What is OOAD & Why OOAD?

What is OOAD?

Object-oriented analysis and design (OOAD) in the process of creating software helps us look at and plan a system using ideas about "objects" - like parts of the software that represent things in the real world. In OOAD, we first check out the whole system we need for a certain issue, then find and name the "objects" in that system. After that, we figure out how these objects are connected and make a plan for the whole system.

Why OOAD?

The object-oriented analysis and design process is highly well-known. The following are a few reasons why it is quite renowned among the community:

1. *It's extremely easy to understand, which helps in creating models of complex problems.*
2. *Concepts such as inheritance help make the data reusable and scalable.*
3. *It's easily maintainable, which helps identify the issues in the early processes and saves time.*

In short, Easy to understand and communicate between a technical and non-technical person in an organization about a project.

Object-Oriented Analysis (OOA):

Objective:
The goal of analysis is to understand and define the problem domain, identify the system's requirements, and create a conceptual model of the system.

Activities:
Requirement Gathering: *Gather information about the system's requirements from stakeholders, users, and existing documentation.*

Problem Definition: *Clearly define the problem that the system is intended to solve.*

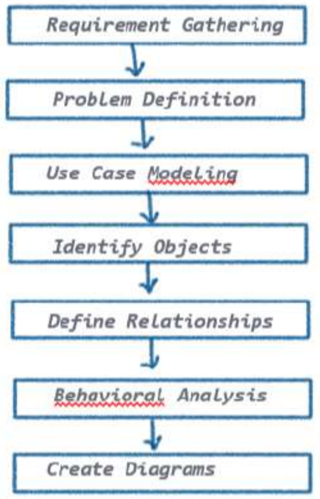
Use Case Modeling: *Identify and define use cases, which represent the interactions between the system and external entities (users or other systems).*

Identify Objects: *Identify and model the key entities or objects in the problem domain. Objects encapsulate both data and behavior.*

Define Relationships: *Establish relationships between objects, representing how they interact and collaborate.*

Behavioral Analysis: *Specify the dynamic behavior of the system by defining scenarios and state transitions.*

Create Diagrams: *Use tools like Unified Modeling Language (UML) to create diagrams (class diagrams, use case diagrams, etc.) that visually represent the system's structure and behavior.*



Object-Oriented Design (OOD):

Objective:
The goal of design is to transform the conceptual model created during analysis into a detailed and implementable plan for the software system.

Activities:

Architectural Design: *Define the overall architecture of the system, including high-level structures and components.*

Detailed Design: *Elaborate on the architecture by specifying the details of each class, defining their attributes, methods, and relationships.*

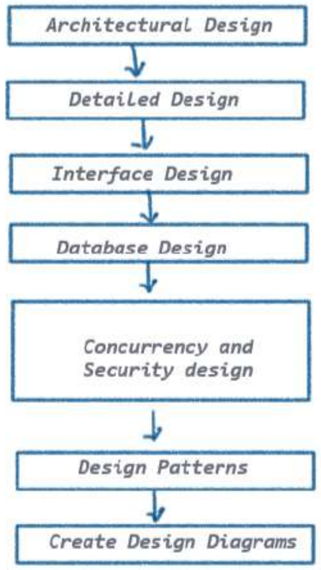
Interface Design: *Design the interfaces (methods and data) that allow objects to interact with each other.*

Database Design: *If applicable, design the database schema and define how objects will be persisted.*

Concurrency and Security Design: *Address issues related to concurrent execution, data integrity, and system security.*

Design Patterns: *Apply design patterns to solve common design problems and promote reusability.*

Create Design Diagrams: *Use UML or other design notations to create diagrams that document the detailed design, such as class diagrams, sequence diagrams, and component diagrams.*


Architectural Design
Detailed Design
Interface Design
Database Design
Concurrency and Security design
Design Patterns
Create Design Diagrams