SOFTWARE ENGINEERING & PROJECT MANAGEMENT (SEPM)

MODULE 2 – Requirements Analysis & Cost Estimation

Based on Mumbai University PYQs (2023–2025)

This document contains detailed descriptive answers for ALL Module 2 questions from the uploaded SEPM question bank.

Covered Questions:

1. Elaborate COCOMO Model for Cost Estimation
2. Explain Functional and Non-Functional Requirements
3. Importance of Requirement Analysis
4. Requirement Engineering Tasks
5. SRS for Hospital Management System
6. SRS for Railway Reservation System
7. SRS for University Management System
8. Cost Estimation Numerical
9. Equivalence Partitioning and Boundary Value Analysis
10. Test Case Design Questions

Q1. Elaborate the COCOMO Model for Cost Estimation.

Introduction

Software projects require proper planning before development begins. Organizations need to estimate:

- Development effort
- Project cost
- Time required
- Team size
- Resources required

Software Cost Estimation helps organizations predict these factors.

One of the most popular cost estimation techniques is the COCOMO Model.

COCOMO stands for:

Constructive Cost Model

It was proposed by Barry Boehm in 1981.

Definition of COCOMO Model

COCOMO is an empirical software cost estimation model used to estimate:

1. Development effort
2. Project duration
3. Staffing requirements
4. Development cost

based on the size of software.

Software size is measured in:

KLOC (Kilo Lines of Code)

Objectives of COCOMO

1. Estimate project cost accurately.
2. Predict development time.
3. Estimate manpower requirements.
4. Improve project planning.
5. Reduce project risks.

Types of COCOMO Models

COCOMO is divided into three categories:

1. Basic COCOMO

- Simple estimation model

- Uses only KLOC
- Suitable for small projects

---

2. Intermediate COCOMO

- Uses KLOC and cost drivers
- More accurate estimation
- Considers product, hardware, personnel, and project attributes

---

3. Detailed COCOMO

- Most advanced version
- Estimates effort phase-wise
- Used for large and complex projects

Software Project Categories in COCOMO

Projects are classified into three modes:

Mode	Description
Organic	Small and simple projects with experienced teams
Semi-Detached	Medium complexity projects
Embedded	Complex projects with strict constraints

---

1. Organic Mode

Characteristics

- Small teams
- Familiar technology
- Less rigid requirements
- Simple applications

Examples

- Payroll System
- Inventory Management System

2. Semi-Detached Mode

Characteristics

- Medium-sized projects
- Mixed experience team
- Moderate complexity

Examples

- Database Systems
- Compilers

---

3. Embedded Mode

Characteristics

- Very complex systems
- Strict hardware/software constraints
- Real-time systems

Examples

- Air Traffic Control System
- Medical Equipment Software

Basic COCOMO Equations

Effort Equation

```
Effort = a × (KLOC)^b
```

Where:

- Effort = Person-Months
- KLOC = Thousand Lines of Code
- a and b are constants

Development Time Equation

```
Development Time = c × (Effort)^d
```

Where:

- Development Time = Months
- c and d are constants

Constants Used in Basic COCOMO

Mode	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semi-Detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Numerical Example on COCOMO

Problem Statement

Assume:

- Project Size = 10 KLOC
- Organic Mode

Find: 1. Effort 2. Development Time

Step 1: Calculate Effort

Formula:

```
Effort = 2.4 × (10)^1.05
```

Calculation:

```
Effort ≈ 26.9 Person-Months
```

Step 2: Calculate Development Time

Formula:

```
Time = 2.5 × (26.9)^0.38
```

Calculation:

```
Time ≈ 8.7 Months
```

Step 3: Calculate Average Team Size

Formula:

```
Team Size = Effort / Time
```

Calculation:

```
Team Size = 26.9 / 8.7
          ≈ 3 Persons
```

Advantages of COCOMO

1. Simple and easy to use.
2. Accurate for traditional projects.
3. Helps in budgeting.
4. Helps in manpower planning.
5. Useful for project scheduling.

Disadvantages of COCOMO

1. Depends heavily on LOC estimation.
2. Not suitable for Agile projects.
3. Difficult for modern dynamic systems.
4. Accuracy depends on historical data.

Applications of COCOMO

- Software project planning
- Cost estimation
- Schedule prediction
- Resource management

Conclusion

COCOMO is one of the most widely used software estimation models. It helps organizations estimate effort, time, and project cost efficiently.

Q2. Explain Functional and Non-Functional Requirements.

Introduction

Requirements are the foundation of software development.

A software requirement specifies:

- What the system should do
- How the system should perform

Requirements are mainly classified into:

1. Functional Requirements
2. Non-Functional Requirements

Functional Requirements

Functional requirements describe the functions and services provided by the system.

