OBJECT ORIENTED SYSTEM (OOS)

Complete Notes – Units 1 to 7

(Prepared for M.Sc. Information System Engineering – Final Exam)

UNIT 1

---

1. Complex Systems: Definition and Attributes

A complex system consists of many interacting components whose overall behavior emerges from these interactions. Attributes include hierarchy, emergence, interdependency, nonlinearity, adaptability, uncertainty, and self-organization.

2. Software Complexity Justification

Software systems are inherently complex due to logical complexity, changing requirements, concurrency, integration issues, security needs, and large state spaces.

3. Need of Object-Oriented Systems

OOS reduces complexity through abstraction, modularity, encapsulation, inheritance, and reusability.

4. Object-Oriented vs Traditional SDLC

Traditional is function-based, sequential; OOSD is iterative, object-based, modular, and maintains high reusability.

UNIT 2

---

1. Characteristics of OOD

Abstraction, encapsulation, modularity, hierarchy, polymorphism, reusability, low coupling, high cohesion.

2. OOA & OOD as Parallel

Due to iterative refinement, shared models, and continuous validation.

3. SOLID Principles

SRP, OCP, LSP, ISP, DIP.

4. Class & Object + Relationships

Class: blueprint. Object: instance. Relationships: association, aggregation, composition, inheritance, dependency, realization.

UNIT 3

---

1. UML & Significance

Standard modeling language for visualization, design, communication, documentation.

2. Class Diagram Example

Shows classes, attributes, methods, and relationships.

3. Use Case & Class Diagram

Includes actors, use cases, and static class structure.

4. Sequence Diagram Example

Models time-ordered message interactions (e.g., ATM withdrawal).

UNIT 4

---

1. Domain Analysis & Stages

Stages: scoping, data collection, concept identification, modeling, classification, domain dictionary, validation.

2. Class-Based Modeling

Identify classes, attributes, methods, relationships; draw class diagrams.

3. Scenario-Based Modeling

Use cases, activity diagrams, event modeling, sequence diagrams.

4. Behavioral Modeling

State machines, transitions, events, workflows.

5. Activities of Domain Analysis

Scoping → modeling → scenario analysis → behavior analysis → validation → documentation.

UNIT 5

---

1. Definition & Role of Agent

Agents are autonomous, reactive, proactive entities improving performance via distribution, adaptability, resource management.

2. Agent Properties

Autonomy, reactivity, proactiveness, social ability, adaptability, mobility, goal-driven behavior.

3. OOP vs AOP

OOP: objects, encapsulation.

AOP: agents, autonomy, communication, reasoning.

4. Agent Class Representation

Use «agent» stereotype; include beliefs, goals, plans; show interactions with environment.

UNIT 6

---

1. Internal vs External Quality

Internal: design, modularity, maintainability.

External: usability, reliability, performance, security.

2. Software Metric Example

Cyclomatic complexity = E – N + 2.

3. OO Metrics

CK metrics: WMC, DIT, NOC, CBO, RFC, LCOM.

4. Candidate Solution Matrix

Comparison table for selecting best programming language.

UNIT 7

---

1. Methodology Role

Ensures structured development, quality, risk management.

2. Test Case Example

Login test case with inputs and expected outputs.

3. Prototyping Types

Throw-away, evolutionary, incremental, horizontal, vertical.

4. Evolutionary vs Throw-Away

Evolutionary grows into system; throw-away discarded.

5. Verification vs Validation + Testing Strategies

Verification: process correctness.

Validation: product correctness.

Strategies: unit, integration, system, acceptance, regression, stress, smoke.

END OF NOTES
