineering

Lecture # 23
24 March

Rubab Jaffar
rubab.jaffar@nu.edu.pk

Intro. to Software Engineering
SE-110

TODAY'S OUTLINE

- Functional and non-functional requirements

- Requirements engineering processes

- Requirements elicitation

- Requirements specification

- Requirements validation

- Requirements change

STRUCTURED SPECIFICATIONS

- An approach to writing requirements where the freedom of the requirements writer is limited and requirements are written in a standard way.

- This works well for some types of requirements e.g. requirements for embedded control system but is sometimes too rigid for writing business system requirements.

FORM-BASED SPECIFICATIONS

- Definition of the function or entity.

- Description of inputs and where they come from.

- Description of outputs and where they go to.

- Information about the information needed for the computation and other entities used.

- Description of the action to be taken.

- Pre and post conditions (if appropriate).

- The side effects (if any) of the function.

A STRUCTURED SPECIFICATION OF A REQUIREMENT FOR AN INSULIN PUMP

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.

A STRUCTURED SPECIFICATION OF A REQUIREMENT FOR AN INSULIN PUMP

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.

Requirements

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.

TABULAR SPECIFICATION

- Used to supplement natural language.

- Particularly useful when you have to define a number of possible alternative courses of action.

- For example, the insulin pump systems bases its computations on the rate of change of blood sugar level and the tabular specification explains how to calculate the insulin requirement for different scenarios.

TABULAR SPECIFICATION OF COMPUTATION FOR AN INSULIN PUMP

Condition	Action
Sugar level falling ($r2 < r1$)	CompDose = 0
Sugar level stable ($r2 = r1$)	CompDose = 0
Sugar level increasing and rate of increase decreasing (($r2 - r1$) < ($r1 - r0$))	CompDose = 0
Sugar level increasing and rate of increase stable or increasing (($r2 - r1$) ≥ ($r1 - r0$))	CompDose = round (($r2 - r1$)/4) If rounded result = 0 then CompDose = MinimumDose

USE CASES

- Use-cases are a kind of scenario that are included in the UML.

- Use cases 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 (see Chapter 5).

