# Syllabus.

## UNIT 1

### Concepts of Software Modeling.

Software Modeling: Introduction to Software modelling, Advantages, Principles.

Evolution of Software Modeling & Design Methods: Object oriented analysis & design methods, Concurrent, Distributed Design Methods & Real Time Design Methods, Model Driven Architecture, 4+1 Architecture, Introduction to UML, UML modeling building Blocks, COMET Use-Case-Based Software Life Cycle.

Requirements Study: Requirement Analysis, SRS design, Requirements Modeling.

Use Case: Actor & Use Case identification. Use case relationship (Include, Extend), Use Case Generalization, Actor Generalization), use case template.

Case Study: Real life applications (Online Shopping System).

PAGE NO.:
DATE: / /

# UNIT - 2

## Static Modeling

Study of classes (analysis level & design level classes).

Methods for identification of classes: RUP (Rational Unified Process), CRC (class, Responsibilities & Collaboration), Use of Noun verb analysis (for identifying entity classes), Controller classes & boundary classes.

Class Diagram: Relationship between class, Generalization / Specialization Hierarchy, Composition & aggregation Hierarchies, Association classes, Constraints

Objects diagram, Package diagram, Component diagram, Composite structure diagram, Deployment diagram.

# UNIT - 3.

## Dynamic Modeling.

Activity diagram: Different Types of nodes, Control flow, Activity Partition, Exception handler, Interruptable activity region, Input & output parameters, Pins.

Interactive diagrams : Sequence diagram, Interaction overview, State machine diagram, Advanced State Machine diagram, Communication diagram, Timing diagram.

## UNIT - 4

Introduction to Software Architecture, Imp of Software Architecture, Architectural structure & views.

Architectural Patterns : Common module, Common Component-and-connector, Common allocation

Quality Attributes : Architectures & Requirements, quality Attributes & considerations.

## UNIT - 5

Architectural Design and Documentation

Architecture in the life Cycle : Architecture in Agile Projects, Architecture & Requirements, Designing an Architecture.

Documenting Software Architecture : Notations, Choosing & Combining Views, Building documentation, Package Documentation Behaviour, Documenting Architecture in Agile Software Development Project.

PAGE NO.:
DATE: / /

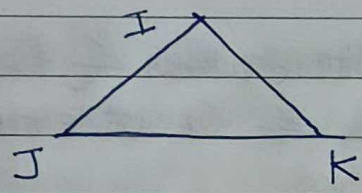
UNIT - 1

# Concepts of Software Modelling.

* Contents

- Why model ?

  Collection of artifacts

- Principles of modeling.
- What is Unified Modelling Language.
- Conceptual Model of the UML.
- 

Information, Knowledge, Judgment

Why model ?

- analyze the problem - domain

  simplify reality

  capture requirements

  visualize system entirely

  Specify the structure of system.

Separ. 3
Sep Lay 4

PAGE NO.:
DATE: / /

- Design the Solution

- Principles of modelling :

Choose your model well :-

- Every model must be expressed at different levels of precision.
- Best models are connected to reality
- No single model suffices.

= What is Unified Modeling Language.

---

* Behavioural Things :

State Machine : a behaviour that specifies the sequence of 'states' an object goes through, during its lifetime.

A 'state' is condition or situation during the lifetime of an object during which it exhibits certain characteristics and/or performs

engine Idling

PAGE NO.:
DATE: / /

* Grouping Things:

The organizational part of the UML model, provides a higher level of abstraction (granularity)

Package: a general element that comprises UML elements - structural, behavioral or even grouping things. Packages are conceptual groupings of the system & need not necessarily be implemented as cohesive

* Annotational Things: The explanatory model, adds ~~of~~ information / meaning to the model elements.

* Note:

Notation:

* Relationships:

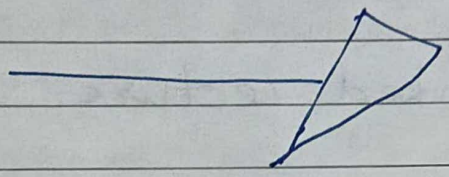
• Dependency: a semantic relationship where a change in one thing (the independent thing) causes a change in the semantics of the other things.

Articulates the meaning of the links between things.

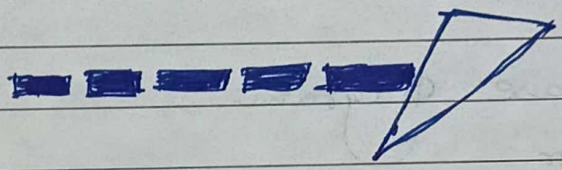
PAGE NO.:
DATE: / /

* Association : a structural relationship that describes the connection between 2 things.