They specify:

- Inputs
- Outputs
- Operations
- Processing logic

Functional requirements describe:

“What the system should do.”

Examples of Functional Requirements

1. User Login
2. User Registration
3. Report Generation
4. Payment Processing
5. Search Functionality
6. Ticket Booking
7. File Upload

Characteristics of Functional Requirements

1. Describe system behavior.
2. Define specific functionality.
3. Easy to test.
4. Directly related to user requirements.

Non-Functional Requirements

Non-functional requirements specify quality attributes of the system.

They describe:

“How the system should perform.”

Examples of Non-Functional Requirements

1. Security
2. Reliability
3. Scalability
4. Performance
5. Portability
6. Maintainability
7. Availability
8. Efficiency

Types of Non-Functional Requirements

Type	Description
Performance	Response time and speed
Security	Protection from unauthorized access
Reliability	Continuous operation without failure
Scalability	Ability to handle increased load
Portability	Ability to work on different platforms
Usability	Ease of use
Maintainability	Easy software maintenance

Difference Between Functional and Non-Functional Requirements

Functional Requirements	Non-Functional Requirements
Describe what system does	Describe how system performs
Feature-oriented	Quality-oriented
Directly visible to users	Usually not directly visible

Functional Requirements	Non-Functional Requirements
Example: Login System	Example: Fast Login Response
Mandatory operations	Quality constraints

Importance of Requirements

1. Help developers understand system.
2. Reduce development errors.
3. Improve customer satisfaction.
4. Provide project direction.
5. Help testing and validation.

Conclusion

Both functional and non-functional requirements are essential for building efficient, reliable, and high-quality software systems.

Q3. Importance of Requirement Analysis and Explain Requirement Engineering Tasks.

Introduction

Requirement Analysis is one of the most important phases of Software Engineering.

Incorrect requirements may lead to:

- Project failure
- Increased cost
- Delays
- Customer dissatisfaction

Requirement Analysis ensures that customer needs are properly understood.

Requirement Analysis

Requirement Analysis is the process of studying, understanding, documenting, and validating software requirements.

It helps developers understand:

- Customer expectations
- System functionality
- Project scope

Importance of Requirement Analysis

1. Clear Understanding of Requirements

Developers understand customer expectations clearly.

2. Reduces Development Errors

Proper analysis prevents misunderstandings.

3. Improves Software Quality

Well-defined requirements improve reliability.

4. Reduces Cost and Time

Early error detection reduces rework.

5. Better Communication

Improves coordination between customers and developers.

6. Proper Project Planning

Helps estimate cost, time, and resources.

7. Scope Management

Prevents unnecessary requirement changes.

Requirement Engineering

Requirement Engineering is the process of:

- Gathering
- Analyzing
- Specifying
- Validating
- Managing requirements

throughout the software lifecycle.

Requirement Engineering Tasks

---

1. Inception

Initial communication with customers takes place.

Objectives:

- Understand problem
- Identify stakeholders
- Define project goals

---

2. Elicitation

Requirements are collected from customers.

Techniques:

- Interviews
- Questionnaires
- Brainstorming
- Observation

---

3. Elaboration

Requirements are analyzed in detail.

Activities:

- Requirement modeling
- Scenario analysis
- Use case preparation

---

4. Negotiation

Conflicts between requirements are resolved.

Example:

Customer wants high security and low cost simultaneously.

---

5. Specification

Requirements are documented formally.

Usually prepared in:

SRS (Software Requirement Specification)

---

6. Validation

Requirements are checked for correctness.

Validation Questions:

- Are requirements complete?
- Are requirements correct?
- Are requirements feasible?

---

7. Requirement Management

Requirements are managed throughout the project.

Activities:

- Requirement tracking
- Change management
- Version control

Diagram of Requirement Engineering Process

```
Inception → Elicitation → Elaboration → Negotiation → Specification → Validation
→ Management
```

Advantages of Requirement Engineering

1. Improves software quality.
2. Reduces project failure.
3. Better customer satisfaction.
4. Improves communication.
5. Reduces maintenance cost.

Conclusion

Requirement Analysis and Requirement Engineering are critical activities that ensure successful software development.

Q4. Develop SRS for Hospital Management System.

Introduction

SRS stands for:

Software Requirement Specification

An SRS document describes the functional and non-functional requirements of a software system.

SRS for Hospital Management System

---

1. Introduction

Purpose

The purpose of Hospital Management System is to automate hospital activities such as:

- Patient registration
- Doctor management
- Billing
- Appointment scheduling
- Medical records management

Scope

The system helps:

- Doctors
- Patients
- Receptionists
- Hospital administration

It improves efficiency and reduces paperwork.

