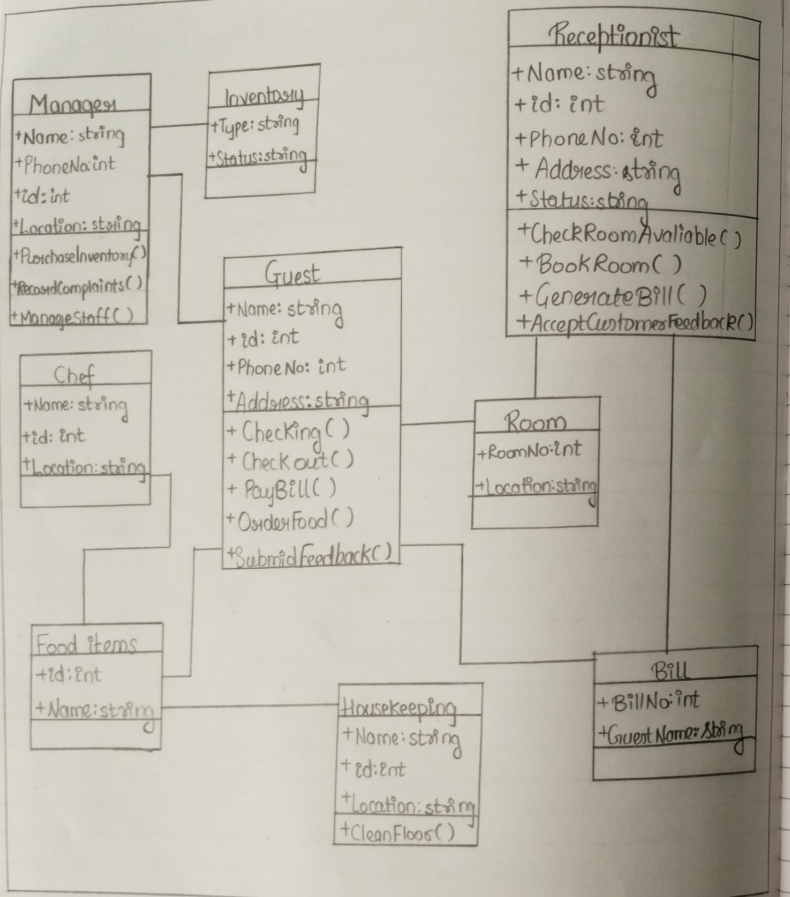


EXPERIMENT : No. Practical 1

DATE 1

AIM: Study and implementation of Class Diagrams.

Solution:

A class diagram is a static representation of a software system, illustrating the types of objects and their relationships. It highlights classes, attributes, functions and their associations. It is a structural diagram encompassing classes, interfaces, associations, collaborations and constraints.

Components

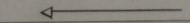
Class: Represents a blueprint for objects, containing attributes and methods.
Examples of classes are passengers, planes or tickets.

Class
Attribute

Attribute: It represents a characteristics or properties of a class. Characteristics of interest of a passenger, for example, are name and age.

Class
Attribute

Generalization: It is a relationship between two classes: a general class and a special class:



Teacher's Sign.: _____

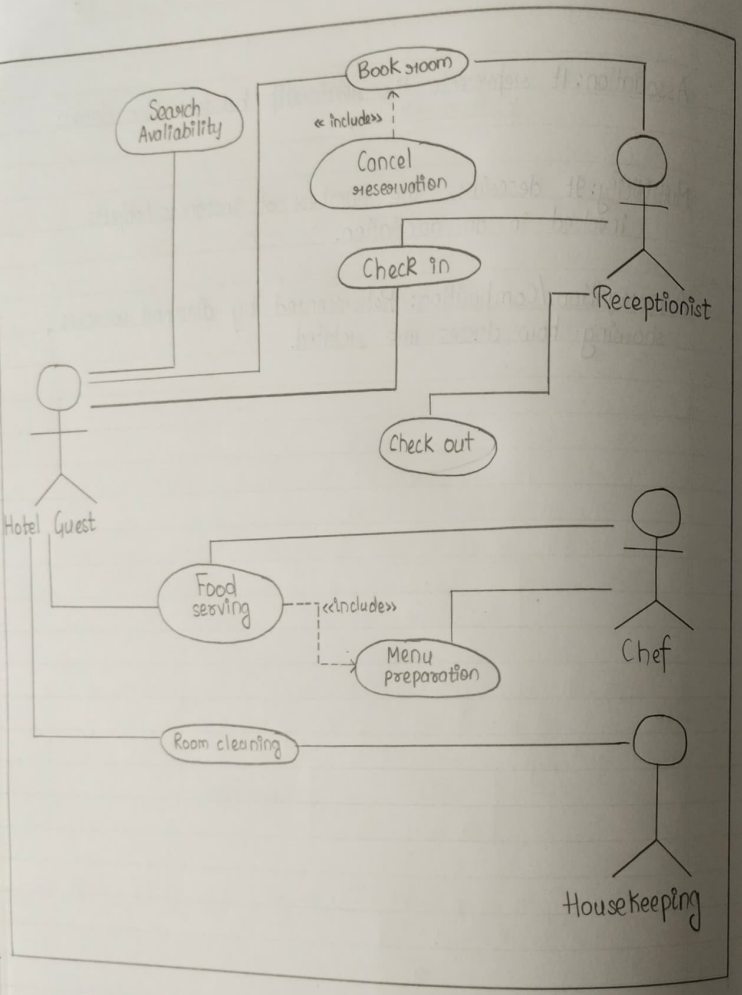
EXPERIMENT : No. _____ DATE 2

Association: It represents the relationship between two classes:
is flown with →

Multiplicity: It describes the number of instances / objects involved in an association.

Aggregation / Composition: Represented by diamond arrows, showing how classes are related.

Teacher's Sign.: _____



EXPERIMENT : No. Practical 2

DATE 3

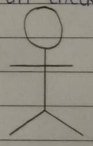
AIM: Study and implementation of Use Case Diagram

Solution:

A use case diagram is a graphic depiction of interactions among elements of a system. A use case is a methodology used in system analysis to identify, clasify and organize system requirements. The actors, usually individuals involved with system define according to their roles.

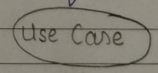
Components

Actors: Actors are external entities that interact with the system. They can be users, other systems, or even hardware devices. Example, the role of an check-in employee.



Actor

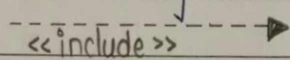
Use Case: Use case represents the various ways that an actor interacts with the system. They describe a specific functionality or a set of related functionalities.



Teacher's Sign.:

Association: It represents a communication or interaction between an actor and a use case. This relationship signifies that the actor is involved in the functionality described by the use case.
It is depicted by a line connecting the actor to the use case:

Include: Indicates that a use case includes the functionality of another use case. This relationship promotes modular and reusable design. It is denoted by a dashed arrow pointed from including use case to the included use case:



EXPERIMENT : No. Practical 3 DATE 5

AIM: Study and implementation of Entity Relationship(ER) Diagram.

Solution:

An entity relationship model, also called an Entity Relationship(ER) diagram, is a graphical representation of entities and their relationship to each other, typically used in regard to the organization of data within databases or information system.

Components

Entity: A definable thing, such as a person, object, concept or event, that can have data store about it. Eg, a car, student or customer. a things

Entity type: A group of definable things, Eg, cars, students, customers.

Entity Set: An Entity is object of Entity Type and a set of all entities is called an Entity Set.

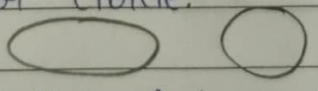
Entity categories: Entities are categorize as strong, weak or associative.

Entity keys: Refers to an attribute that uniquely defines an entity is an entity set. It can be super, candidate or primary.

Relationship: Describes how entities are related to each other.

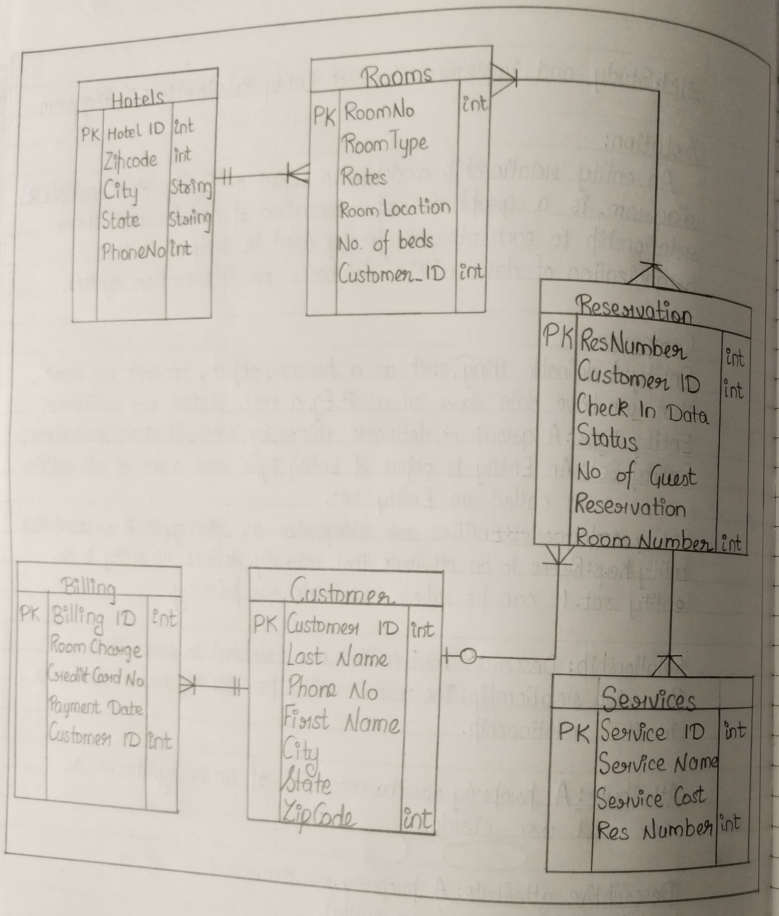
Recursive relationship: The same entity participates more than once in the relationship.