* Generalization : Relationship : ("class" or "superclass")

Notation : 

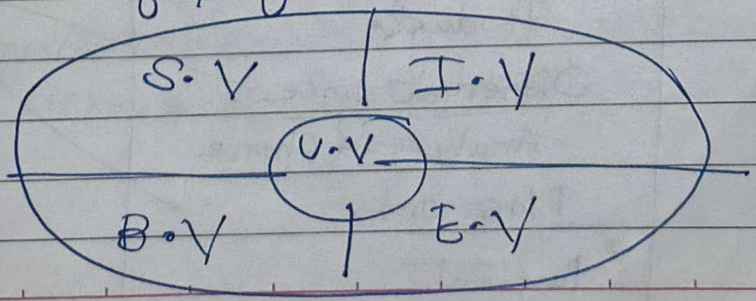
Realization : "a collaboration the Use Case." realizes.

Notation : 

Diagrams :

* Use on Diagrams :
  1. Class Diagrams : OOAD,
  2. Object Diagrams :
  3. Use Case Diagrams :

* Dimensions :
  .... of software architecture.



PAGE NO.:
DATE: / /

RULES:

UML has semantic rules for:
- Names, Scope, Visibility, Integrity, Execution.

Common Mechanisms:

8/2/2023
Wednesday

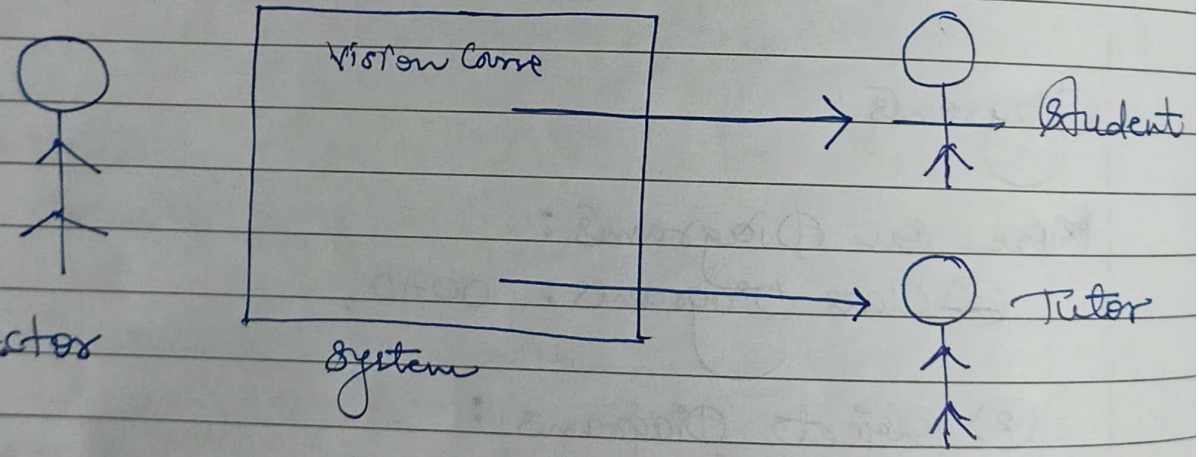
Bunked Lecture.

