Phase – 1: Planning and Analysis

Requirement Engineering, Requirements, Types of requirements, Functional and non-functional requirements

Suleman Shahid

Based on slides from

1

"The hardest single part of building a software system is deciding what to build…No other part of the work so cripples the resulting system if done wrong. No other part is difficult to rectify later"
-Fred Brooks

2

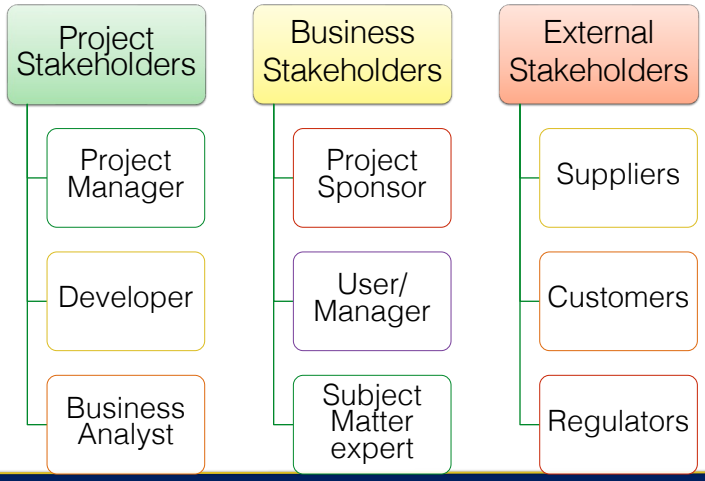
POLL

Requirements – Examples

- The system shall maintain the record of all payments made to employees on account of salaries, bonuses, travel/daily allowances, medical allowances, etc.
- The system shall interface with the central computer to send daily sales and inventory data from every retail store

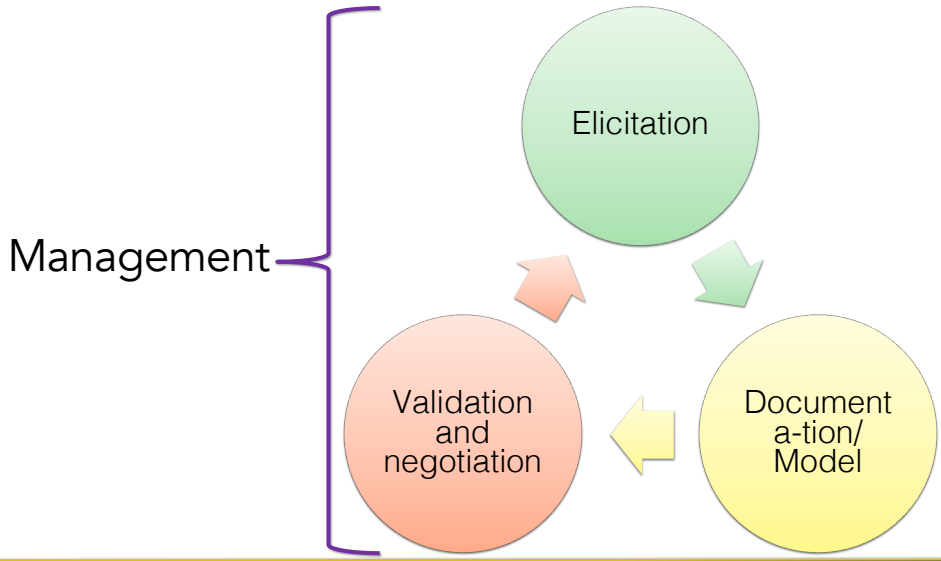
Stakeholders

- Stakeholder is anyone affected by or interested in the business/project.



5

Requirement Engineering Activities

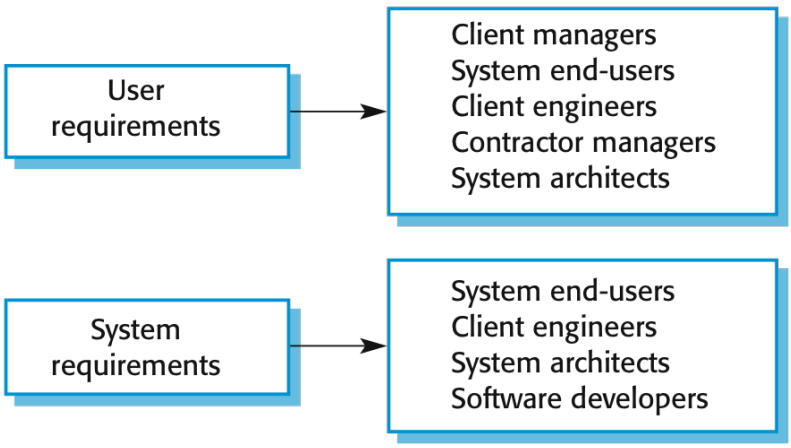


6

Requirements – Types

- User requirements
  - What services a system will provide to end users
  - Constraints under which system must operate
  - Written for customers
  - No technical details
- System Requirements
  - Detailed description of functions, services and operational constraints

Requirements endpoints



Requirement – Example

User Requirement

1. The MHC-PMS shall generate monthly management reports showing the cost of drugs prescribed by each clinic during that month.

System Requirement

1.1 On the last working day of each month, a summary of the drugs prescribed, their cost, and the prescribing clinics shall be generated.
1.2 The system shall automatically generate the report for printing after 17.30 on the last working day of the month.
1.3 A report shall be created for each clinic and shall list the individual drug names, the total number of prescriptions, the number of doses prescribed, and the total cost of the prescribed drugs.
1.4 If drugs are available in different dose units (e.g., 10 mg, 20 mg) separate reports shall be created for each dose unit.
1.5 Access to all cost reports shall be restricted to authorized users listed on a management access control list.

