REQUIREMENT ENGINEERING

BY:HIRA SHAHID

REVISION

- Functional Requirements
- Non- Functional Requirements
- Characteristics of good requirements
- Incomplete/Ambiguous Requirements
- Usability Requirements

DOMAIN REQUIREMENTS

- The system's operational domain imposes requirements on the system.
  - The system safety shall be assured according to standard IEC 60601-1:Medical Electrical Equipment – Part 1:General Requirements for Basic Safety and Essential Performance.
  - 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.
- If domain requirements are not satisfied, the system may be unworkable.

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.

CHARACTERISTICS OF GOOD REQUIREMENTS

1. Numbered
2. Inspected
3. Unambiguous
4. Testable
5. Complete
6. Consistent
7. Understandabl e
8. Traceable
9. Feasible
10. Modifiable
11. Legal

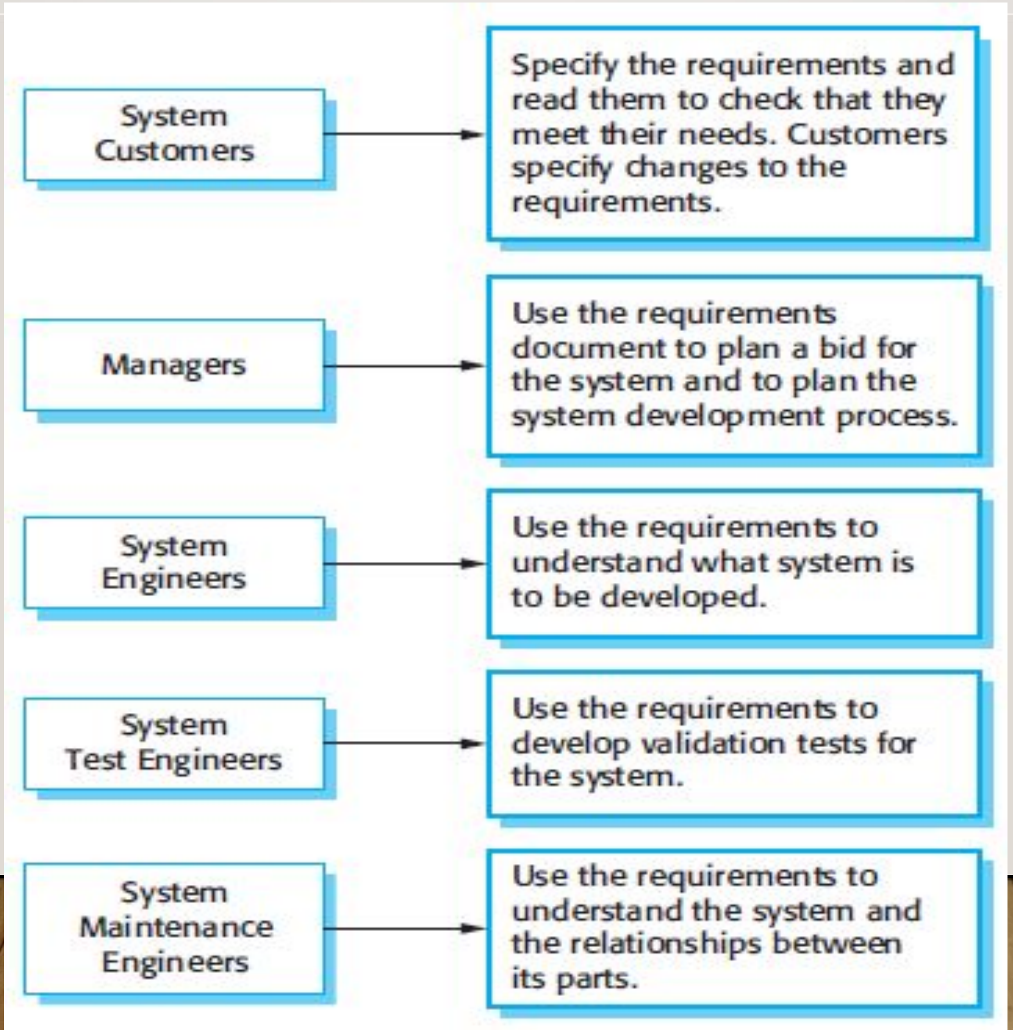
THE SOFTWARE REQUIREMENTS DOCUMENT

- 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.