* Use Cases

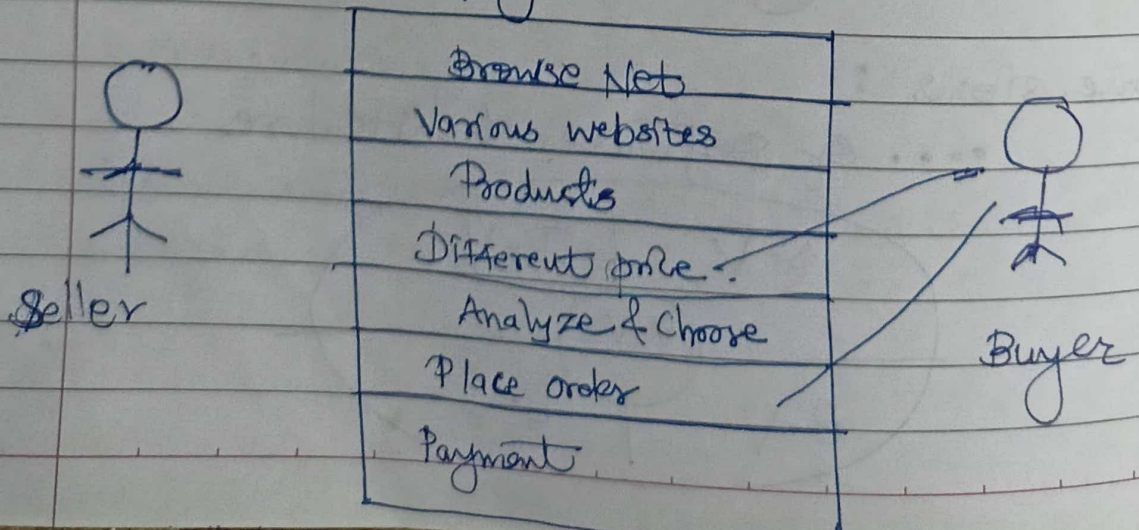
15/2/2023
Monday

Use Case diagram

Course ~~Administrator~~




Internet Shopping World.



Actors: Actors are people or external systems that interact with our systems.

Use Case Relationship:

Includes: (Process Order) (Validate login)

Excludes: (Hire Employees) (Hire International Employees)
Extends.

(Validate user.)

Generalization: (Retinal Scan) (Key Card Scan)

Includes Dependency:

Extends Dependency: defines, a use-case that is variation of another, usually for handling an abnormal situation.

* Design use case model with moderal system & identify entity involved in it.

actors, functionality & relationships exists between actors & use case.

PAGE NO.:
DATE: / /

20/02/2023
Monday

* ① Documenting Use Cases.

Use Case Model : Relationship can be Generalⁿ, Direct Access, Include, exclude.

---

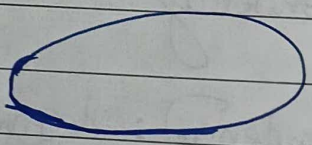
* Benefits of use cases : user variable functions.

Relationship between use cases documents opportunities for reuse.

Use Cases provide a basis for system testing.

Modeling Actor.

Secondary Use Case = ATM HELP / Extension points.

Primary Use Case = 

---

* UNIT-2 : Static Modelling

21/02/2023
Tuesday.

UML → Analysis & Design.

Use Case Model, Component Diagram, Inception, Targetted users.

Handling → □

Java → this pointer

PAGE NO.:
DATE: / /

* Study of Classes:

Classes, or instances of classes.

* Object-Oriented Analysis (Overview)

- Emphasizes a conceptual solutions fulfills.
- Library Information System
- Rational Unified Process (RUP)

Productivity.

Testing Phase: B testing

* Phases in the Process:

Major Milestones.

Inception Elaboration Construction Transition.

- Various aspect of information.
-

Elaborating phase → Defining, Validating.

Constructing phase → process optimization.

Transition phase → Getting User feedback.

