


CS403 SOFTWARE ENGINEERING (RGPV)

UNIT-3 : SOFTWARE DESIGN

 **Exam Focus:** Unit-3 se UML Diagrams, Architectural Design aur Design Principles par direct questions aate hain. UML sabse important topic hai.


UNIT OVERVIEW

Why This Unit is Important?

Requirement ke baad software ka blueprint (design) banaya jata hai.

Jaise ghar banane se pehle map banaya jata hai, waise hi coding se pehle software design banaya jata hai.

Weightage

 Approx 12-18 Marks

Most Repeated Questions

 Explain Software Design Process

 Explain Design Principles

 Explain UML Diagrams

 Explain Architectural Styles

🔥 Explain Component Based Design

🔥 Explain SA/SD Methodology

1. SOFTWARE DESIGN PROCESS



Simple Explanation

Requirements ko coding ke liye blueprint me convert karna Software Design Process kehlata hai.

Definition

"Software design is the process of transforming requirements into a representation of software that can be implemented."

Design Process Steps

Requirements



Analysis



Design



Coding



Testing

Objectives

✅ Better understanding

✓ Reduce complexity

✓ Improve maintainability

Advantages

- Easy coding
 - Easy maintenance
 - Better quality
-

2. DESIGN CONCEPTS

Important Theory Question

1. Abstraction

Unnecessary details hide karna.

Example

Car chalane ke liye engine ka pura design nahi pata hona chahiye.

2. Modularity

System ko small modules me divide karna.

Example

Library System

- Login Module
- Book Module
- Student Module

3. Information Hiding

Internal details hidden rehte hain.

4. Functional Independence

Har module apna kaam independently kare.

5. Refinement

Step-by-step detailed design banana.

Memory Trick

AMIFR

A → Abstraction

M → Modularity

I → Information Hiding

F → Functional Independence

R → Refinement

3. DESIGN PRINCIPLES

Most Important

Principle 1

Design should be understandable.

Principle 2

Design should be modular.

Principle 3

Design should be reusable.

Principle 4

Design should be maintainable.

Principle 5

Design should minimize complexity.

Exam Definition

"Design principles are guidelines used to create effective, maintainable and reliable software designs."

4. UML (UNIFIED MODELING

LANGUAGE) ★★★★★

Most Important Topic

Every Year Expected

Definition

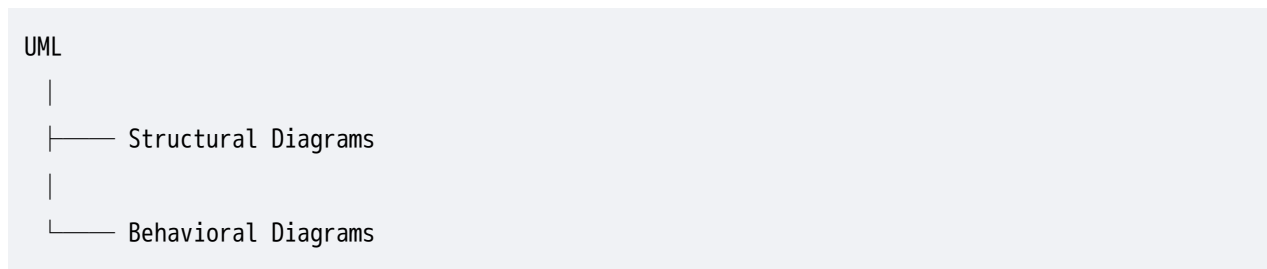
UML is a standard graphical language used to visualize, specify and document software systems.

Why UML?

- Better communication
 - Easy understanding
 - Visual representation
-

UML DIAGRAMS

UML diagrams mainly two categories:



STRUCTURAL UML DIAGRAMS

1. CLASS DIAGRAM

Most Important

Purpose

Shows classes, attributes and methods.

Example

```
+-----+
| Student |
+-----+
| Name    |
| RollNo  |
+-----+
| Login() |
| Register() |
+-----+
```

Components

- Class Name
 - Attributes
 - Methods
-

Exam Question

Draw and explain Class Diagram.

2. OBJECT DIAGRAM

Purpose

Shows real objects at runtime.

Example

```
Student1 : Student
```

3. COMPONENT DIAGRAM

Purpose

Shows software components.

Example

```
graph TD;
  A[Login Component] --- B[Database Component];
```

4. DEPLOYMENT DIAGRAM

Purpose

Shows hardware deployment.

Example

```
graph TD;
  A[Client PC] --- B[Web Server];
  B --- C[Database Server];
```

5. PACKAGE DIAGRAM

Purpose

Shows grouping of classes.

Example

Package: Student

Package: Faculty

BEHAVIORAL UML DIAGRAMS

6. USE CASE DIAGRAM

Already important from Unit-2

Purpose

Shows interaction between user and system.

Example

```
Customer
|
Withdraw Money
|
ATM System
```

7. ACTIVITY DIAGRAM

Frequently Asked

Purpose

Shows workflow of activities.

Example

```
graph TD; Start --> Login; Login --> Check_Password[Check Password]; Check_Password --> Success; Success --> End;
```

Start
↓
Login
↓
Check Password
↓
Success
↓
End

Similar To

Flowchart

8. SEQUENCE DIAGRAM

Very Important

Purpose

Shows sequence of messages.

