UNIT-III

Q1] Define event and explain in brief different types of event.

→ 1) Signal Event

- A signal event represent the asynchronous communication between objects.
- It involves sending or receiving a signal.

2) Call Event

- A call event occurs when an operation is invoked or called
- It is a synchronous event which means the sender waits for response.

3] Time. Event

- Time Event is triggered when a specific time is reached or after a time interval.
- Can be defined using keywords like after(time) or at(time).

4] ~~Change~~ Change Event

- A change event occurs when a certain condition becomes true.
- It depends on logical expressions or state of variable

- A event is something that happens at a particular time in a system & may cause change in the state of an object.
- In UML, an event is used in ~~state~~ diagram. to show how and when an object moves from one state to another.

Q2] In a content of a state diagram, what are concurrent substates? Elaborate with an examples.

→
- Concurrent Substates are also called as orthogonal Substates
- Concurrent Substates are the multiple states that run at the same time inside a main diagram.
- These Substates run independently on each other that means they do not affect each other directly.
- They help in modelling real-world systems where many things happen at the same time.

Key points

1) Shown using dashed-line dividing it into region.
2) Each region have its own initial & final State.
3) All Concurrent states start when the main state becomes active.
4) States are independent of each other.
5) Parallel Execution of all substates.

- When both the substate & main state reaches final state, control from two region joins back into one flow

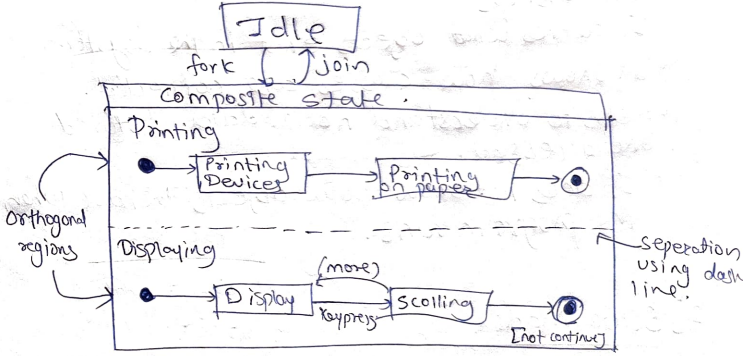


fig → Concurrent Substates.

Q] Write a short note on: Relation of class model & state model

→ 1] Class Model

- Class model shows the static structure of the system
- It includes classes, attributes, operations & relationships
- It defines what objects exist in the system
- It shows how classes are connected.
- Helps to understand how data is stored and accessed.
- It does not show how objects behave when the program runs.

2] State Model

- State model shows dynamic behaviour of the system
- It explains how objects changes the state after execution
- It is made using state diagram.
- States are connected with transition which are triggered by events.

Relation between class model & state model. →

1] The class model defines the object & the state model defines how that object behaves

2] The method or operations in the class model are used as events in the state model.

3] Whenever operation happens in class model the changes happens in state model.

4] Both model are required to give complete understanding of an ~~object~~ system → structure & behaviour.

Q.] Explain the difference between sequenced Collaboration diagram with example.

→

~~Feature~~ Feature.	Sequence Diagram	Collaboration Diagram
1] main focus	Time & order of message	which object talk to each other not timing
2] Time flow	Time moves top to bottom	No time is shown
3] Layout style	Vertical layout	Graph Layout
4] Object Representation	Object with lifelines	Objects shown as boxes only
5] Numbering	No need for message numbers,	Message no. are required to show which message come first
6] Return arrows	Can show return arrows	Return messages are not shown
7] Best use case	when we want to show step by step process.	When we want to show object relationship or communication

Q.] Explain the following by drawing neat fragment on a state diagram

i] Entry and exit actions of a states. ii] History states iii] Nested States iv] Activities.

→ i] Entry & exit actions of a State.

- The setup actions that need to be performed ~~before~~ on entry of particular state is called entry effects.
- Similarly while leaving ~~le~~existing exiting the states there are some cleanup actions that need to be performed ~~while~~ is called as exiting action/effect.

These are written as

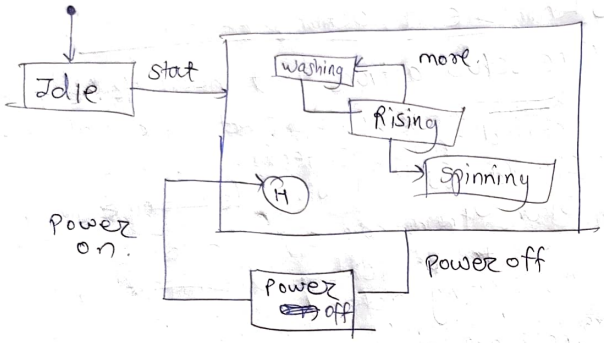
- entry/action
- exit/action.

ii]

ii] History State

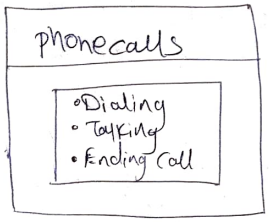
- History is used to remember previous state of state machine when it was interrupted.
- Ex→ Consider the state machine for washing machine. The washing machine washes, rinses & spins. But suppose if power is cut then the machine stops and again when

power gets on the machine start working from that state only refers as History state.



iii] Nested states (Substates).

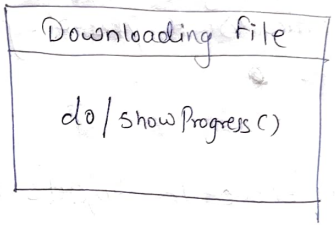
- Nested States is also called as composite state.
- It helps to organize ~~st~~ complex & larger states.



iv] Activities

→ This refers to "do-activities"

- When objects are in some state then they perform some activities this are called do-activities.
- The activities continue until it is interrupted or completed
- This ~~activ~~ activities are ~~not~~ called with "do"



Q] Explain state Diagram? Draw & discuss notation used for state machine, Triggers & ports, Transitions & conditions, Initials and final state

→

- A state Diagram shows how an object or system changes from one state to another based on events.
- It helps to model the possible states of an object.

* Notations of State diagram

1] States → A state is a condition or situation in which an object can be at given time.

Shown as rounded rectangle.

it may contains

- entry/action
- exit/action
- do/activity.

2] Initial state:

- Shows where the object start its life
- only one initial state per diagram.

Notation → ●

3] Final State.

- Represents the end of the objects behavior or life.
- only one or few final states possible.

Notation → ◉

4] Transitions.

- Shows the movement from one state to another.
- Caused by a trigger.
- May include guard conditions & actions.

5] Trigger & Event

- A trigger is an event caused by transition.
- An event can be signal, call, input or time.

6] Gaurd Condition

- A boolean expression that must be true for the transition to happen
- Written inside square brackets [ ]

7] Actions.

- Tasks or operation to perform during the transition.
- Written after / on the arrow.

5] Ports:

- Used in composite state for showing interaction point
- Represent as small square on the state boundary.

8] What is state? Explain the contents of a state in detail

→

- state is current condition or situation of an object during its lifetime.
- It shows what object is doing or waiting for particular state.
- It is used in state diagram to model the dynamic behavior of an object.

P.T.O

Content of a state

1] Name of state
- Label or title of state
- Must be unique or descriptive.
- Generally written inside state box

2] Entry/Exit Action

3] Internal Transitions → These are the transitions happened without changing states

4] Substates :
- The state having multiple states running parallely is called substates.

5] Defered Events
- Event that are postponed and not handled in the current state.
- Written with keyword defferr.

Q] What is interaction & behaviour modelling ? List different diagram used for this modeling ? Draw Components used in 2 diagram.

→ ◦ Interaction Modeling

- Interaction modeling is all about showing how objects communicates with each other to do a specific task.