Making the product available to

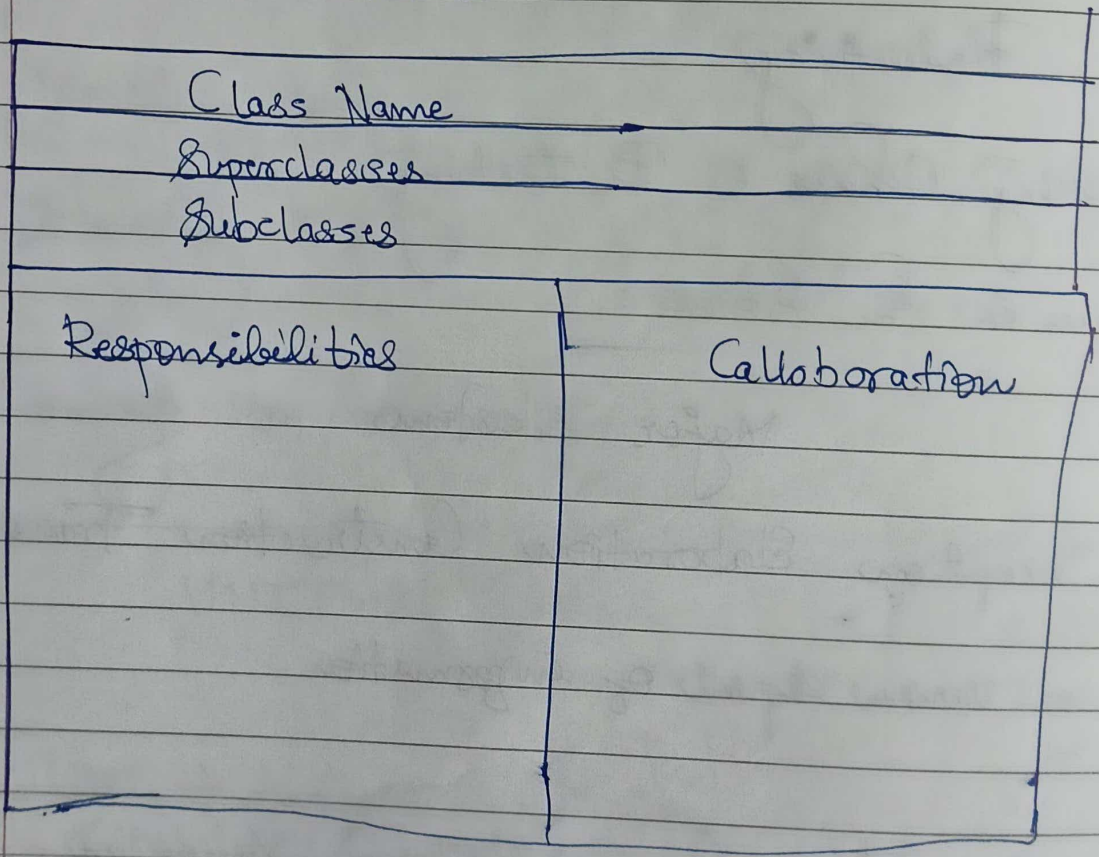
PAGE NO:
DATE: / /

22/2/2023
Wednesday

* Copyright → 2 projects
* CRC → Class Responsibilities, Collaboration
* Workflow : Artifacts, Roles, functionalities.
  ↓
  Piece of information

Activities : Tasks performed by people representing particular roles in order to produce artifacts

* CRC Card Format.




Responsibility : Knowledge Class maintains or Service class provides.

Collaborator : knowledge or Services are needed to fulfill a responsibility.

Use Case - Verbs
class - Nouns.

putpixel [2,1]

PAGE NO.:
DATE: / /

| Class Name : Patient | Class Type : (Abstract, Concrete, Domain) |
| --- | --- |
| Responsibilities | Collaboration |
| Attributes | ~~Generalization~~ Relationship Generalization<br>~~Aggregation~~ / Relationship<br>~~Association~~ : ~~Appointment~~. |

* Explain the CRC Modelling Technique

Program : The iterative steps for CRC modeling.

- Find classes
- Find Responsibilities
- Define Collaboration.
- Define the use-cases.

* Association, Aggregation Rules.

* What is Classifier :

* Scope : instance, Classifier, Abstract, Root, Leaf, & Polymorphic

PAGE NO.:
DATE: / /

* Complete Syntax: Complete Attribute, Complete operating syntax.

Object Constrained Language

Association Navigation, Association Visibility

Aggregation, Composition, Self-association (reflexive association), Generalization (inheritance).

* Case Study:

The hospital wishes to maintain the database to assist with the administration of its wards & operating theatres & maintain information related to its patients, surgeons & nurse. Most patients are assigned to wards on admittance & each ward may contain many patients. How were senior surgeon at hospital may have private patients who are assigned to private rooms a nurse may or may not assigned to ward and he or she cannot assign to more than one ward

Answer: Step 1: For finding classes

Description: Theatre may have many nurses assigned to it patient may have no of operations the information to recorded about an operation the types of operations patients surgeon involves, date, time. Only one surgeon may perform an operation, any other surgeon present being considered as a assisting operation

PAGE NO.:
DATE: / /

Step 1: For finding classes we use noun, phrase approach.
Each Consultant has specialization.

Step 2: Classes with roles: Stores information of these staffs.

Hospital name, Hospital Staff associated with it.

General information: Patient information & consulting doctor for that patient.