AGILE METHODS AND REQUIREMENTS

- Many agile methods argue that producing a requirements document is a waste of time as requirements change so quickly.
- The document is therefore always out of date.
- Methods such as XP use incremental requirements engineering and express requirements as ‘user stories’.
- This is practical for business systems but problematic for systems that require a lot of pre-delivery analysis (e.g. critical systems) or systems developed by several teams.

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.

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.
- **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.

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.
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

NATURAL LANGUAGE SPECIFICATION

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

EXAMPLE REQUIREMENTS FOR THE INSULIN PUMP SOFTWARE SYSTEM

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.)

GUIDELINES FOR WRITING REQUIREMENTS

- **Invent a standard format** and use it for all requirements.(single sentence!!)
- **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** (specialised words).
- **Include an explanation** (rationale) of why a requirement is necessary.

PROBLEMS WITH NATURAL LANGUAGE

- Lack of clarity
  - it is sometimes difficult to use language in a precise and unambiguous way, without making the document wordy and difficult to read
- Requirements confusion
  - Functional, non-functional requirements and design information tend to be mixed-up.
- Requirements amalgamation
  - Several different requirements may be expressed together in a single requirement.

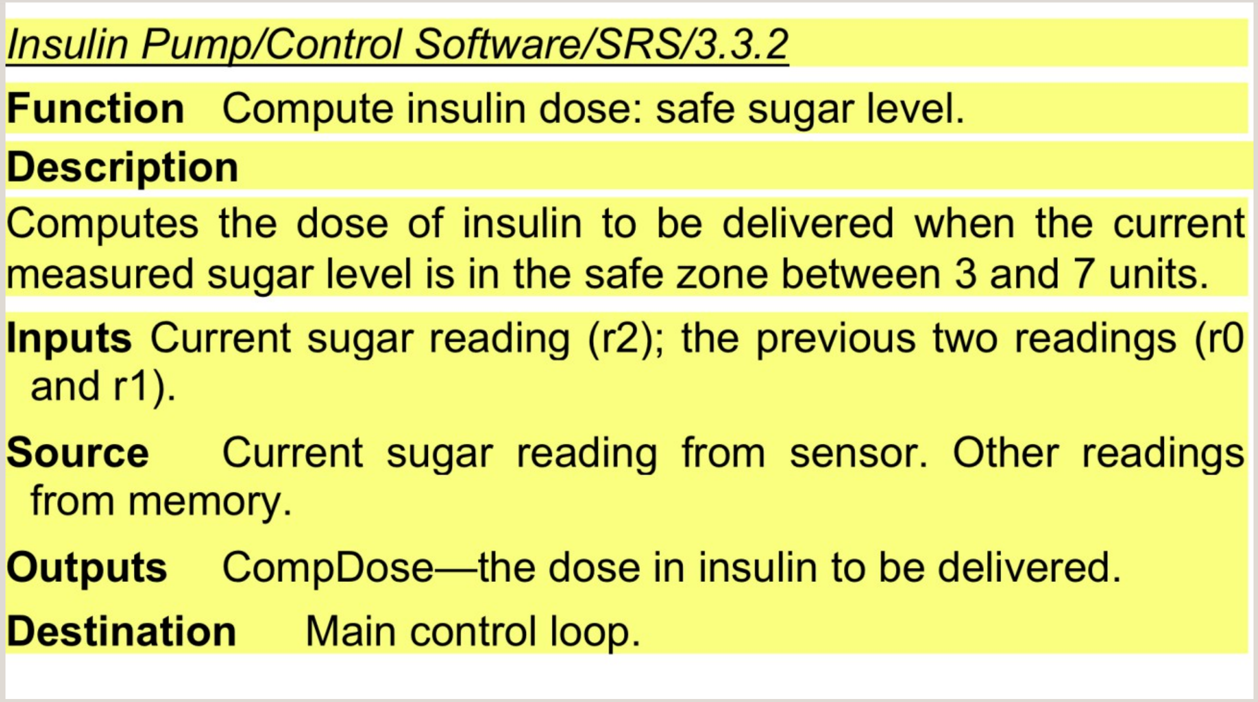
STRUCTURED SPECIFICATIONS

- 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.
- Description of inputs and where they come from.
- Description of outputs and where they go to.
- 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.