Example

User → Login Page

Login Page → Database

Database → Login Page

Login Page → User

Key Point

Time sequence is important.

9. STATE CHART DIAGRAM

Purpose

Shows states of an object.

Example

Idle

↓

Processing

↓

Completed

10. COMMUNICATION DIAGRAM



Purpose

Shows interaction among objects.

Example

Customer ↔ ATM ↔ Bank Server

UML QUICK REVISION TABLE



Diagram	Purpose
Class	Structure of classes
Object	Runtime objects
Component	Software modules
Deployment	Hardware setup
Package	Grouping
Use Case	User interaction
Activity	Workflow
Sequence	Message sequence
State Chart	Object states
Communication	Object communication

5. ARCHITECTURAL DESIGN



Most Important

Definition

High-level structure of software system.

Example

Presentation Layer



Business Layer



Database Layer

Advantages

- Better organization
 - Easy maintenance
-

6. ARCHITECTURAL VIEWS

Logical View

System functionality.

Process View

Runtime processes.

Development View

Program modules.

Physical View

Hardware deployment.

Memory Trick

LPDP

Logical

Process

Development

Physical

7. ARCHITECTURAL STYLES



Frequently Asked

1. Layered Architecture

Presentation Layer



Business Layer

↓
Database Layer

Example:

College ERP

2. Client-Server Architecture

Client
↓
Server

Example:

Web Applications

3. Data-Centered Architecture

Users
↓
Database

Example:

Banking Systems

4. Pipe and Filter Architecture

Input
↓
Filter 1
↓

Filter 2



Output

Example:

Compiler

8. USER INTERFACE DESIGN

Definition

Design of screens and user interaction.

Goals

- Easy to use
 - Attractive
 - Consistent
-

Good UI Characteristics

- Simplicity
 - Consistency
 - Feedback
 - Error Prevention
-

9. FUNCTION ORIENTED DESIGN



Definition

Design based on functions performed by system.

Example

ATM

```
ATM
↓
Withdraw
↓
Deposit
↓
Balance Inquiry
```

Tools

- DFD
 - Structure Chart
-

10. SA/SD (STRUCTURED ANALYSIS / STRUCTURED DESIGN) ★★★★★

Frequently Asked

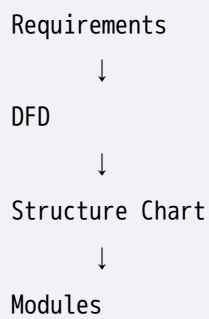
Structured Analysis

Requirement analysis using DFD.

Structured Design

Converting DFD into modules.

Diagram



Advantages

- Easy understanding
 - Modular design
-

11. COMPONENT BASED DESIGN



Important Topic

Definition

Design using reusable software components.

Example

E-Commerce Website

Login Component

Payment Component

Cart Component

Notification Component

Advantages

- ✓ Reusability
 - ✓ Faster Development
 - ✓ Reduced Cost
-

Disadvantages

- ✗ Compatibility Issues
 - ✗ Dependency Problems
-

12. DESIGN METRICS

Definition

Measures used to evaluate software design quality.

Examples

Size Metrics

Number of modules.

Complexity Metrics

Cyclomatic Complexity.

Coupling Metrics

Dependency between modules.

Cohesion Metrics

Internal strength of module.

Cohesion vs Coupling

Cohesion	Coupling
Within Module	Between Modules
Should be High	Should be Low
Good Design	Bad if High

 Frequently Asked

UNIT-3 IMPORTANT QUESTIONS

2 Marks

1. Define UML.
 2. What is Architectural Design?
 3. What is SA/SD?
 4. Define Design Metrics.
-

5 Marks

1. Explain Design Concepts.
 2. Explain Design Principles.
 3. Explain Component Based Design.
 4. Explain Architectural Views.
-

7 Marks

1. Explain UML Diagrams.
 2. Explain Architectural Styles.
 3. Explain SA/SD Methodology.
 4. Explain Function Oriented Design.
-

10 Marks

1. Explain Software Design Process.
 2. Explain UML with suitable diagrams.
 3. Explain Architectural Design and Architectural Styles.
 4. Explain Component Based Design with advantages and disadvantages.
-

PYQ ANALYSIS & 2026 PREDICTION

Topic	Trend	Probability
UML Diagrams	Frequently Asked	Very High
Architectural Styles	Frequently Asked	Very High
Design Principles	Frequently Asked	High
Component Based Design	Frequently Asked	High
SA/SD	Sometimes Asked	High
Design Metrics	Sometimes Asked	Medium

LAST NIGHT REVISION SHEET

- ✓ UML = Visual Language
- ✓ Class Diagram = Classes
- ✓ Use Case Diagram = User Interaction
- ✓ Activity Diagram = Workflow
- ✓ Sequence Diagram = Message Flow
- ✓ Architectural Design = High-Level Structure
- ✓ Layered Architecture = Presentation → Business → Database
- ✓ Component Based Design = Reusable Components
- ✓ Cohesion = High
- ✓ Coupling = Low