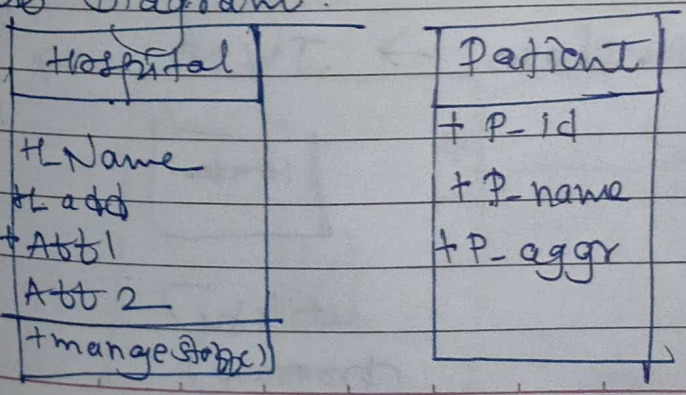
Ward:
Nurse:

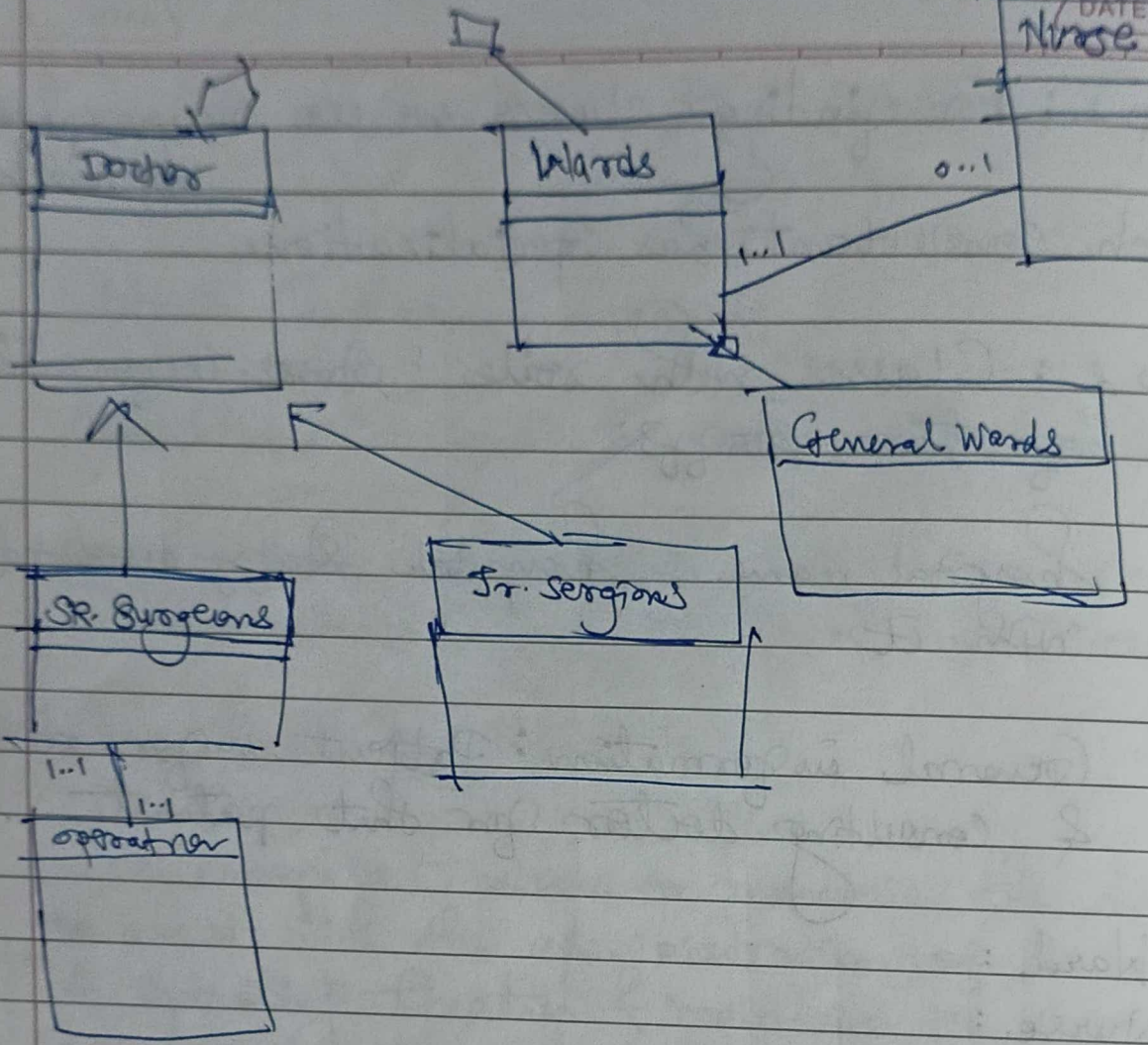
Step 3: Association between objects & classes:

Step 4: Attributes & operations of the doctor

Nurse: allocated ward.
Ward: Capacity
Operation: performed

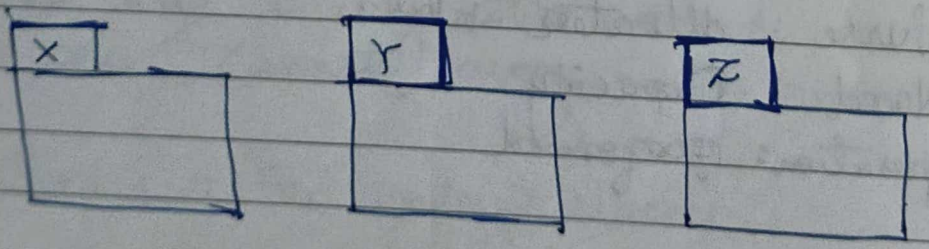
Class Diagram:






* Package Diagram :

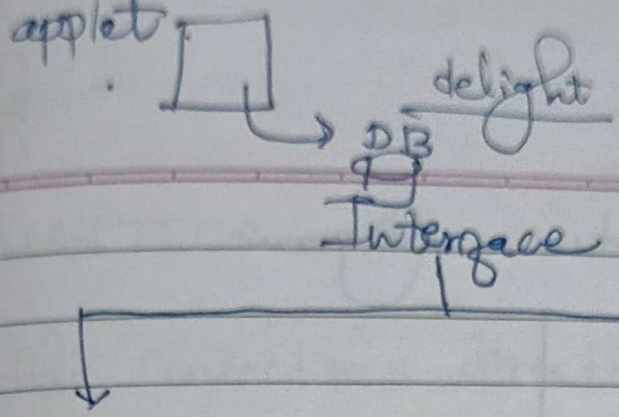
* Import vs Access




- java → .class → JVM → .exe

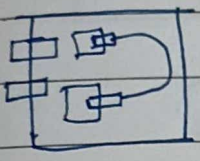
loader

utility programs.




~~Reserved~~ Provided ○—

~~Required~~ Required ⊂



interface << >> , ○

* Component Based Development:

~~Shopping~~ Cart:

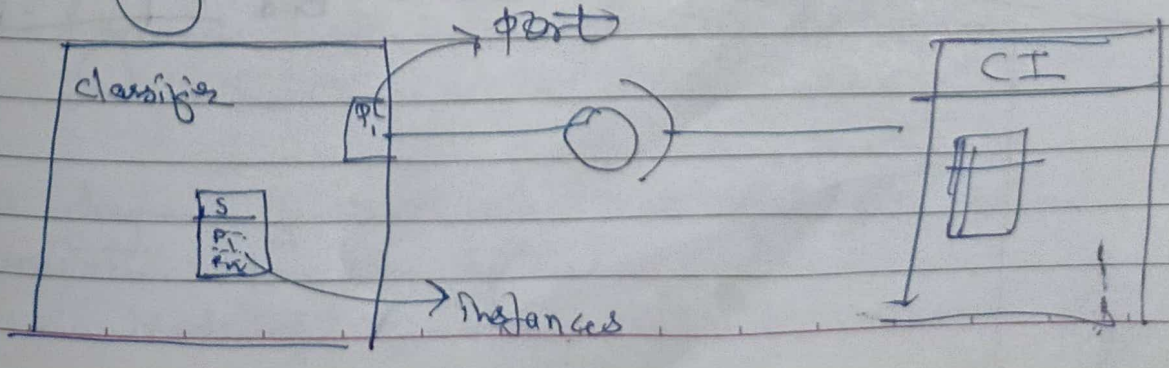
~~Specification~~ Elements: Realization Elements

Eg. auxillary class, abstract class. Interface implementation

02/3/2023
Thursday

Copyright filing.

