

Unit III

Object Oriented Analysis Process, use case Driven Object oriented Analysis, use Case Model, object Analysis: classification, classification Theory, Approaches for identifying classes, Responsibilities and collaborators, identifying obj. Relationships, Attributes & methods: Associations, Super-Sub class relationships, A Part of Relationships - Aggregation, class Responsibilities, obj. Responsibilities.

Object Oriented Analysis:- → Determined the system requirements, & recognized the classes & relationship b/w the classes.
→ Main purpose of OOA is the recognize appln domain & specific requirements of the System.

- Three techniques of OOA:-
- ① Object modeling
 - ② Dynamic "
 - ③ functional "

Analysis:- is the process of transforming a problem definition from a fuzzy set of facts & myths into a coherent statement of a system's requirements.

④ Obj. Modeling:- (definition back)

The process of Obj. Modeling:-

1. Recognize obj. & grouped into classes

2. relationship b/w the classes

3. User obj. model diag. is generated

4. Define attributes of user obj.

5. Define the operation need to perform

④ * Dynamic Modeling: → Explain how single obj. responds to events.

→ Main aim: Examine the behavior of the obj. regarding time & external change.

Process of Dynamic Modeling :-

1. State of every obj. is recognized

2. Recognize the event

3. Generate dynamic model diag. enclose with state transition diag.

4. Communicate with every state regarding obj. attribute

5. Verify state transition diag.

⑤ * Functional Modeling: — last component of the OOA.
→ It shows the processes executed in an obj. & how the data change when it moves b/w the methods.

Process of func. Modeling:

1. All I/P & O/P are recognized

2. Create a data flow diag. to show functional dependencies

3. Identify the motive of every fun.

4. Identify the constraints

5. Describe the optimization criteria

OOA Process :-

1. Identify the actors:

→ Who is using the system?

→ Or, in the case of a new system, who will be using system?

2. Develop a simple business process model using UML activity diagram.

3. Develop the use case:

→ What the users are doing with the system?

→ Or, in the case of a new system, what users will be doing with the system?

4. Prepare interaction diagrams:

→ Determine the seq.

→ Develop collaboration diag.

5. Classification - develop a static UML class diagram:

→ Identify classes

→ " relationships

→ " attributes

→ " methods

6. Generate & refine: If needed, repeat the preceding steps.

Use-Case Model:— are scenarios for understanding system requirements.

→ It describes the uses of the system & shows the courses of events that can be performed.

