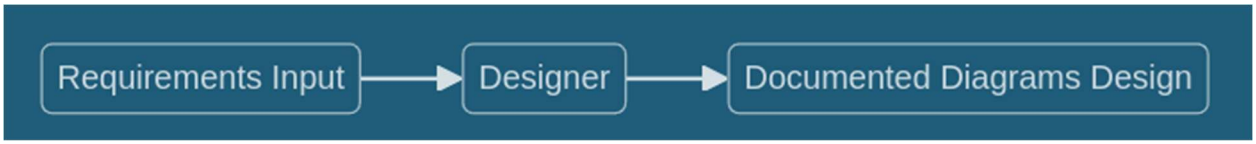
Object Oriented Analysis & Design (OOAD)

Object-Oriented Analysis and Design (OOAD) is a way to design software by thinking of everything as **objects like real-life things**. In OOAD, we first understand what the system needs to do, then identify key objects, and finally decide how these objects will work together. This approach helps make software easier to manage, reuse, and grow. **The purpose of Object-Oriented Analysis and Design (OOAD) is to convert textual requirements into software design and architecture.**

What is Object-Oriented Analysis and Design (OOAD)?

OOAD is **based on the concepts of object-oriented programming (OOP)** and is an organized and systematic approach to designing and developing software systems. It is a software engineering paradigm that integrates two distinct but closely related processes: Object-Oriented Analysis (OOA) and Object-Oriented Design (OOD). The typical steps in OOAD can be summarized as follows:



Important Aspects of OOAD

Below are some important aspects of OOAD:

- **Object-Oriented Programming:** In this the real-world items are represented/mapped as **software objects** with attributes and methods that relate to their actions.
- **Design Patterns:** Design patterns are used by OOAD to help developers in **building software systems** that are more efficient and maintainable.
- **UML Diagrams:** UML diagrams are used in OOAD to **represent the different components and interactions of a software system**.
- **Use Cases:** OOAD uses use cases to help **developers understand the requirements of a system** and to design software systems that meet those requirements.

Object-Oriented Analysis

Object-Oriented Analysis (OOA) is the **process of understanding and analyzing the system requirements** by looking at the problem scenario in terms of objects.

- These objects **represent real-world entities** or concepts that are relevant to the system being developed.
- During OOA, **the goal is to identify the objects**, their attributes, behaviors, and relationships, **without focusing on how the system will be implemented**.

Can summaries the Object-Oriented Analysis (OOA) as follows:

- process of analyzing a task (also known as a problem domain)
- finding and describing objects
- typical:
  - set of use cases
  - one or more class diagrams
  - a number of interaction diagrams

Object-Oriented Design (OOD)

In the **object-oriented software development process**, the **analysis model**, which is **initially formed through object-oriented analysis (OOA)**, undergoes a transformation during object-oriented design (OOD) i.e. **implementation of the conceptual model developed in OOA**. This evolution is crucial because it shapes the analysis model into a detailed design model.

Furthermore, as part of the **object-oriented design process**, it is essential to define specific aspects:

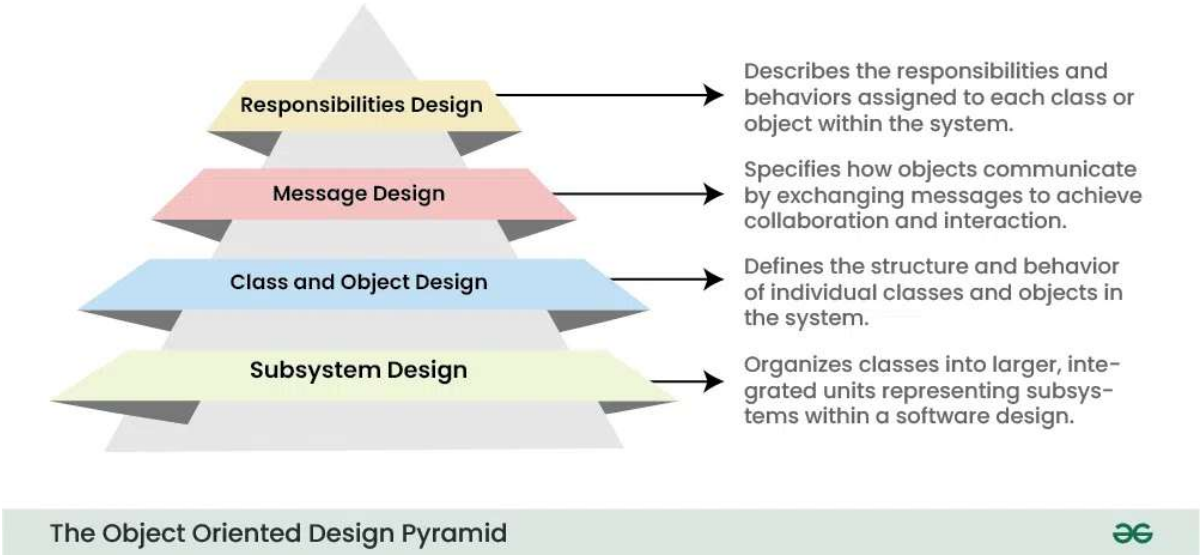
- **Data Organization of Attributes:**
  - OOD involves **specifying how data attributes** are organized within the objects.
  - This includes determining the **types of data each object** will hold and how they relate to one another.
- **Procedural Description of Operations:**

- OOD requires a **procedural description for each operation** that an object can perform.
- This involves detailing the steps or **processes involved in carrying out specific tasks.**

Can summaries the Object-Oriented Analysis (OOD) as follows:

- **Defining software objects** and how they collaborate to fulfill the requirements.
- **Constraints to conceptual model** produced in object-oriented analysis
- **Concepts in the analysis model are mapped onto implementation** classes and interfaces resulting in a model of the solution domain.

Below diagram shows a design pyramid for object-oriented systems. It has the following four layers.



The Object Oriented Design Pyramid

1. **The Subsystem Layer:** It represents the **subsystem that enables software to achieve user requirements** and implement technical frameworks that meet user needs.

2. **The Class and Object Layer:** It represents the **class hierarchies** that enable the system to develop using generalization and specialization. **This layer also represents each object.**
3. **Message Layer:** This layer **deals with how objects interact with each other**. It includes **messages sent between objects**, **method calls**, and the flow of control within the system.
4. **The Responsibilities Layer:** It focuses on the **responsibilities of individual objects**. This includes **defining the behavior of each class**, specifying what each object is responsible for, and how it responds to messages.

Benefits of Object-Oriented Analysis and Design (OOAD)

- It increases the **modularity and maintainability of software** by encouraging the creation of tiny, **reusable parts** that can be combined to create more complex systems.
- It provides a **high-level, abstract representation** of a software system, making understanding and maintenance easier.
- It **promotes object-oriented design principles and the reuse of objects**, which lowers the amount of code that must be produced and raises the quality of the program.
- Software engineers can **use the same language and method that OOAD provides to communicate** and work together more successfully in groups.
- It can **assist developers in creating scalable software systems** that can adapt to changing user needs and business demands over time.

Real world applications of Object-Oriented Analysis and Design (OOAD)