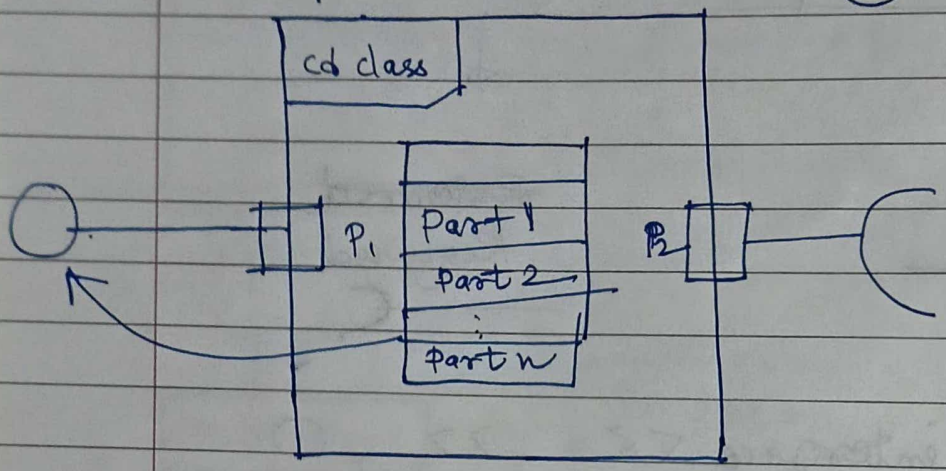
Component Diagram.