Types of diagram.

1] Sequence diagram

2] Communication diagram.

3] Timing Diagram

◦ Behavior Modelling

- Behavior modeling is concerned with the internal working & activities of objects or components. It focuses on how object changes due to different state event & condition.

Diagram

1] State machine Diagram.

2] Activity Diagram.

3] Use case Diagram

Return, create object & destroy object message in the context of sequence diagram

→

1] Synchronous Message

- The sender waits for receiver to complete the message.

Notation → Solid line with filled arrow (⟶)

2] Asynchronous message.

- The sender doesnot wait for the receiver to ~~send~~ respond.

Notation → Solid black line with opened arrow (⟶)

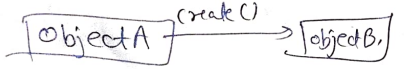
3] Return message.

- Receiver sends back acknowledgement.

Notation → Dotted line with arrow (⇢)

4] Create Object message →

- One object creates new object during the interaction,
- Notation → Arrow pointing to a new object's lifeline

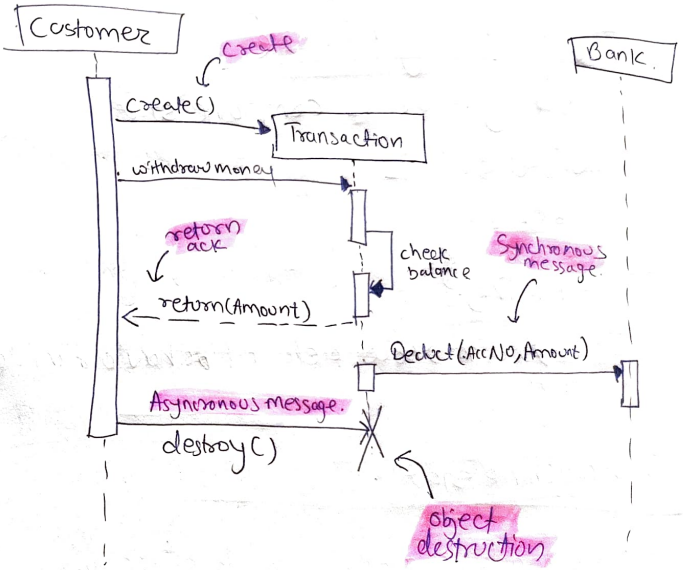


5] Destroy Object message

- An object is deleted or destroyed.

Notation A large X symbol at the end of object's lifeline.

eg



Q] In the context of a state diagram, explain concurrent states, nested states, sub-machine states & composite states with suitable examples

→ 1) concurrent → back

2) Nested state.

- Nested state are the inner states inside the composite state. These show what happens inside a Higher-level state.

eg → washing machine

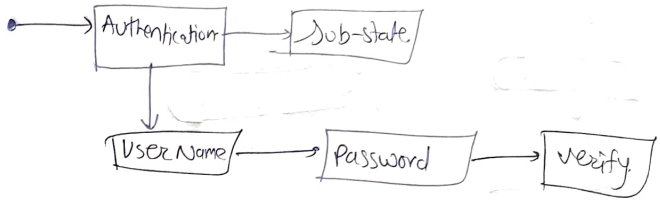
- washing mode is composite state.
- Nested state is
  - Soaking
  - Washing
  - Rinsing.
- It is used to show detailed behaviors inside a state

3) Sub-machine state.

- A sub-machine state is a state refers to another complete state machine. It is like calling a seperate diagram to avoid repeating same logic.

eg → In login authentication is main state

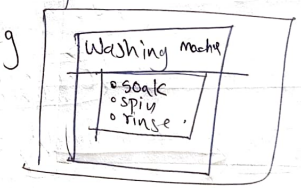
- Instead of drawing again & Authentication is sub-machine state that will be called.



4] Composite State

- A composite state is a state that contains other state inside it. It helps in grouping similar things together to simplify the diagram.

eg



Q] Dynamic Diagrams in UML

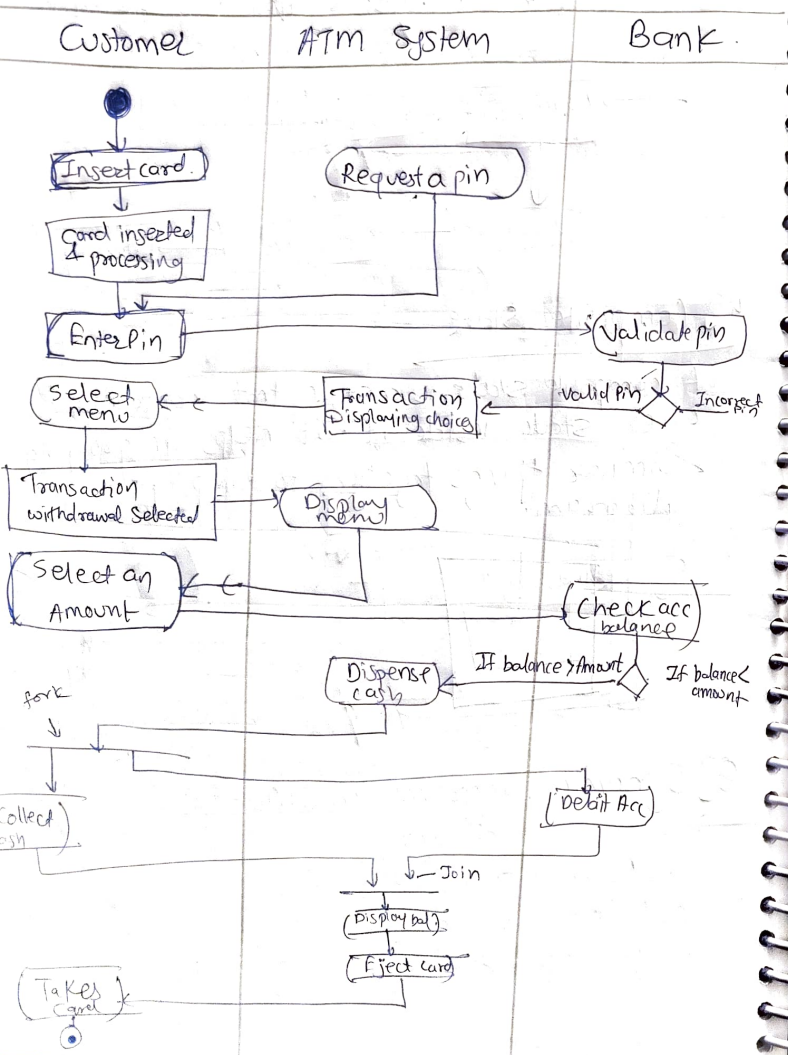
→ a) Sequence diagram
b) Activity diagram
c) Statemachine diagram
d) Use case diagram