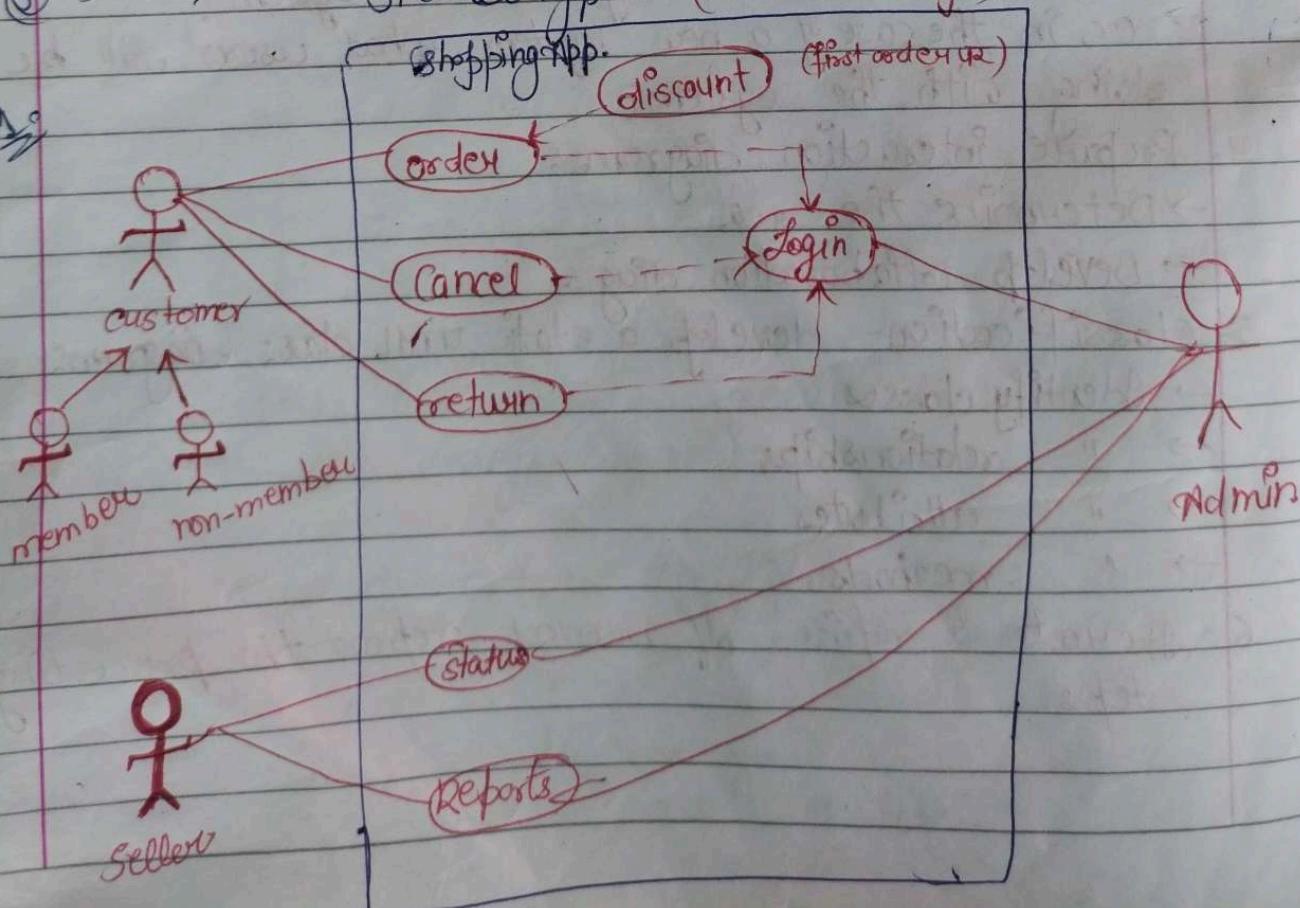
use-case diagram:

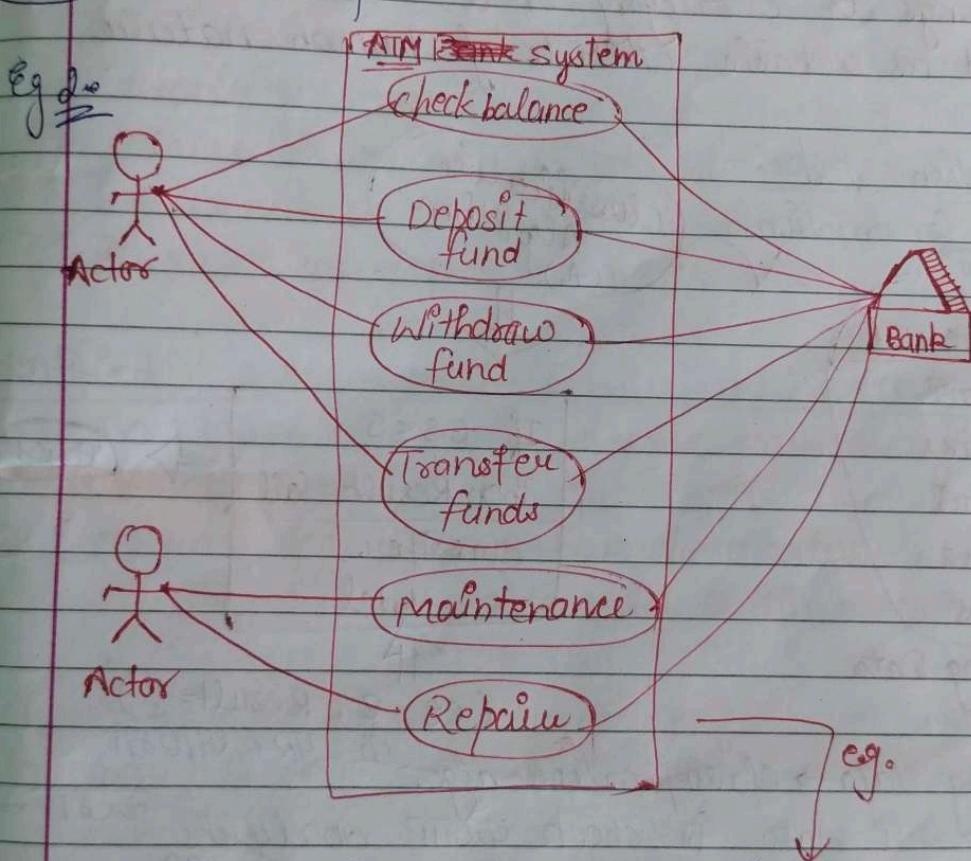
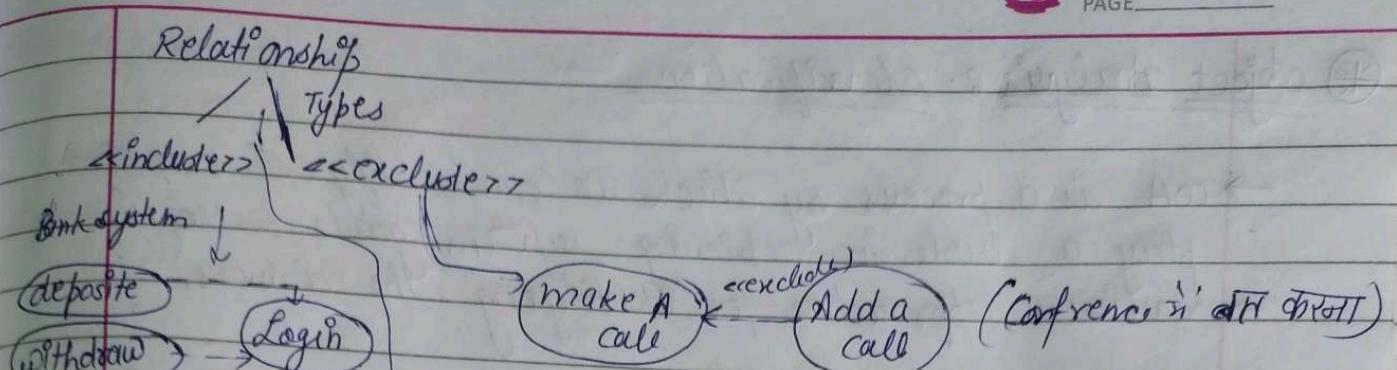
① Component → Actor - ♂ → play role in a system
 (i) could be a real person or a group system or a group of persons.

② → use-case
 capability
 — → connector → draw b/w the user & the use-case (for interaction)

④ → → Generalisation (user को divide करना)

⑤ - - - → - stereotype - (relationship)





* Use-Case Driven Approach :- following steps :-

- Identify the actor.
- processes of the system
- Develop the use case
- Prepare interaction program.
 - Determine the sq.
 - Develop the collaboration diag.
- Develop a UML class diagram.
- Iterate & refine.

* Object Analysis : Classification :-

- OOA is a process by which we can identify classes that play a role in achieving system goals & requirements.
- Various approaches for identifying the classes.

Classification :- Is the process of checking to see if an obj. belongs to a category or a class, is regarded as a basic attribute of human nature.

How classification works :-

(i) Model Construction.

~~Classification~~

Classification
Algorithm
(C.A.)

CS	Result
5	Fail
6	Pass
2	Fail
7	Pass
9	Pass

Training Data

If $CS \leq 5$
Then Result = Fail
classifier
Model

→ Pass

($CS = 8$, Result = ?)

(Training data → Classification algo.)

C.A. → उस data को check करता analysis करता →
determine करता, every line को execute करता →
or one line state produce करता तो Training
data का satisfy करता)

(ii) Model usage → This model works on the fact

($CS = 8$, Result = ?) that our model
is working

properly or not

$8 \leq 5 \Rightarrow \text{False}$

⇒ so else (Pass)

(*) Approaches for Identifying classes :-

1. Name Phrase Approach
2. Use-Case driven "
3. Common class "
4. Class Responsibilities & Collaborators (CRC)

(*) Name Phrase approach :- using the method, you have to read through the use cases, interviews, & requirements. Specification carefully, looking for noun phrases.

- Examining the use-case / requirements.
- Nouns in textual forms are selected & considered to be the classes.
- Plural classes are converted in to a single classes.