---

2. Functional Requirements

Patient Module

- Patient Registration
- Patient Login
- View Medical History
- Book Appointment

Doctor Module

- Doctor Login
- View Appointments
- Update Patient Records
- Prescribe Medicines

Reception Module

- Appointment Management
- Patient Record Management
- Billing Management

Admin Module

- Add/Remove Doctors
- Generate Reports
- Manage Database

---

3. Non-Functional Requirements

Security

- Password protection
- Data encryption

Performance

- Fast response time

Reliability

- 24×7 system availability

Maintainability

- Easy updates and maintenance

---

4. Hardware Requirements

- Processor: Intel i5 or above
- RAM: 8 GB
- Hard Disk: 500 GB

---

5. Software Requirements

- Operating System: Windows/Linux
- Database: MySQL
- Frontend: HTML/CSS/JS
- Backend: Java/Python/PHP

---

6. Advantages of Hospital Management System

1. Reduced paperwork.
2. Faster patient management.
3. Improved hospital efficiency.
4. Better record management.
5. Reduced human errors.

Conclusion

Hospital Management System automates hospital operations and improves healthcare management.

Q5. Prepare SRS for Railway Reservation System.

Introduction

Railway Reservation System is used to automate railway booking activities.

It allows users to:

- Book tickets
- Cancel tickets
- Check train availability

- View schedules

Functional Requirements

User Module

- Registration
- Login
- Search Trains
- Book Tickets
- Cancel Tickets

Admin Module

- Manage train schedules
- Update fares
- Generate reports

Non-Functional Requirements

- Security
- Reliability
- Availability
- Performance

Advantages

1. Fast ticket booking.
2. Reduced manual work.
3. Better passenger management.
4. Online accessibility.

Conclusion

Railway Reservation System improves railway management efficiency and passenger convenience.

Q6. Prepare SRS for University Management System.

Introduction

University Management System automates academic and administrative activities.

Functional Requirements

Student Module

- Student Registration
- Course Enrollment
- Result Viewing

Faculty Module

- Attendance Management
- Marks Upload
- Timetable Management

Admin Module

- Student Record Management
- Faculty Management
- Report Generation

Non-Functional Requirements

- Security
- Scalability
- Performance
- Reliability

Advantages

1. Centralized management.
2. Reduced paperwork.
3. Better academic management.
4. Improved communication.

Conclusion

University Management System simplifies academic administration and improves efficiency.

Q7. Explain Equivalence Partitioning and Boundary Value Analysis.

Introduction

Testing is important to ensure software quality.

Equivalence Partitioning and Boundary Value Analysis are Black Box Testing techniques.

Equivalence Partitioning (EP)

Equivalence Partitioning divides input data into valid and invalid classes.

Each class is tested using one representative value.

Example

Input Range: 1 to 1000

Valid Class

1–1000

Invalid Classes

- Less than 1
- Greater than 1000

Test Cases using EP

Test Case	Input	Expected Result
TC1	500	Valid
TC2	0	Invalid
TC3	1001	Invalid

Boundary Value Analysis (BVA)

BVA focuses on boundary values because errors usually occur at boundaries.

Boundary Values

For range 1 to 1000:

- 0
- 1
- 2
- 999
- 1000
- 1001

Test Cases using BVA

Test Case	Input	Expected Result
TC1	0	Invalid
TC2	1	Valid
TC3	2	Valid

Test Case	Input	Expected Result
TC4	999	Valid
TC5	1000	Valid
TC6	1001	Invalid

Advantages of EP and BVA

1. Reduces number of test cases.
2. Improves testing efficiency.
3. Detects boundary errors.
4. Saves testing time.

Conclusion

Equivalence Partitioning and Boundary Value Analysis are effective Black Box Testing techniques used for designing efficient test cases.

Q8. Cost Estimation Numerical Problem.

Problem

Assume:

- Project Size = 10,000 LOC
- Salary = Rs 25,000/month
- Organic Mode

Find:

1. Effort
2. Development Time
3. Total Cost

Step 1: Convert LOC into KLOC

```
KLOC = 10,000 / 1000
      = 10 KLOC
```

Step 2: Calculate Effort

```
Effort = 2.4 × (10)^1.05
       ≈ 26.9 Person-Months
```

Step 3: Calculate Development Time

```
Time = 2.5 × (26.9)^0.38
     ≈ 8.7 Months
```

Step 4: Calculate Total Cost

```
Cost = Effort × Salary
     = 26.9 × 25,000
     = Rs 6,72,500
```

Final Answer

Parameter	Value
Effort	26.9 Person-Months
Development Time	8.7 Months
Total Cost	Rs 6,72,500

Conclusion

COCOMO helps estimate software project effort, time, and development cost accurately.

END OF MODULE 2