- UML sequence diagrams may be used to add detail to use-cases by showing the sequence of event processing in the system.
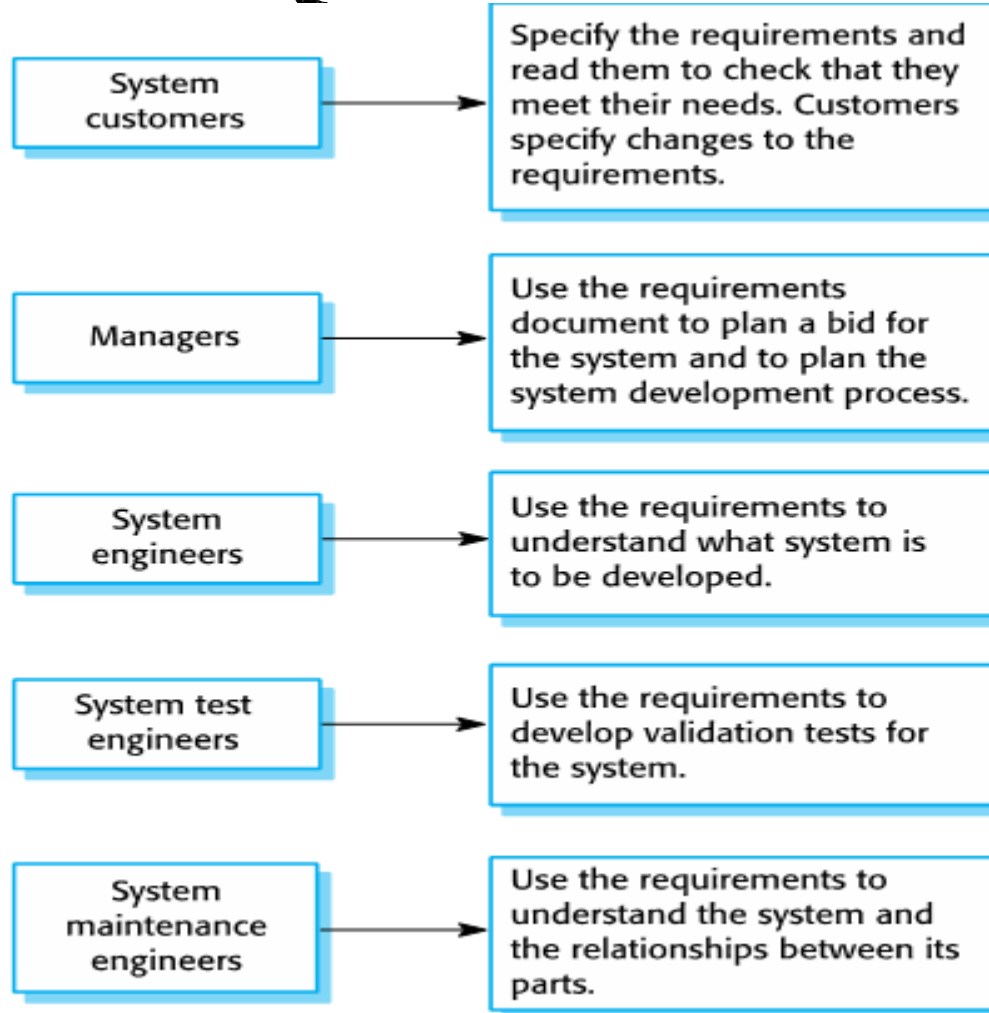
USE CASES FOR THE MENTCARE SYSTEM
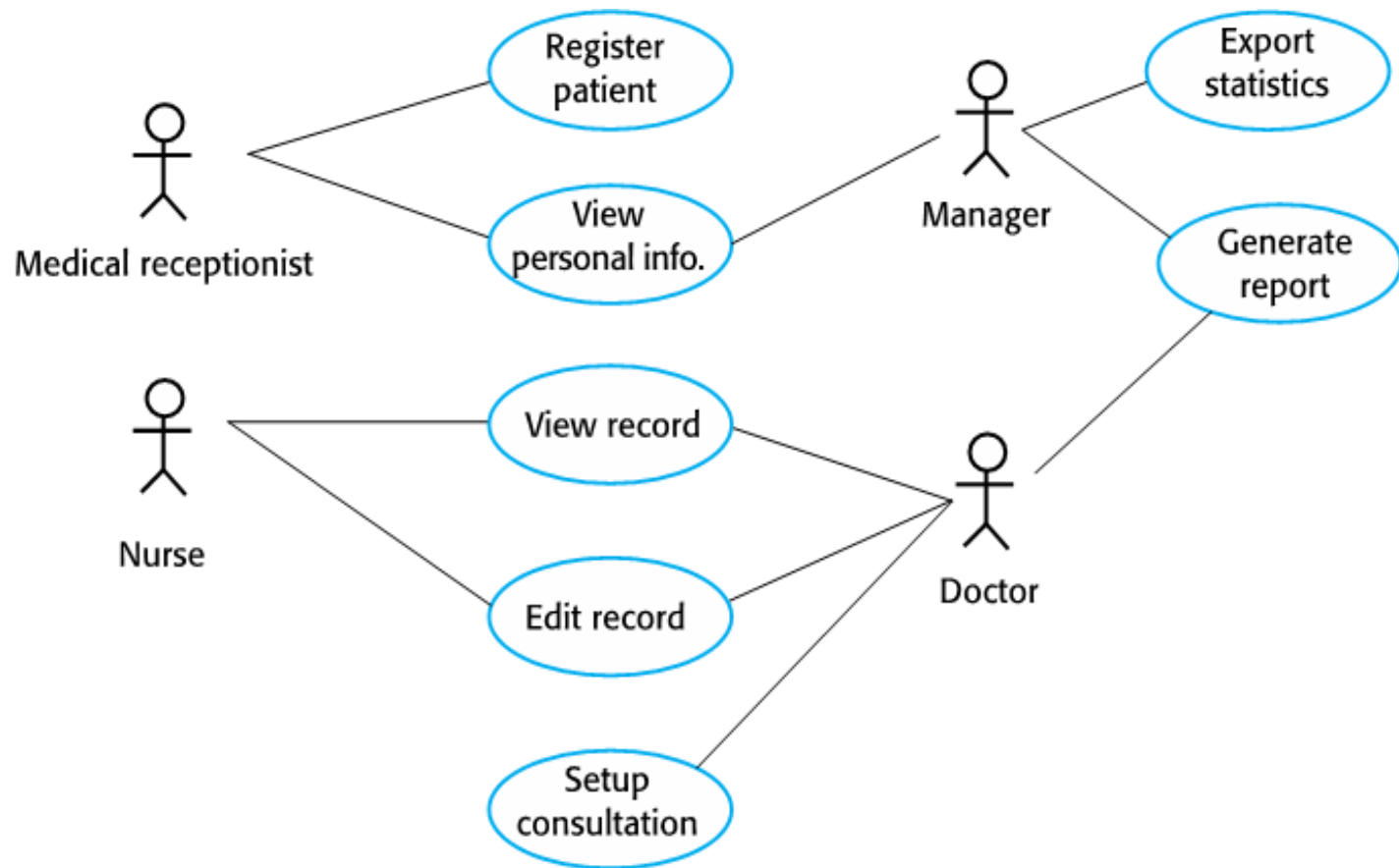
THE SOFTWARE REQUIREMENTS DOCUMENT