(*) Identified classes are grouped into 3 categories :-

- (a) Irrelevant classes → they are unnecessary classes.
- (b) Relevant " → " the necessary "
- (c) Fuzzy " → they are the classes where exist some uncertainty in their existence.

I.R. F.U.Z.Z.Y. R.

(*) Common class pattern Approach :- A set of classes that are common for all domains are listed & classes are identified based on the category. The set of class category is listed based on the previous knowledge (past experience)

The class patterns are :- → Concept Class that represent the whole business activity e.g:- Savings Bank class

represent a class person, group of person, resources & facilities. e.g.- CSEDEPT.

→ Event " " represent some event at a particular instance of time. (e.g:- Transaction class)

represent the individuals who play some role in the system (in any association)

Place " Tangible things & Device "



→ Place class → represent physical locations which is needed to record some details or the place itself is recorded in detail.

→ Tangible things & Device class → includes tangible objects & devices like sensors involved in the system.

→ CRC is technique used for identifying classes, responsibilities & therefore their attributes & methods → CRC helps us identify classes

(*) classes, collaborators, Responsibilities Approach

→ Classes represents group of similar objects.

→ Responsibilities represent the attributes & methods (responsibility of the class)

→ Collaborators represent other objects whose interaction is needed to fulfill the responsibilities of the class/ object.

(*) Use-Case Driven Approach: → To identify objects of a system & their behaviors, the lowest level of executable use cases is further analyzed with a seq. & collaboration diagram pair.

→ By walking through the steps, you can determine what objects are necessary for the steps to take place.

e.g.: Previous one

(*) Collaboration CRC Cards

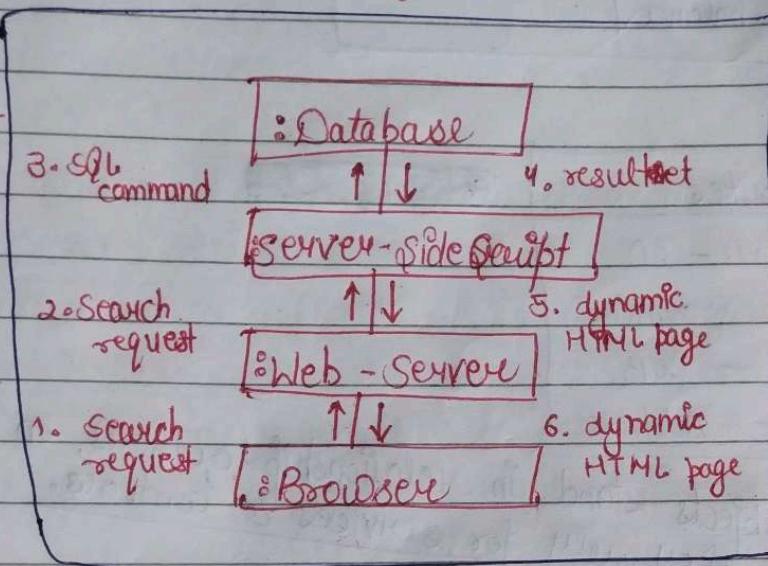
(*) Collaboration Diagram: → is a name given to the interaction among two or more classes/objects.

→ It shows:

(*) objects & their links to each other, as well as how messages are sent b/w the linked objects.

- Collaboration shows the implementation of an operation or the realization of a use case.
- The focus here is on msg. → obj. roles instead of time

c. Diagram :-

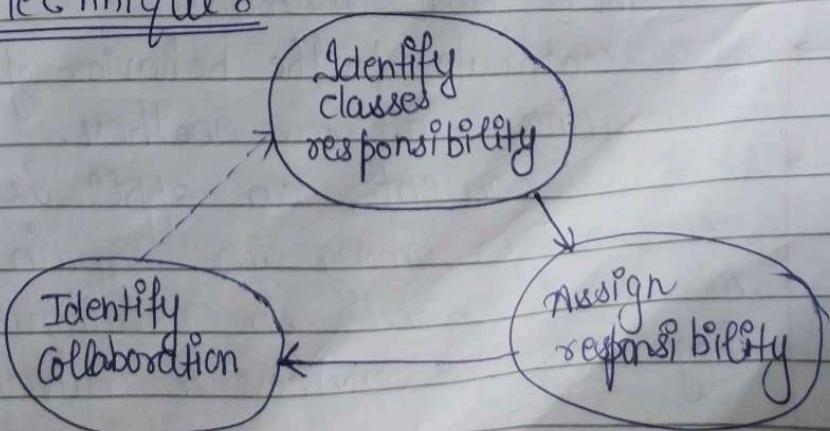


- It is useful when we want to refer to a particular interaction.
- To illustrate the coordination of obj. structure & flow of control.