Attribute: A property or characteristics of an entity. Shown as an oval or circle.



Descriptive attribute: A property or characteristic of a relationship (versus of an entity).

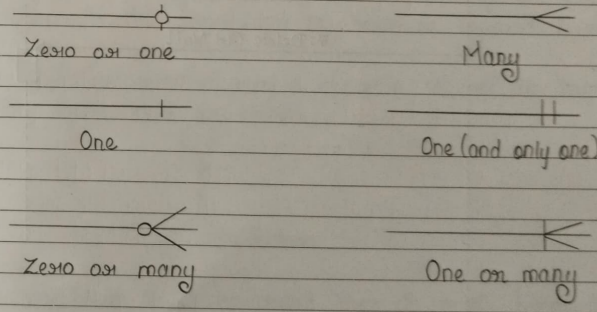
Teacher's Sign.: _____



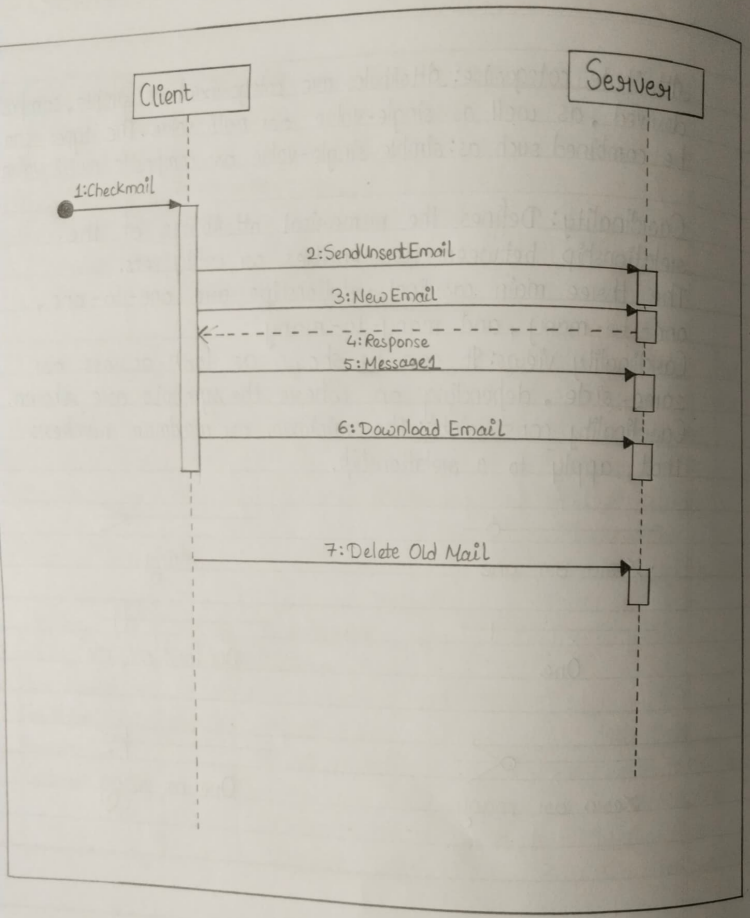
EXPERIMENT : No. _____ DATE 6

Attribute categorise: Attributes are categorized as simple, composite, derived, as well as single-value or multi-value. The types can be combined such as: simple single-value or composite multi-value.

Cardinality: Defines the numerical attributes of the relationship between two entities or entity sets.
The three main cardinal relationships are one-to-one, one-to-many, and many-to-many.
Cardinality views: It can be shown as look-across or same-side, depending on where the symbols are shown.
Cardinality constraints: The minimum or maximum numbers that apply to a relationship.