Activity diagrams

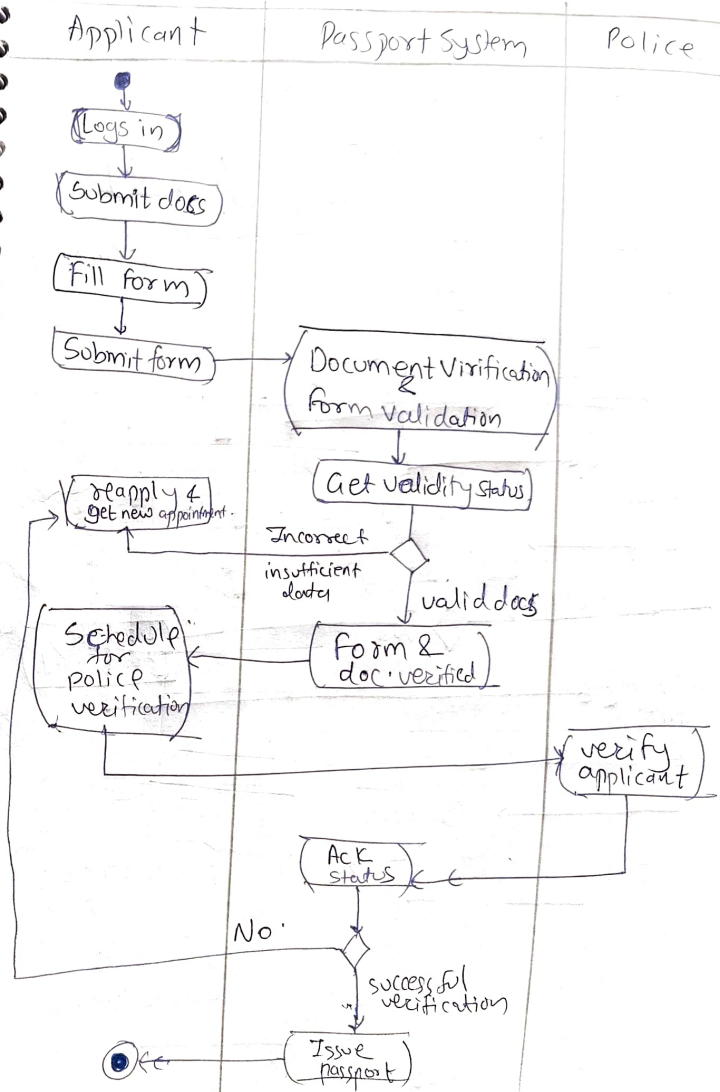
Q1) ATM ~~system~~ system

Technical

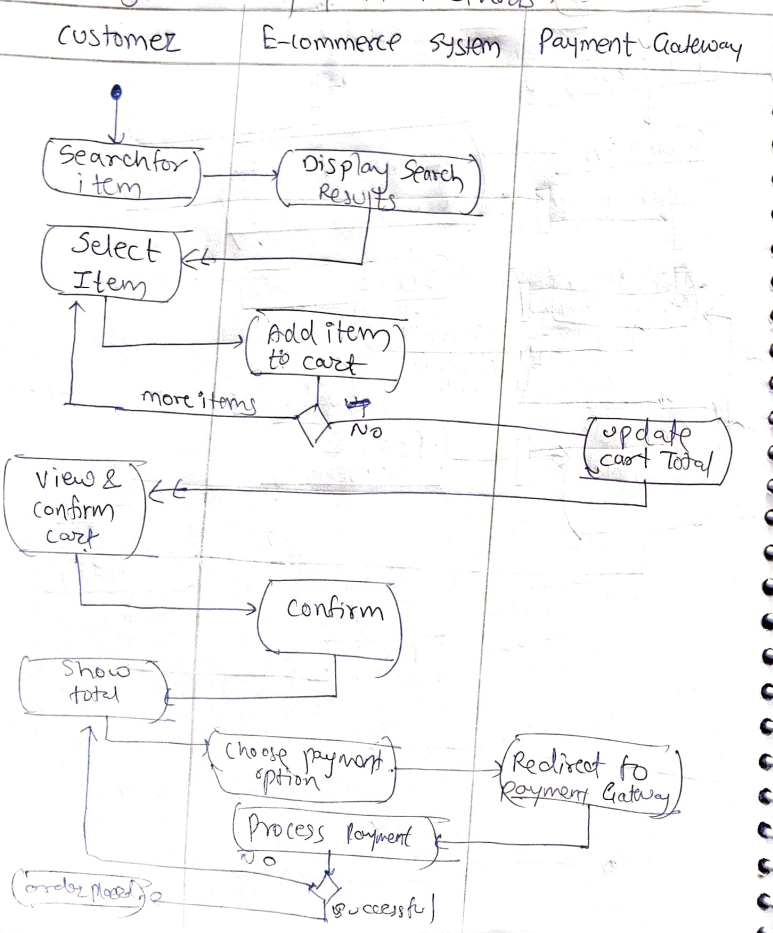
Q] 3.1.6



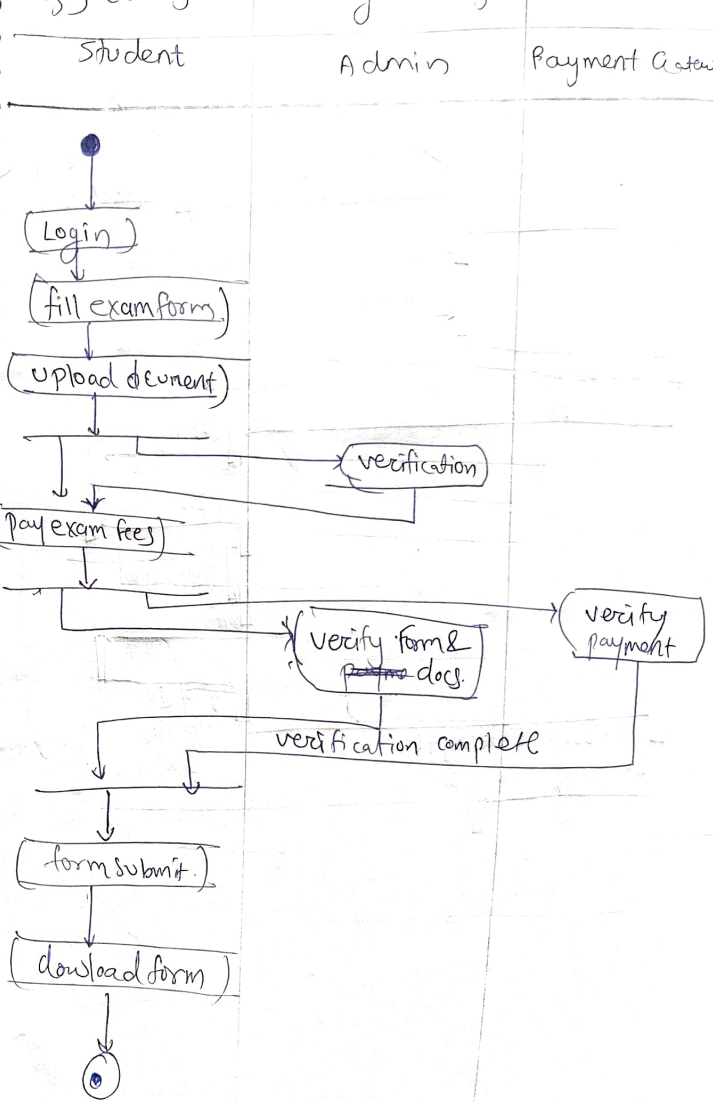
Q] 3.1.9 Technical (Passport)



Q] A popular e-commerce website application serves customers to purchase items. Customer can select item by using 'search item' facility & add it in the cart. The cart total is calculated & customer can pay bill by using various payment methods.