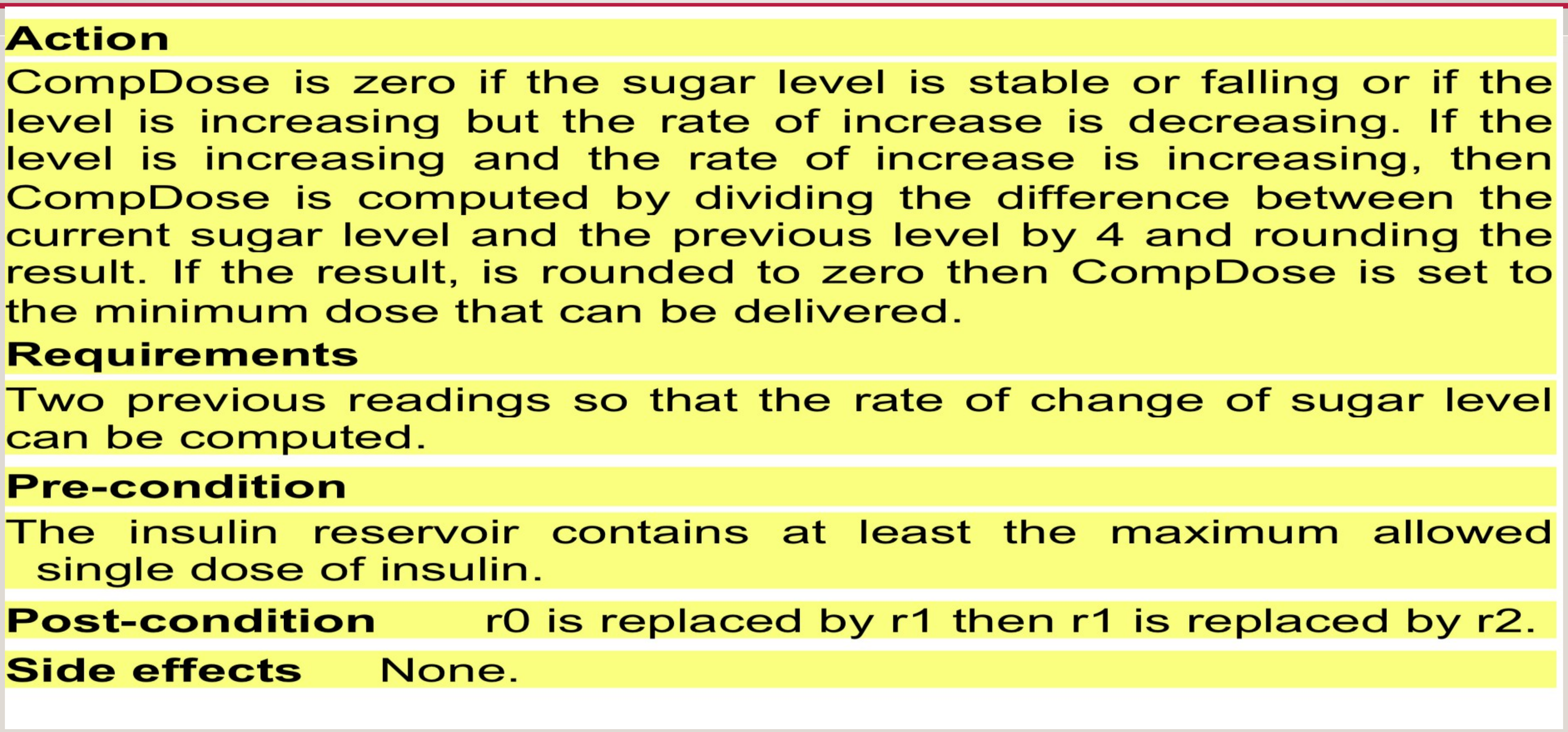
Inputs Current sugar reading (r2); the previous two readings (r0 and r1).

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

THANK
S