④ CRC Card :- → Stands on class, Responsibilities & Collaborators developed by Cunningham, Wilkerson & Beck.

- CRC can be used for identifying classes & their responsibilities.

Process of CRC Technique :-



one-side
Real side
front side
Responsibilities

CUSTOMER	collaborations
	withdraw cash
	deposit cash
	check balance

other-side
Real side
front side

Attributes	
customer - ID	
" - name	
" - DOB	
" - Address	

All objects stand in relationship to others, on whom they rely for services & controls.

Identifying Object Relationships, Attributes, & Methods

Goals:-

- Analyzing relationships among classes.
- Identifying associations.
- Association patterns
- Identifying super- and subclass hierarchies.

Introduction:-

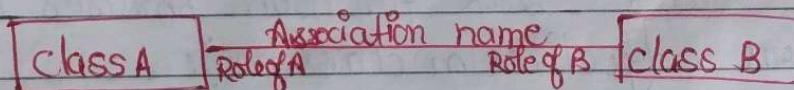
- Identifying aggregation or a-part-of compositions.
- class responsibilities.
- Identifying attributes & methods by analyzing use-cases and other UML diagrams.
- obj.'s contribute to the behavior of the system by collaborating with one another.
- In oo environment, an appln is the interactions & relationships among its domain objects.
- All objects stand in relationship to others, on whom they rely for services & controls.

Objects Relationships :-

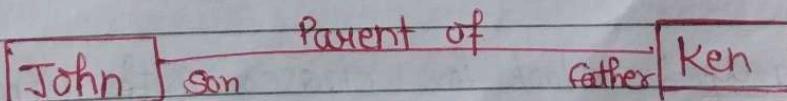
Three types of relationships among objects are :-

- * Association
- * Super-Sub structure (also known as generalization hierarchy)
- * Aggregation & a-part-of structure.

- * Associations :- → A reference from one class to another is an association.
- Basically a dependency b/w two or more classes is an association.
- For eg. :- Jackie works for John.
- It is physical or conceptual connection b/w two or more objects.
- It is shown as a class symbol connected by solid line along with the association rules.



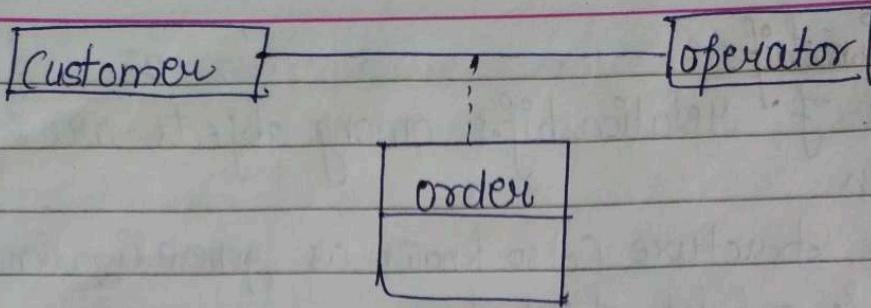
- ⇒ The role name should be closer to the each class describe this role.



Common Association Patterns :- It includes :-

- (i) Location Association :- next To, part of, contained in, ingredient of, etc :-
For eg:- cheddar cheese is an ingredient of the French soup.

- (ii) Communication Association :- talk to, order to.
For eg:- a customer places an order with an operator person.



- ④ Superclass - Subclass Relationships : → This relationship represents the inheritance relationships b/w classes.
- These relationships are also called as Generalization hierarchy ; allow objects to be built from other objects.
- ~~such relat~~ Sub - classes are more specialized versions of their super- classes.