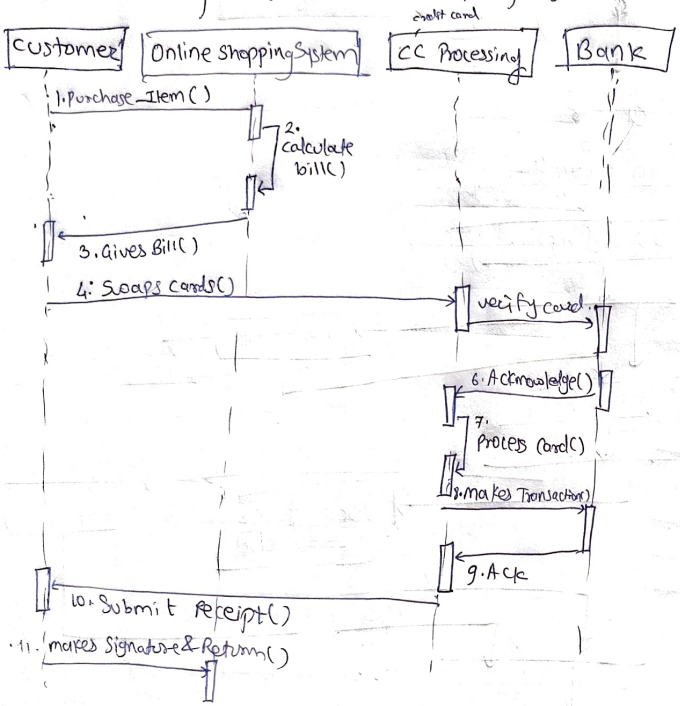


Q] College's Library management system

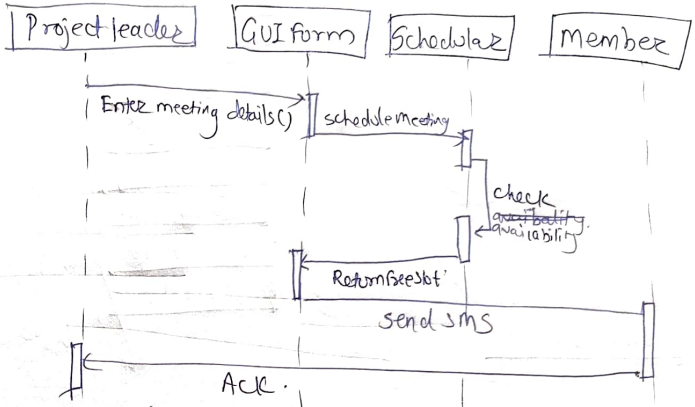


Sequence diagram

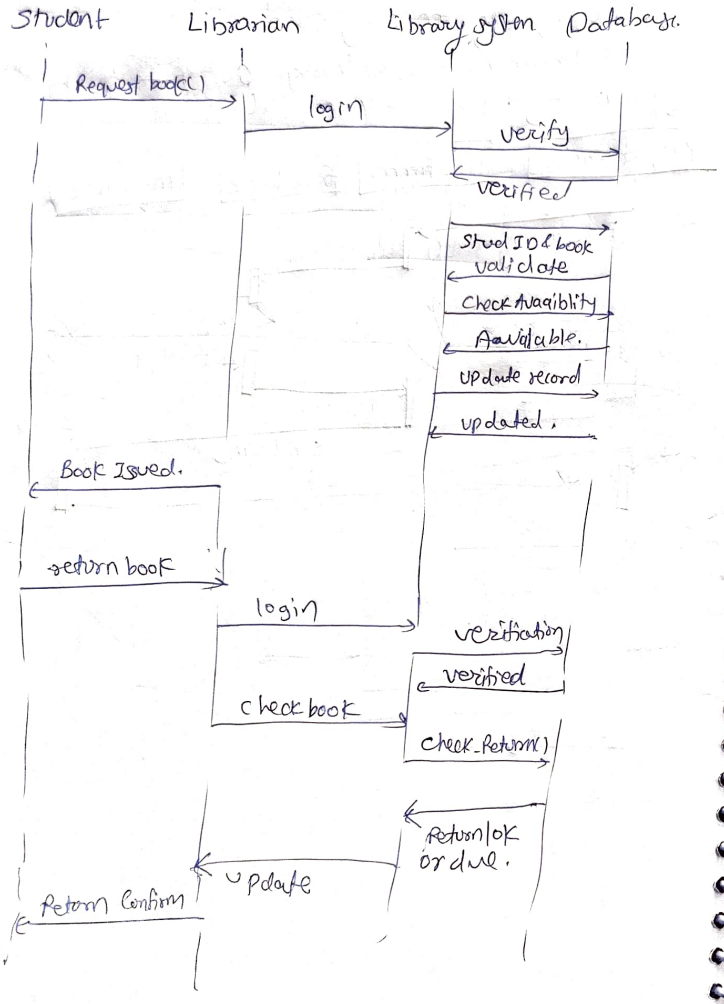
Q] Sequence diagram for online shopping system



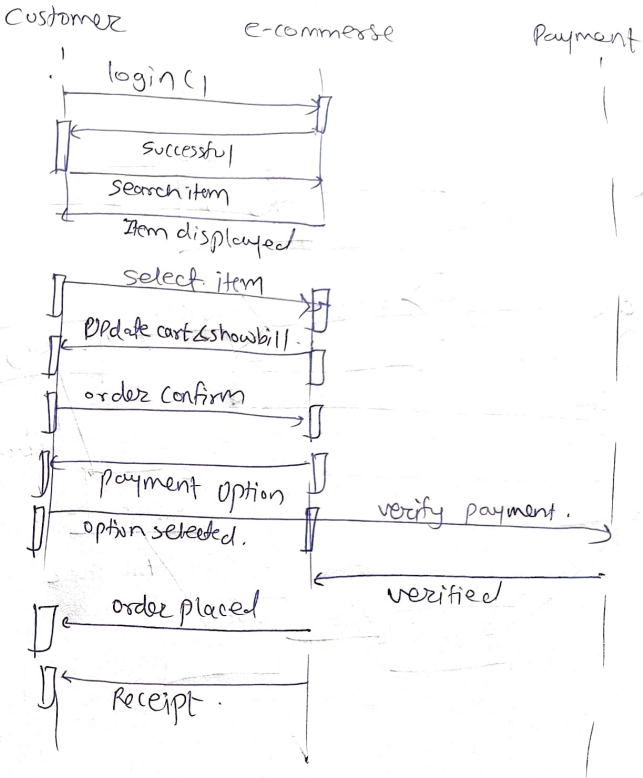
Q] The project leader schedules a meeting of project by group by using meeting schedular. Project leader interacts ~~with~~ through GUI ~~form~~ form to schedule the meeting. Draw sequence diagram



Q] College Library Mangement system

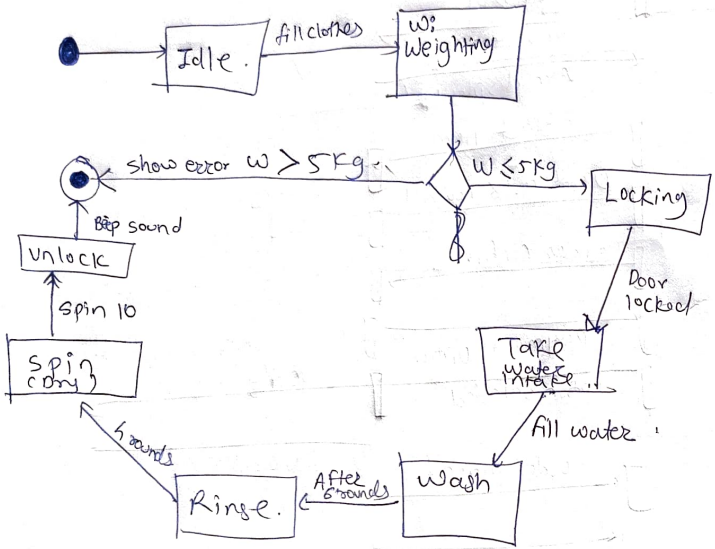


Q] Ecomerse site.



State diagrams

① Washing machine.



② Coffee vending machine.

UNIT-4 Object oriented Design Process

Q] What is OCL? Explain its mechanism, what is expression in it, precondition and postcondition. Explain its application ~~with~~ example of OCL.

→

- Designing model are represented ~~with~~ by UML but graphical representation is not enough.
- OCL stands for Object constraints Language (OCL)
- It is formal language to define expressions on UML model. mainly to specify constraints and rules that cannot be easily expressed by diagram.
- ~~OCL is used to write constraints in~~
- OCL have four parts:

1) Context → Defines situation of element

2) Property → It represent characteristics of the element.