<< interface >>

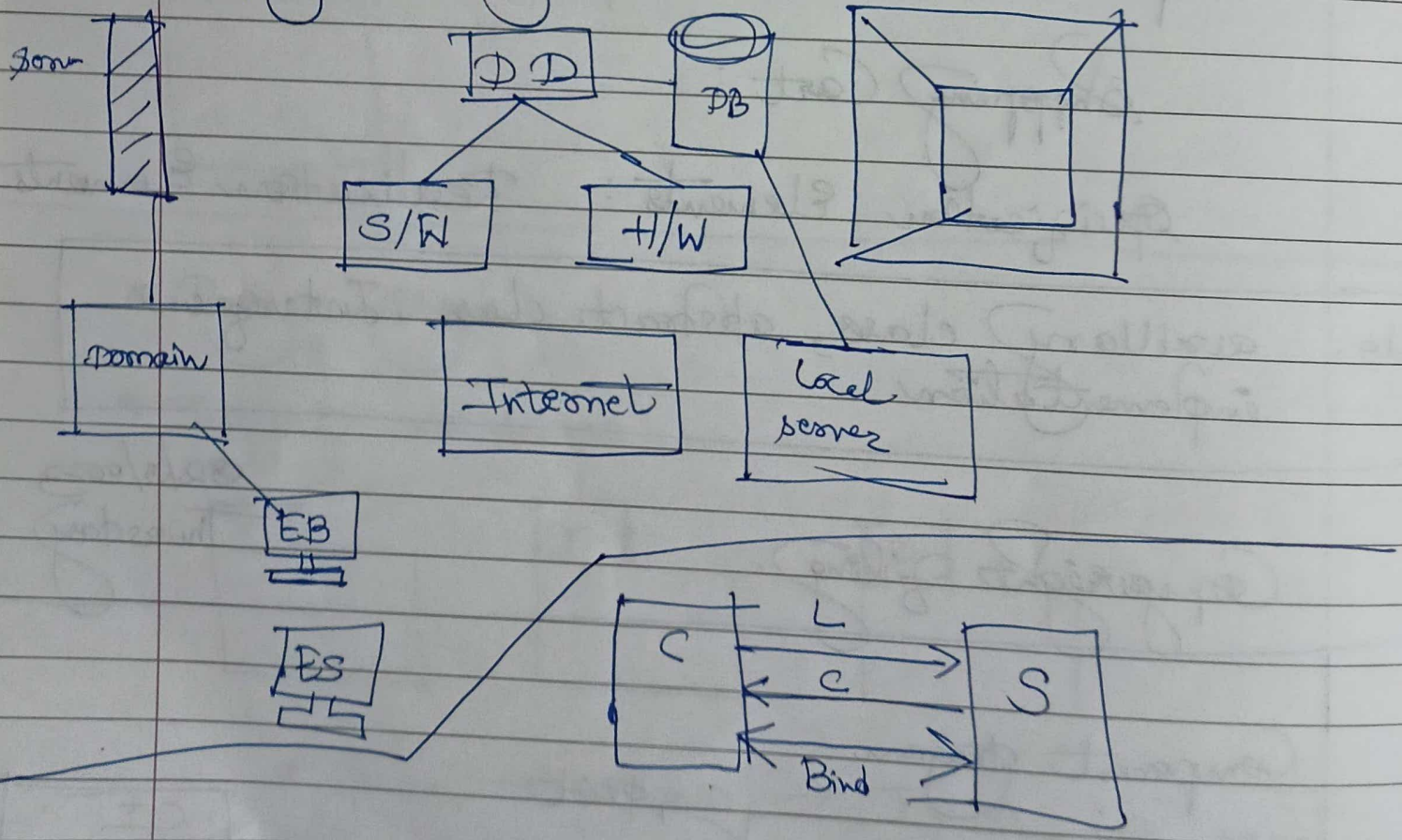
PAGE NO.:
DATE : / /

## * Composite Structure Diagram




CSD

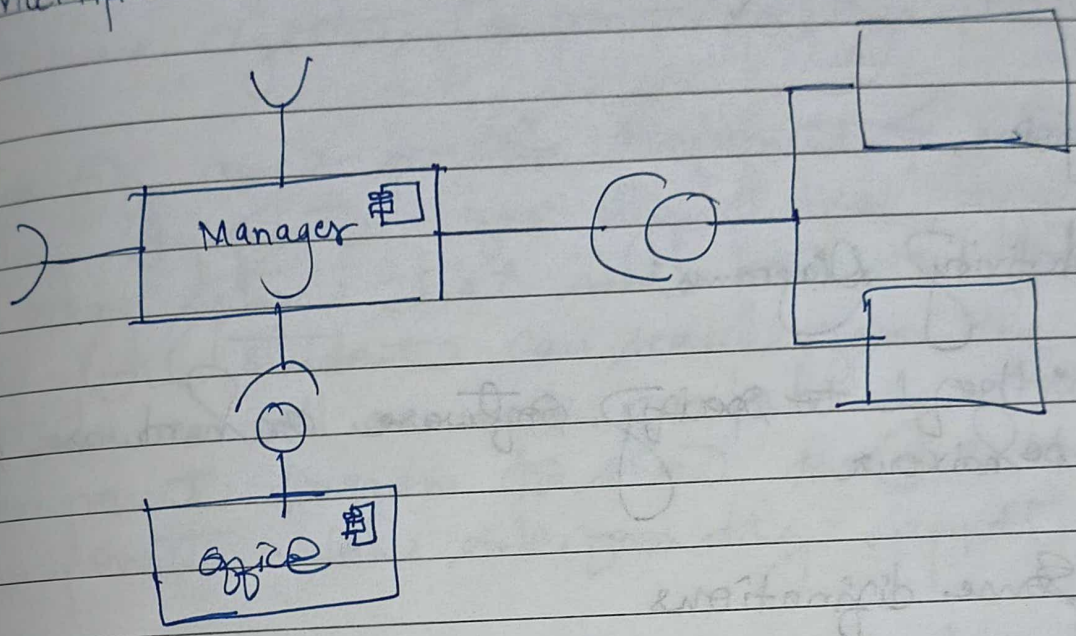
## * Deployment Diagram :



PAGE NO.:
DATE: / /

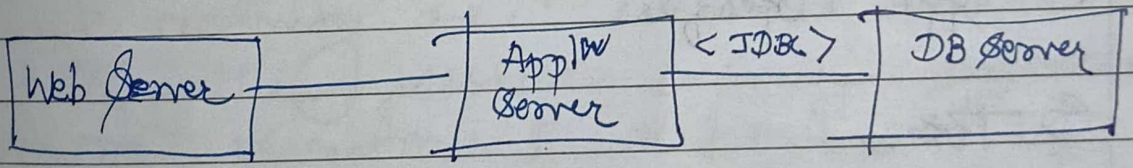
SEMANTICS:

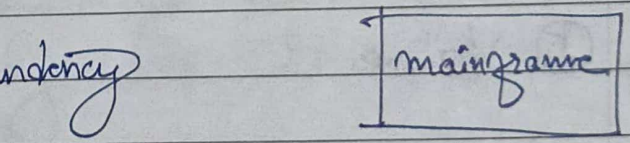
Multiple connections directed from




Deployment Diagram:

the run-time configuration of hardware nodes and the software components that run on those nodes.




<<manifestation>> dependency

* Unit 3: Dynamic modeling: Activity Diagrams in UML 2.0

Static diagrams are one that does not change as operation done on system.

Dynamic Modeling is one changable according to

# UNIT-3

PAGE NO.:
DATE: / /

## Dynamic Modelling : activity diagrams.

* Fork  one to many

* join,

* Activity Diagrams:
  - Useful to specify software or hardware system behaviour.

* Some difinations
  1. Flow : permits the interaction between two nodes of activity diagram.
  2. State : Condition or situation in life.
  3. Type : Specifies
  4. Token :
  5. Value :

* Actions : Act 1()  Act 2()
  Executable Link

* Activity Nodes:

  Action Nodes  Control Nodes  Object Nodes