Guidelines for Identifying Super-Sub Relationships :-

- Top-Down :- Look for noun phrases composed of various adjectives or class name.
Eg:- Military Aircraft and Civilian Aircraft.
Only specialize when the Sub classes have significant behavior.
- Bottom-up :- ★ look for classes with similar attributes or methods. ★ Group them by moving the common attributes & methods to super class.
★ Do not force classes to fit a preconceived generalization structure.
- Reusability :- → Move attributes & methods as high as possible in the hierarchy.
→ At the same time do not create very specialized classes at the top of hierarchy.

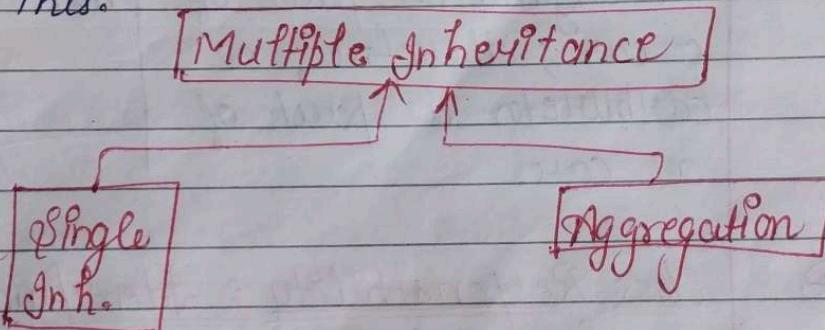


DATE _____

PAGE _____

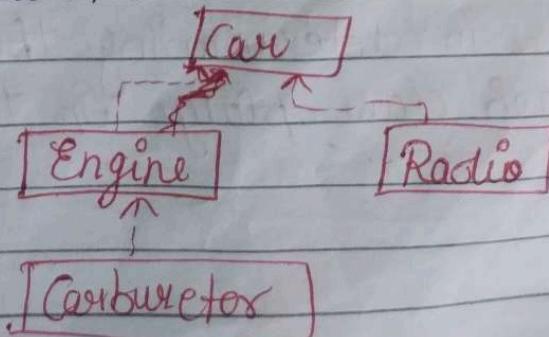
⇒ This balancing act can be achieved through several iterations.

- (iv) Multiple Inheritance :-
- ① Avoid ^{excessive} multiple use of M.I.
 - ② It is also more difficult to understand programs written in M.I. system.
 - ③ One way to achieve the benefits of M.I. is to inherit from the most appropriate class & add an obj. of other class as an attribute.
 - ④ M.I. can be represented as an aggregation of a single inheritance & Aggregation. This meta model reflects this.



A-part of Relationship - Aggregation :-

- ⇒ A-part-of relationship also called aggregation, represents the situation where a class consists of several component classes.
- ⇒ This doesn't mean that the class behaves like its parts.
- ⇒ For eg. A car consists of many other classes, one of them is a radio, but a car doesn't behave like a radio.



⇒ Two major properties of a-part-of-relationship are:

(*) Transitivity

(*) Antisymmetry.

Transitivity

If A is part of B & B is part of C, then A is part of C.

For eg. → a carburetor is part of an engine & an engine is part of a car; therefore, a carburetor is part of a car.

Antisymmetry

If A is part of B, then B is not part of A.

For eg. An engine is part of a car, but a car is not part of an engine.



Class Responsibility : Identifying Attributes & Methods :-

⇒ Identify attributes & methods, like finding classes is a difficult activity.

⇒ The use cases & other UML diagrams will be our guide for identifying attributes, methods & relationships among classes.

⇒ Attributes can be identified by analyzing the use cases, seq. / collab. boration, activity & state diagrams.

★ Assign each responsibility to the class that it logically belongs to.

★ This also aids us in determining the purpose & the role that each class plays in the appn.

④ Object Responsibility & Attributes

→ Information that the system needs to remember.

⑤ Object Resp. & Methods & Messg.

Methods & messages are the work horses of object-oriented systems.

→ In O-O environment, every piece of data, or object, is surrounded by a rich set of routines called methods.

→ Seq. diagrams can assist us in defining the services the objects must provide.

⑥ Classification Theory

→ classification is process of checking to see if an obj. belongs to a category or class.

→ classes are imp. mechanism for classifying objects
chief role of class is to define attributes methods,
applicability of its instances.

→ class is specification of structure, behavior &
description of an obj.

→ classification concerned more with identifying
class of an obj. than individual objects within
a system.