3) Operations → Various arithmetical, logical operations can be carried out by design

4) Keyword → Some keyword are stored in language with some meaning ex, if, else, not, or, implies.

- In OCL self is used to define/represent the current object.

  eg self.age , self.name etc..

- The invariant is for describing the condition that must always hold true of all instances of the class.

- eg → context Employee.
  inv: self.age > 20

→ age should always greater than 20 of every employee.

- Preconditions → The conditions that must be true before a method or operation executes.

Syntax → ~~Syntax~~ context <classifier> :: <operation (parameter)>
pre <constraint name> : <bool OCL expression>

eg →

context Employee : Registration (i : integer)
~~Constraint~~
pre : self.age >= 18 & i > 0

Precondition is that employee age should be always greater than 18 years & value of i is also greater than 0

- postcondition → It is the condition that holds true after the method or operation executes

Syntax → context <~~const~~ classifier> :: <operation <parameter>>
~~Con~~post <constraint name> : <Boolean OCL expression>

eg →

context Person :: Authenticity ()
post : self.isValid = true.

person is authentic

- Expressions are the basic building blocks in OCL that evaluate to some value.
- It can be boolean, string, args, integer, etc..

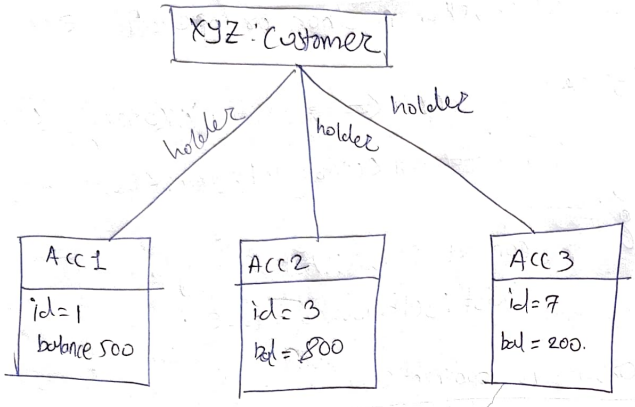
Types

1) Boolean → True or False
2) Logical expression → AND or NOT
3) Arithmatic → +, −, *, /

Application of OCL

1) makes sure data is correct.
2) Check before doing action
3) Check after doing action.
4) Find mistakes in designs.

eg



OCL of above UML diagram is

XYZ .account → select (balance > 300) = {acc1, acc2}

XYZ . accounts . balance → sum() = 1500

XYZ. accounts . includes (ac4) = False.

Q] Explain steps in designing business classes.

→ 1] Identify Boisness classes
- find out the main things involved in the system

eg → book, member, Librarian.

2] Define attributes
- what data each class should store,

eg Book → title, author.

Member → name, memberID

3] Define Operations
- Define activities to be performed.

member → register(), add()

Book → issue(), return().

4) Set relationships betn classes
- Show how they are connected

5] Draw Class Diagram.
- use UML to draw the final class diagram with all classes, attributes, methods & relationships.

Q] With the help of UML, how can we package the classes involved within a system scenario?

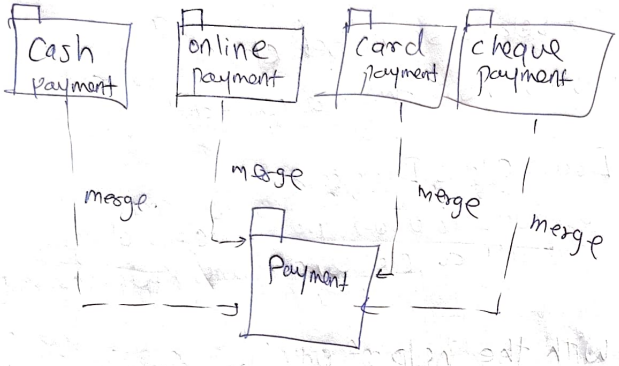
→
- In UML we can group related classes into a package
- It helps system to be organized, clean & easy to understand.
- The package is like a folder that holds related classes, interfaces or diagrams.
- It is used to group elements that belongs together

Steps

1] Identify related classes.
- Group classes that perform similar or related task.

2] Create UML package.
- Draw big rectangle with tab on left (like folder)
- Write package name in tab.

3] place related classes inside the box
- Draw classes inside package box.



Q] Explain purpose of access layer, major tasks performed in access layer & benifits of access layer class. what are the points one should consider while designing the Access Layer classes

→

- An access layer is to ~~create or it~~ the part of the software system that helps to communicate with the places where the data actually reside.
- The access layer connects the buisness layer to the database and also called as Data access Layer
- The main responsibility of access layer is to provide link between buisness or view objects ~~and~~ and data storage.

* Major tasks of access layer *

1] Translate the request :

- It converts the request from business logic into a database query to access layer

2] Translate the Results

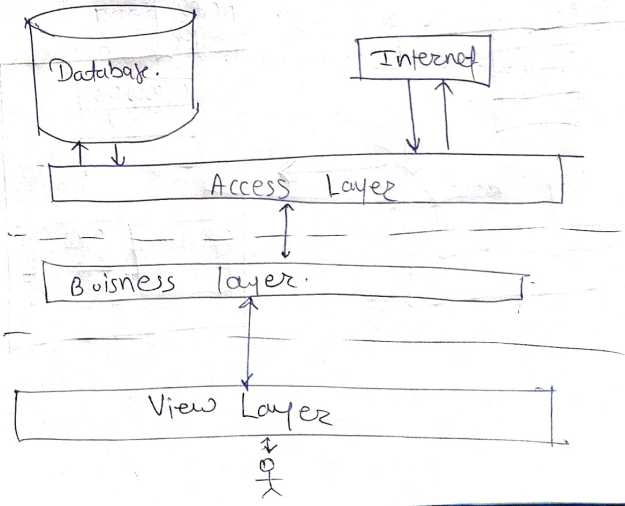
- It converts database results into objects or values that buisness layer can understand

* Benefits of access layer *

1] Reusability.

2] Supports 2-tier Client-Server architecture.

3] Easy integration with buisness layer.



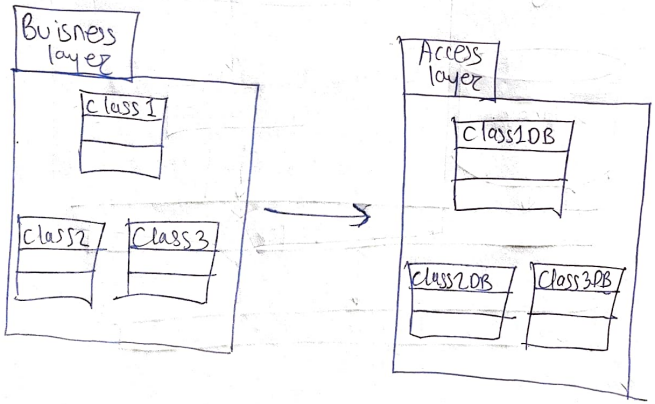
Points to remember

1) Name classes clearly
2) Keep only DB code in access layer.
3) Write reusable & short methods.
4) Handle DB errors properly.

Q] Explain process of access class

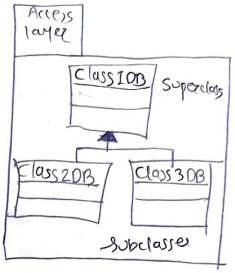
→ 1] Mirror Buisness class package

- Create one access layer class for each business layer class.
- for eg: if there are 3 buisness layer class then create 3 access layer class



2] Mirror Super-Sub relationship

- Define relationship between these access layers classes

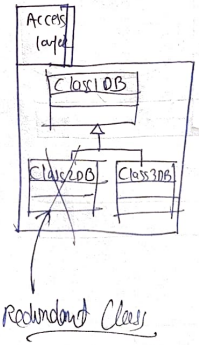


3] Simplify Classes & Relationships :

- In this step eliminate redundant or unnecessary data.