9

Requirements – Example

User Requirement

- The system shall accurately compute sale totals including discounts, taxes, refunds, and rebates; print an accurate receipt; and update inventory counts accordingly.

System requirement ?

10

Requirements – Example

User Requirement

- The system shall accurately compute sale totals including discounts, taxes, refunds, and rebates; print an accurate receipt; and update inventory counts accordingly.

Some System requirement

- Each sale shall be assigned a sales ID.
- Each sale may have one or more sales items.
- Each sale may have one or more rebates.
- Each sale may have only one receipt printed.
- ……

Requirements – Example

System Specification

1.1 The system shall assign a unique sales ID number to each sale transaction.

1.1.1 Each sales ID may have zero or more sales items associated with it, but each sales item must be assigned to exactly one sales ID

….

Requirements – Example

User requirement

- Automated bag processing system: The system shall be able to process 5 bags per minute

System Requirement?

Example (cont.)

User requirement

- Automated bag processing system: The system shall be able to process 5 bags per minute

- **System Requirement**

1.1 The system shall be able to process 5 baggage events per minute in operational mode.

1.1.1 If more than 05 baggage events occur in a one-minute interval, then the system shall …

1.1.2 [more exception handling] …

System Requirements – Types

- Functional Requirements
  - Functional requirements define the functionality that the system to be developed offers.
- Quality Requirements
  - A quality requirement is a requirement that pertains to a quality concern that is not covered by functional requirements.
  - Non-Functional requirements
  - Influence system architecture
- Constraints
  - A constraint is a requirement that limits the solution space beyond what is necessary for meeting the given functional requirements and quality requirements

Functional Requirements – FR

A user shall be able to search the appointments lists for all clinics.

The system shall generate each day, for each clinic, a list of patients who are expected to attend appointments that day.

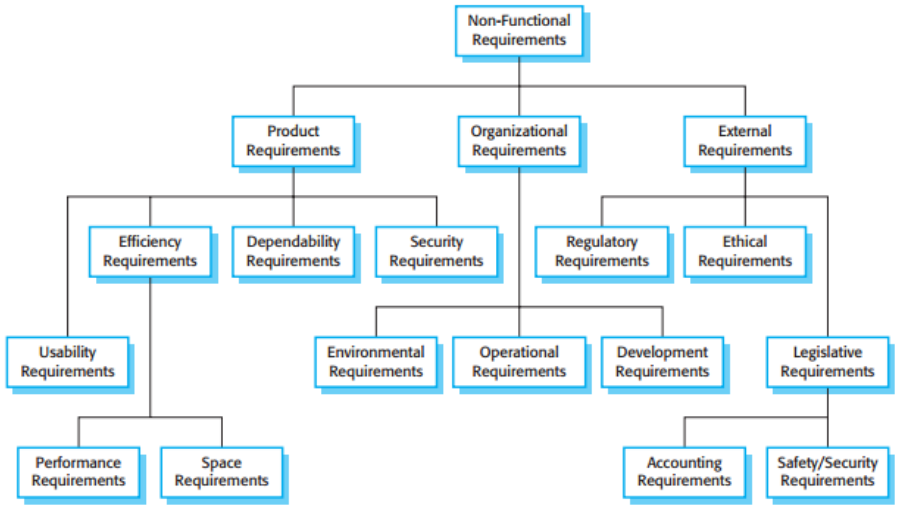
Each staff member using the system shall be uniquely identified by his or her eight-digit employee number.

Non-Functional Requirements – NFRs

- Differ in the implementation
  - Affects the overall architecture of the system
- May result in many functional requirements
  - May define new set of services

R – 001: The system shall generate list of appointments (FR) efficiently (NF
R – 002: Data must be stored (FR) securely (NFR) the system

Non-Functional Requirements



Non-Functional Requirements Examples

PRODUCT REQUIREMENT
The MHC-PMS shall be available to all clinics during normal working hours (Mon–Fri, 08.30–17.30). Downtime within normal working hours shall not exceed five seconds in any one day.

ORGANIZATIONAL REQUIREMENT
Users of the MHC-PMS system shall authenticate themselves using their health authority identity card.

EXTERNAL REQUIREMENT
The system shall implement patient privacy provisions as set out in HStan-03-2006-priv.

Non-Functional Requirements

- System should be efficient and there must not be compromise on speed.
- System should use minimal storage
- It should be easy to use system
- System should be reliable
- System should be robust

Non-Functional Requirements

- Speed
  - Processed transactions/second
  - User event/response time
  - Screen refresh time
- Size
  - K-Bytes
- Ease of use
  - Training time
  - Number of help frames

Non-Functional Requirements

- 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

Requirement Specification (Repeat)

Requirement Document

- Informal
- Semi-Formal
- Formal
- Hybrid

Requirement Document – Informal

- Natural Language

> 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.*)

- Problems
  - ✧ Lack of clarity
  - ✧ Requirements confusion
  - ✧ Requirements amalgamation

Requirement Document – Informal

- Standard Natural language
  - Structured, form-based, tabular

Specification Document HMS

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.)*

Structured Specification

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.
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.

Structured Specification – 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

Tabular Specifications

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

Requirement Document – Semiformal

- Conceptual Models
  - Use Case
    - Overview of system functionality
  - Class Diagrams
    - Describes the static structure of the system
  - Activity Diagrams
    - Describes the dynamic behavior of the system
  - Statechart Diagram
    - Describes the dynamic behavior of the individual object

Requirement Document

- Formal
  - Mathematical specification
- Hybrid

Thank you!