- 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 set of WHAT the system should do rather than HOW it should do it.

USERS OF A REQUIREMENTS DOCUMENT

System customers	Specify the requirements and read them to check that they meet their needs. Customers specify changes to the requirements.
Managers	Use the requirements document to plan a bid for the system and to plan the system development process.
System engineers	Use the requirements to understand what system is to be developed.
System test engineers	Use the requirements to develop validation tests for the system.
System maintenance engineers	Use the requirements to understand the system and the relationships between its parts.

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 also 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 VALIDATION
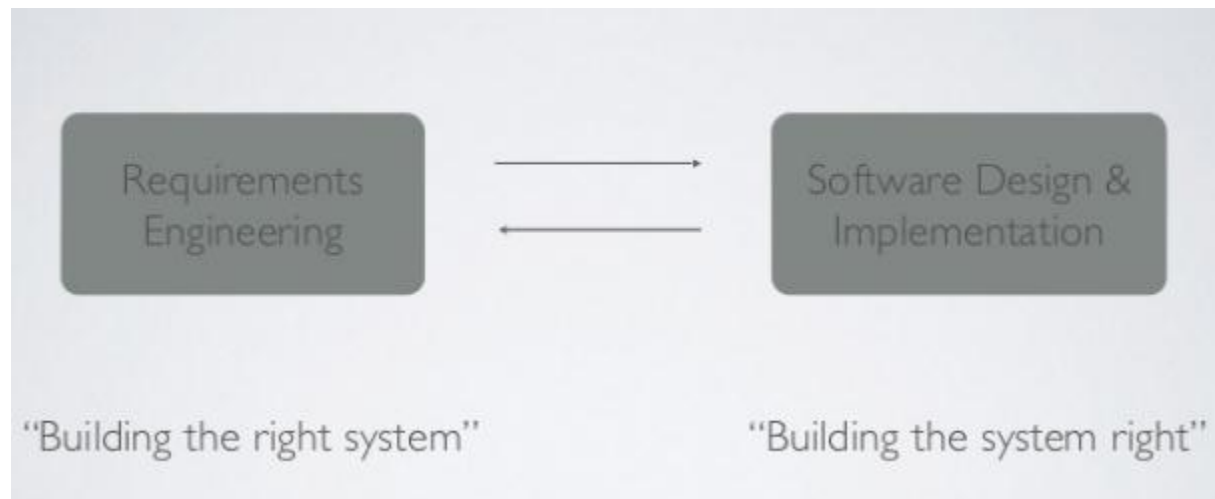
REQUIREMENTS VALIDATION

- 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.