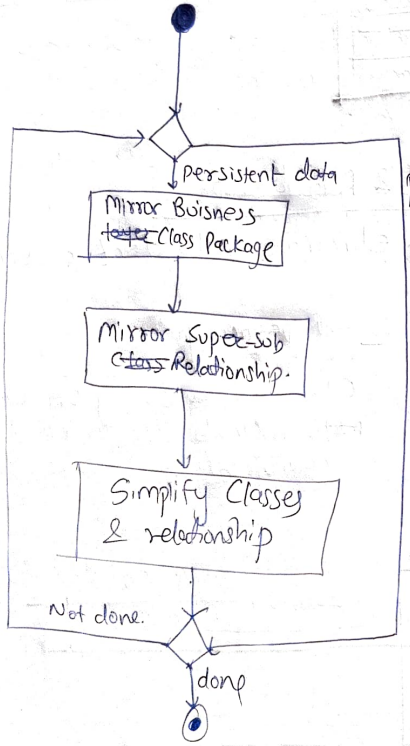
a) Redundand Class → Repeating process

b) Method class → Class contain only one or two methods then they can be combined.



4] Iterate & refine

- Test & improve the classes.
- Refactor the code if needed, remove method classes.
- Repeat until simple, ~~secure~~ reusable & efficient.



Persistant data

Q] Explain object-relation mapping & capabilities to be defined

→
- Object relational mapping (ORM) is a technique that connects object-oriented programming language like Java, Python with relational ~~database~~ like (mysql, oracle, PostgreSQL)
- It helps programmers store & fetch data from the database without writting complex queries.
- It converts classes & object into database tables & rows & also converts database data back into object.

Various mapping capabilities between ~~object~~ relation systems are

- Table - class mapping
- Table - multiple class mapping
- Table - inherited mapping.

- The above types of mapping are of two types.
  1. Map from relational system to object oriented system
  2. Object oriented to relation system.

- In OOP, data is represented by classes & object.
- In relational database, data is stored in tables & rows
- Hence ORM helps in translate bet^n two formats making it easy to save & retrieve objects from the database.

Q] What is object relational systems? How can you map relational data with application objects using mapping tool?

→ • Sometimes the data of the object oriented system needs to be used in relational ~~database~~ tables. But there is fundamental gap between OOP & DBMS.

• To resolve the mis-match between ~~this~~ ~~two~~ OOP & relational system a mapping tool must be established between application object & relational ~~object~~ data.

• Creating an object model from existing relational database is called reverse engineering

• Creating an relational database from existing object model is called forward engineering.

• This forward & backward relationship must get combined in order maintain relationship between the object & relational data.

• Object Relation Mapping is used to map application objects to relational database structures.

Tools used

• Java → hibernet

• Python → Django ORM

Q] Steps in View Layer classes

1) The macro level UI design process

• Main objective of macro process is to identify classes that interact with human actors by analyzing use cases

2) Micro level design activities

a) Designing view layer objects.

b) prototype the view layer interface.

c) User satisfaction test

Q] Explain macro level process:

→ • Use cases help in understanding user's objective & tasks.

• Different users have different needs.

• Main objective of macro process is to identify the classes interacts with human actor by analyzing use case.

• It have two main steps.

1] If the class interacts with human actor go for following else move to another/next stage.

~~1.1~~] Identify view object for class:

• Zoom in view object with the help of ~~sequence or collaboration diagram.~~

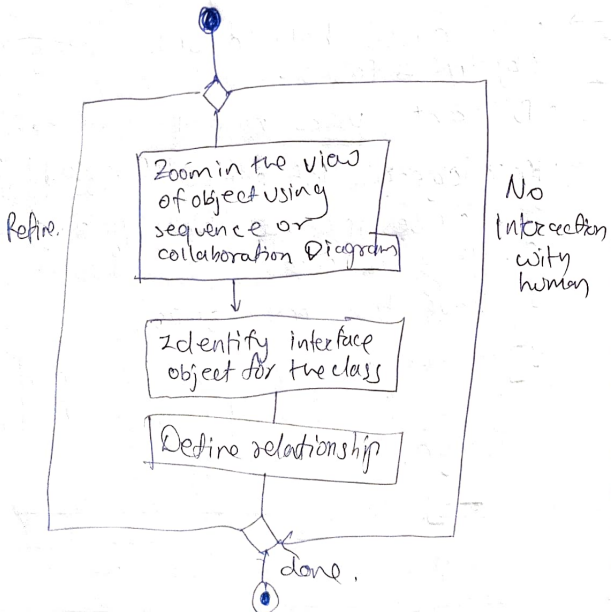
1.1] Identify view object for class :

- Zoom in the view object with the help of sequence or collaboration diagram & identify responsibilities & requirements for this class.

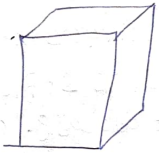
1.2] Define relationship among view object

- The view object are also called as the interface objects. The interface are associated with business classes. Therefore business classes help in defining the relationship among the view classes

2] Refine & Iterate.



Q] Difference betn component & Deployment Diagram

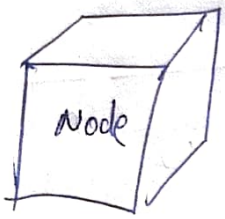
feature	Component Diagram	Deployment Diagram
1) Definition	Shows logical component	Shows physical component
2) Focuses	How system is divided into component	How the system is installed on devices
3) Used for	Understand software structure	Understand hardware installation.
4) Represents	Class, files, packages, libraries	Nodes, artifacts
5) Symbol	rectangle with two tabs on left side	Cube (Node symbol)
6) ex→	Login module, UI component	eg→ mobile device, web server.
7) figure		

• A deployment diagram *

- A deployment diagram in UML is used to model the physical layout of the system.
- It shows how software component are deployed on hardware devices.

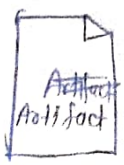
i] Node

- It is a physical device or hardware that hosts software components
- Nodes are shown as 3D box
- Nodes can be physical or virtual.



] Artifacts.

- It is software file like a program, library or database. or document.
- Artifacts are deployed on node & used during system execution



UNIT-5 Software Design principles & pattern

UNIT-6 : Software Architectural Design

1] Designing object oriented Software Architecture.

→ • object oriented software architecture is designed based on some concepts like information hiding, classes & inheritance.

• Classes are designed using information hiding concept.

• OOSA follows the principles like abstraction, inheritence, polymorphism & encapsulation.

1] Designing Information Hiding Classes.

a) Bondary classes : These types of classes is used to represent system's boundary. for example. GUI screen.

b) Control access. → These type of class is used to define buisness logic or flow of control.

c) Entity Classes → These are the classes with some meaning associated to them. These classes have long lifeline

2] Designing Classes, Interfaces & operations

• The class Diagram is designed to show the relationship between these classes

P.T.O

- Once the classes are made we can define their attributes & ~~relation~~ operation.
- Interfaces are also designed to define common behaviour of the multiple classes.

3] Designing class operations from Interaction Modeling

- Interaction modeling includes activity diagram, sequence diagram & communication diagram.
- These diagrams helps to understand how object interacts over time to perform tasks.
- These activities are normally carried out by actor who is interacting with system.

4] Modeling the state chart diagram

- State chart shows the lifecycle of the system - how its state changes with events.
- It helps to understand the workflow of the system.