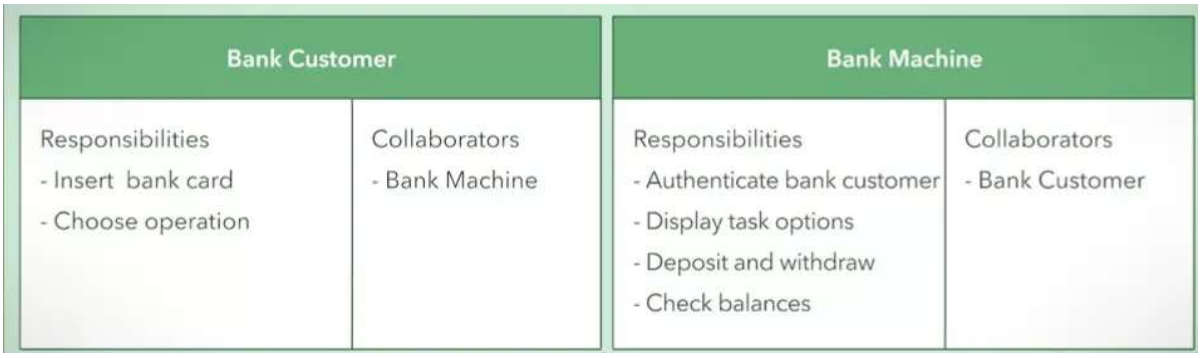
Some examples of OOAD's practical uses are listed below:

- **Banking Software:** In banking systems, OOAD is frequently used to **simulate complex financial transactions**, structures, and customer interactions. Designing adaptable and reliable financial apps is made easier by OOAD's modular and scalable architecture.

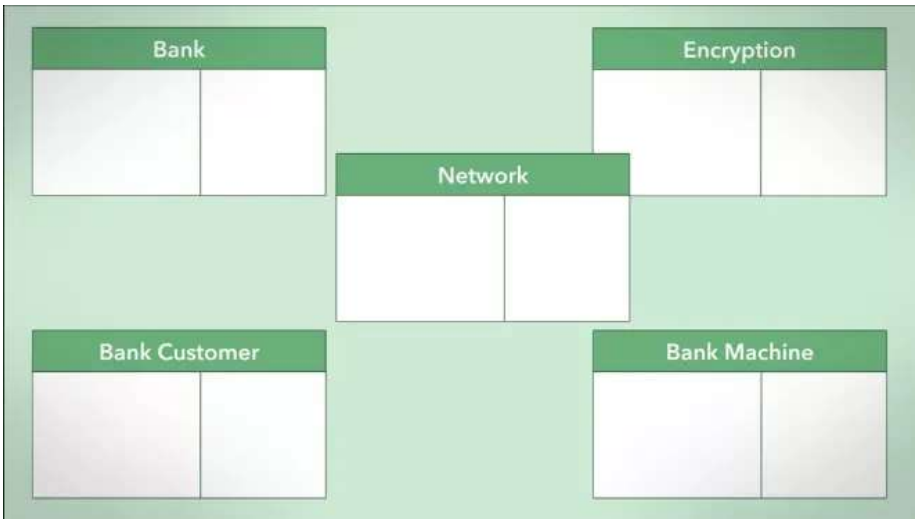
- **Electronic Health Record (EHR) Systems: Patient data, medical records, and healthcare workflows are all modeled using OOAD**. Modular and flexible healthcare apps that may change to meet emerging requirements can be made through object-oriented principles.
- **Flight Control Systems:** OOAD is crucial in designing flight control systems for aircraft. It helps model the **interactions between different components such as navigation systems, sensors, and control surfaces**, ensuring safety and reliability.
- **Telecom Billing Systems:** In the telecom sector, OOAD is **used to model and build billing systems**. It enables the modular and scalable modeling of complex subscription plans, invoicing rules, and client data.
- **Online Shopping Platforms: E-commerce system development frequently makes use of OOAD**. Product catalogs, user profiles, shopping carts, and payment procedures are all modeled, which facilitates platform maintenance and functionality expansion.

This technique is the use of **Class, Responsibility, Collaborator (CRC) cards**. CRC cards help record and organize components into classes, identify component responsibilities, and determine how they collaborate with each other. Therefore, they also help refine the components in your software design.

CRC Card Architecture



Relationship between Bank Customer and Bank Machine CRC



Relationship between multiple CRC cards in banking system