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?

REQUIREMENTS VALIDATION TECHNIQUES

- Requirements reviews
  - Systematic manual analysis of the requirements.

- Prototyping
  - Using an executable model of the system to check requirements. Covered in Chapter 2.

- Test-case generation
  - Developing tests for requirements to check testability.

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.
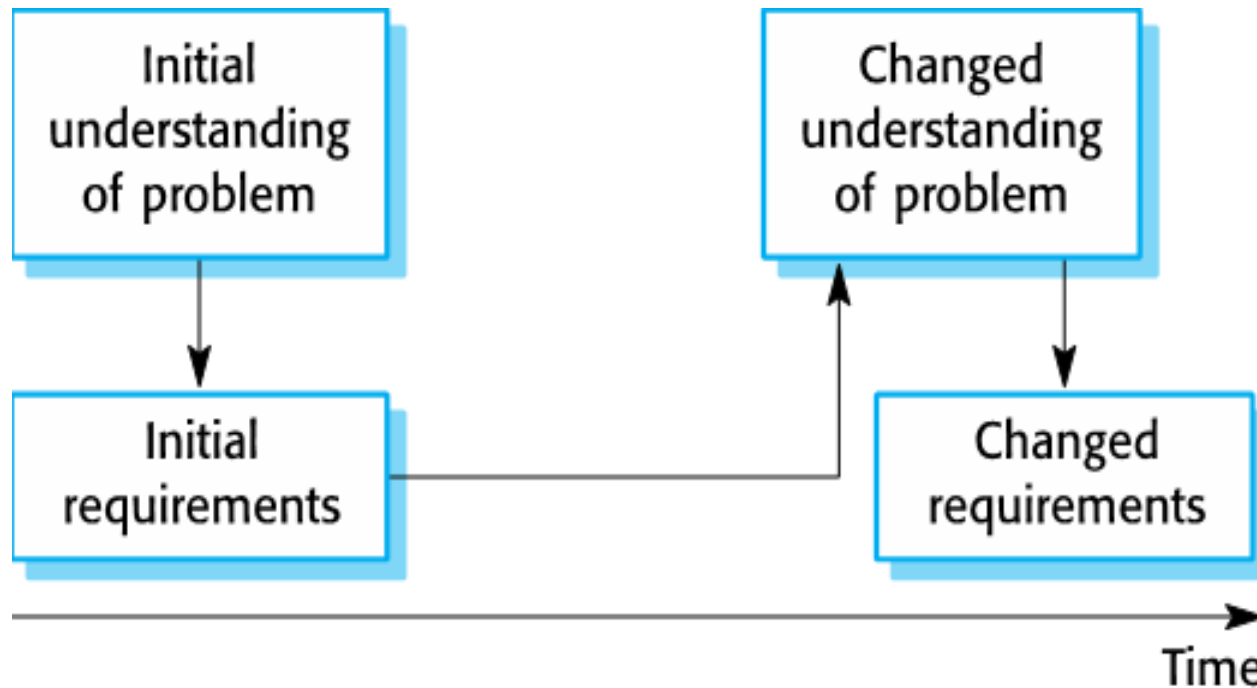
REQUIREMENTS CHANGE

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.

- 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.

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.

REQUIREMENTS EVOLUTION

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.

- 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.

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.

  ▪ *Traceability policies:* These policies define the relationships between each requirement and between the requirements and the system design that should be recorded.

  ▪ Tool Support: that may be used range from specialist requirements management systems to spreadsheets and simple database systems.

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.
  - *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.
  - 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.

REQUIREMENTS CHANGE MANAGEMENT

Identified problem → Problem analysis and change specification → Change analysis and costing → Change implementation → Revised requirements

That is all