Tip → Draw all 4 steps diagrams If que asked seperately

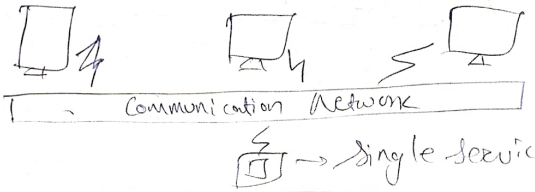
Q] Explain Designing Client/server software Architecture.

→
- The client server architecture is kind of architecture in which a client is a requester for a service and server is provider of services
- There is a difference between server & services. Server is hardware or software system that provides one or more services for multiple client. Where as the service in client/server architecture refers to the software component that can fulfill need of many clients.
- In client/server architecture the service runs on fixed number of nodes & client has fixed connection to the server

- There are various client server architecture pattern
  a) multiple client & single service
  b) multiple client & multiple services
  c) multi-tier client services.

a] multiple-client & single services

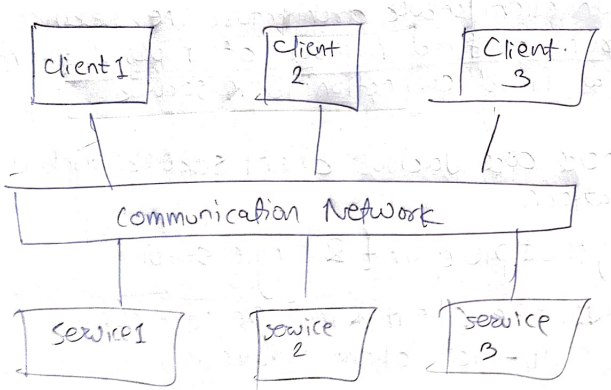
- In this architecture, there are several client that request for same services



- Eg → A bank can have multiple ATM in which multiple client can access single bank service

b] Multiple client – muntiple services

- In this architecture, the multiple client have multiple services, multiple client can not only access multiple services but multiple services can also communicate with each other.



eg → Bank can have multiple ATM where multiple client can have multiple services like, Axis bank user can visit BOB ATM and use Axis bank services.

3] multi-tier client server architecture pattern

- In this architecture, system is split into multiple layers (tiers).
  - Client Tier: (UI)
  - Buisness Tier
  - Data Tier: Database.

Q] Designing Service oriented Software Architectures

→
- Service oriented Software Architecture. (SOP) is the architecture where multipe services communicate on network.
  - Each service is autonomous, reusable & provides specific functionality
  - These services are distributed that means these services execute on different machines.

Key principles of SOA Design

1] Loose coupling
- Services should work independently
- Any change in one service must not affect other

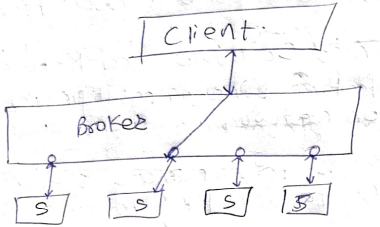
2] Reusability

3] Abstraction

4] Statelessness → Do not store unnecessary data bet" services

5] Clear Interface.

- In service oriented architecture broker is a middle man between client & server.
- A client requests a service. It formats its request & in a specific format & send it to broker. Then the broker choose most suitable server to process the request. When the link bet^n client & the server is setup, they may start communication ~~facing~~ Peering the broker.



S: server.

- There are two types in it

1] Location transperency → If the service moves its location the client dont know about it but broker knows

2] Platform transperency →
- It means service can execute on different platform.

The most used technology used in SOP is web services.

The web server are software system that are displayed on web browser using web protocol.

8] Designing Component based Software Architecture.

- In componet based software architecture the software is structured into components & interfaces between the components are defined.
- Each component has a well-defined interface & can be reused across different applications.
- The primary goal of component based SA is that same software architecture should be capable of being deployed to many different distributed configuration.
- There are two types of components :- composite & simple component
- Composite component is component which is compose of one or more parts of components.
- The simple components have no parts components in it
- The communication between the components is message communication. That means component A sends messege over network to component B.

Distributed-component-based Software Architecture

- ~~There~~ In distributed architecture, components are deployed on different networked systems.

Three steps to design distributed-component-based systems

1] Design the architecture. →
- Identify components & how they communicate.
- Decide client-server / peer-to-peer, multi-tier setup

2] Design the Components-

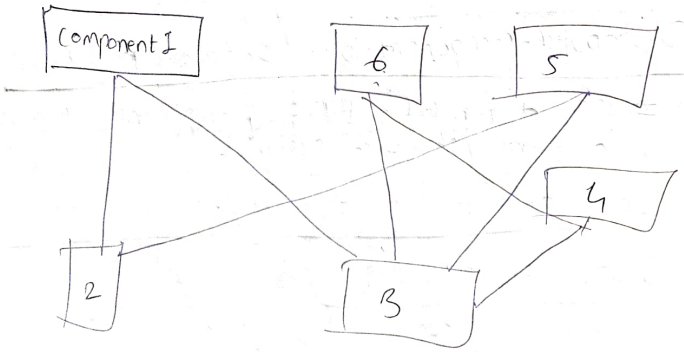
- Define the internal behavior of each components
- Design with low coupling & high cohesion.

3] Deploy the application

- place components on appropriate network nodes
- Ensure proper inter-component communication.

Composite Subsystem & components

- A composite component is formed by combining multiple small components or subcomponents
- It acts as a subsystem that provides higher-level functionality
- Helps in managing by grouping related features together.



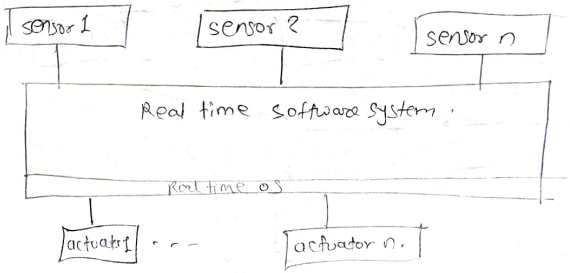
9] Concurrent & Real time Software architectures

Concurrent & Real Time Software Architectures

- Real Time Software architectures are ~~concurrent~~ system that focuses on building software system that can perform multiple task simultaneously (concurrent) & respond within specific time. (real time)
- It is used in application where timely & predictable response is critical like air traffic control, robotics

★ Characteristics of Real Time Software Architecture ★

- Real time Software architecture is based on timing constraints.
- Real time system contains real time application, real time operating systems & real time I/O subsystem.
- System is consistent. i.e, it works same everytime.
- The system ~~not~~ is complex as it have to deal with multiple task that generate multiple inputs & outputs



There are two types of real time system.

1] Hard real time system & 2] Soft real time system.

1] Hard real time system → • It is type of real time system where the deadline is ~~missed~~. very strict failure in dealine causes system failure

eg → Airbags in car system.

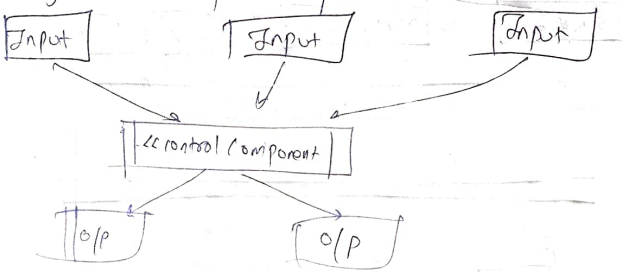
2] Soft real time system. → It is type of real time system where missing dealine is tollerated. but may causes system performance.

eg → Video buffering

Control pattern for Real Time software Architecture.

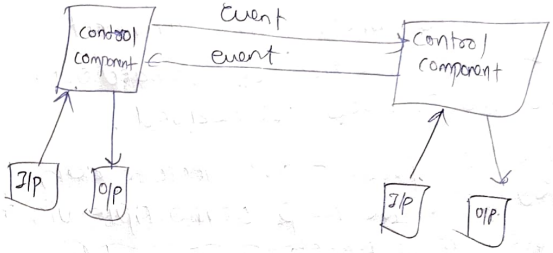
1] Centralized control architectural pattern

→ In this architectural pattern there is one control component that is centralized one. There are multipe input component that sends input to this centralized component & this controlling component sends the output signal to output component.



2] Distributed control pattern

• There are multiple control components which controls the part of the system. Control components communicate through peer to peer communication by sending & receiving events.

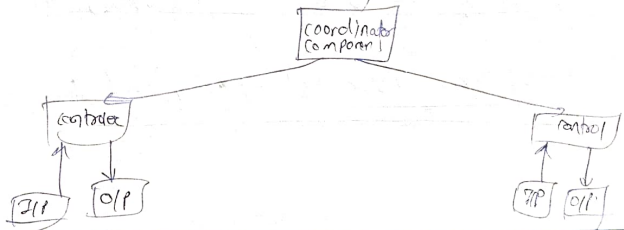


3] Hierarchical control architectural pattern

• In this architecture there are several control components which controls the part of system

• Along with these control component, coordinate component is present which ~~act~~ controls this control components.

• The coordinate component decides the task for each control component & the status is updated to this control component.



8] Designing product line Architecture

→ • Software product line (SPL) architecture is a system that have some common functionality
  • It is reusable system.
  • During development of software product line, the requirement analysis, architecture design & implementation are carried out. Using this the products are derived.
• The single line product architecture can be designed with the help of multiple views that are requirement models, static models, dynamic models.

There are two main process.

1) SPL engineering

→ • These process is also called as domain engineering
  • This focuses on multiple view

2] Software application engineering

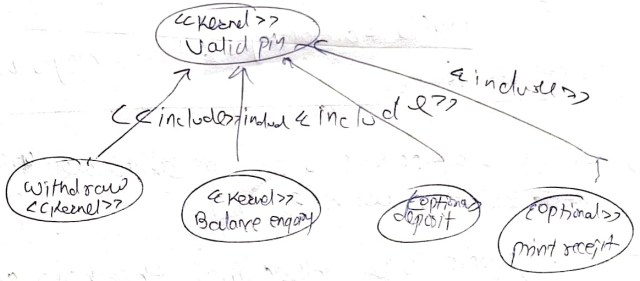
• It is individual product line member derived from the SPL multiple view model.
• For development user select required features.
• These architecture determines which reusable components are needed

Design & Requirements modeling for SPL

• Using the component diagram, the design model for SPL can be represented.
• for requirement modeling use case modeling & ~~design~~ feature modeling, ~~are two~~ are two types.

1] Use case modeling

• The requirements of system are defined in terms of use case and actors



2] Feature modeling.

• It focuses on SPL variability. they are categorized as
1] Common features : supported in all PL
2] Optional feature. required in only some PL
3] Alternative features : choices

8] Explain the Entities involved in documenting a software architectural patterns

→ Software architectural pattern is a reusable sol^ to common software design pattern

Entities

1) Pattern Name
2) Problem
3) Context → Explain the situation where pattern is useful.
4) Solution.
5] Structure
6) Behavior
7] Consequences → advantages & disadvantages of pattern
8] Related Pattern.

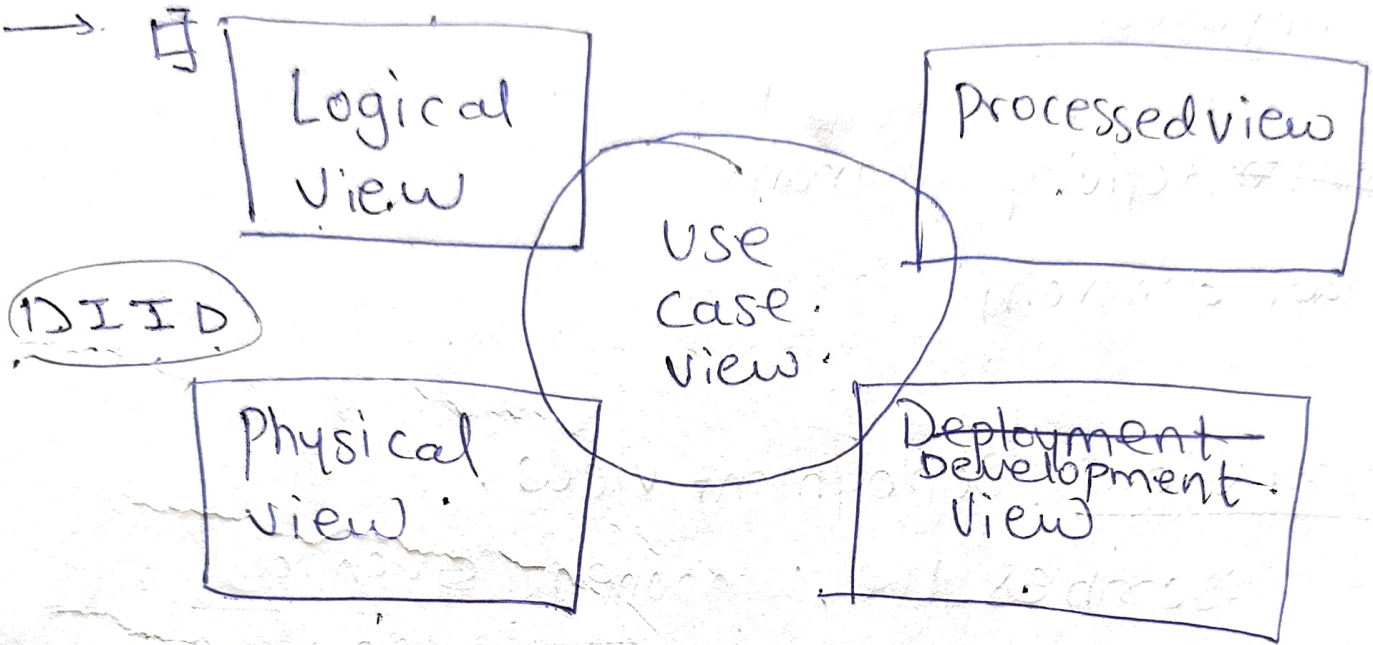
9] What is dynamic view of software architecture ~~with~~ explain with example

→ • Dynamic view describes ~~the~~ how system behaves at runtime.

- It focuses on how components ~~comm~~ interact with each other how they are related.
- Diagram used for Dynamic view
  - Sequence Diagram
  - Activity Diagram
  - Communication Diagram
  - State chart Diagram

9] Explain 4+1 view architecture. 4+1 → 1 2 3 DII

→



1] Logical view → • It is the design view & is responsible for defining the Software System

- Focuses on System Functionality as seen by user
- Useful for designers & developers to understand System ~~det~~ behavior.

Diagram used

1] Class Diagram
2] Object Diagram

2] process~~s~~ view

- It includes process and threads involved in the software system
- Process view mainly refers to the scalability performance, output & fault tolerance of the system
- Useful for architect & designers.

Diagram used

1] ~~Compo~~ Activity Diagram
2] State diagram

3] physical view / Deployment view.

- It describes the component such as hardware required for ~~deve~~ deployment.
- Focuses on network topology & communication

Diagram Used

- Deployment Diagram.

4] Development view / Implementation View

- focuses on the internal organization of the system
- Represents Modules, packages, components.

Diagram used

- Component
- Package.

5] Use case View (+1 view)

- Shows how the user will interact with system
- Overview of system design
- captures the requirements
- Helps stakeholders & buisness analyst.

Diagram Used

- Use case diagram.