Teacher's Sign.: _____



EXPERIMENT : No. Practical 4

DATE 7

AIM: Study and implementation of Sequence Diagram.

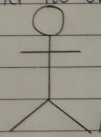
Solution:

A sequence diagram represents the flow of messages in the system and also termed as an event diagram. It helps in representing several dynamic scenarios. It play the role in communication between any two events that are part of the run time.

Components

Comments: The flow of a mutation event is documented with a combination of textual description and a sequence diagram. In comments, the flow logic is shown on topmost level.

Actor: It represents a type of role where it interacts with system and its objects.



Actor

Mutation Event: It is an event that is sent from the use case, so normally from the user interface, to the IT system. The goal of the event is to mutate information in the IT system, meaning to create, change or delete something.

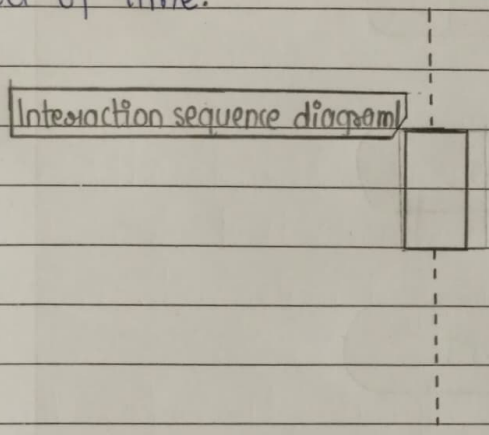
«M» Event /

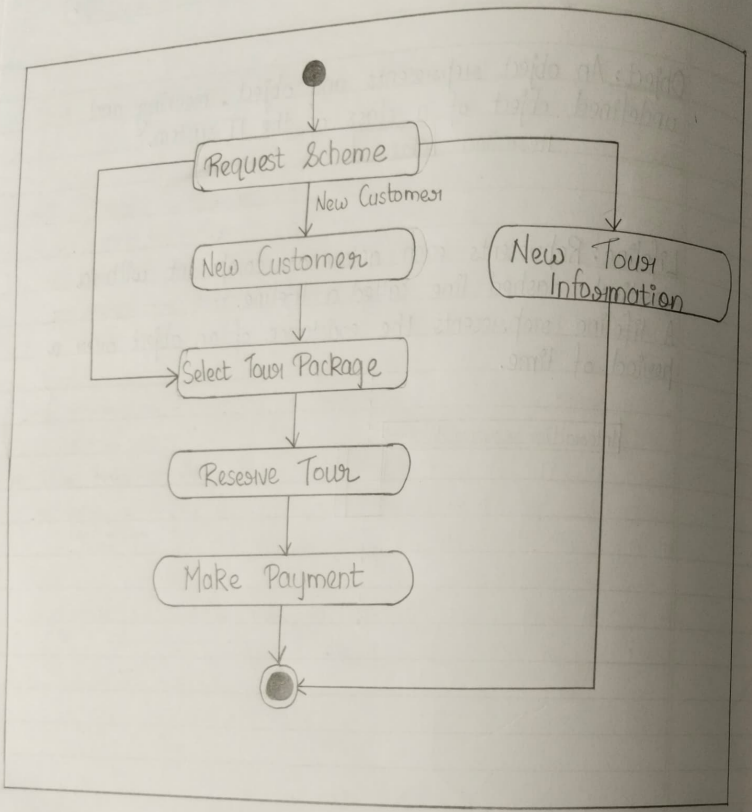
Teacher's Sign.: _____

Object: An object represents any object, meaning and undefined object of a class of the IT system.

Iteration [Class]

Lifeline: Represents each actor or component with a vertical dashed line called a lifeline.
A lifeline represents the existance of an object over a period of time.





EXPERIMENT : No. Practical 5

DATE 09 02 24 (9)

AIM: Study and implementation of state Transition Diagrams.

Solution:

A state diagram is a diagram used in computer science to describe the behaviour of a system considering all the possible states of an object when an event occurs. This behaviour is represented and analyzed in a series of events that occur in one or more possible states.

Components

Initial State: Represents the starting point for all object:
Objects in this state do not yet exist.

State: Represents a condition or situation during the lifetime of an object, determine by its attributes and associations. Not every attribute modification leads to a new state.

State

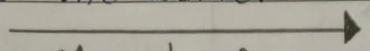
Transition: Represents a change from one state to another.

Internal Transition: A transition from one state to itself. The object handles the event without changing its state.

State
<<M>> event/

Teacher's Sign.:

Mutation Event: Initiates a transition from one state to another, or for an internal transition where the state remains the same.



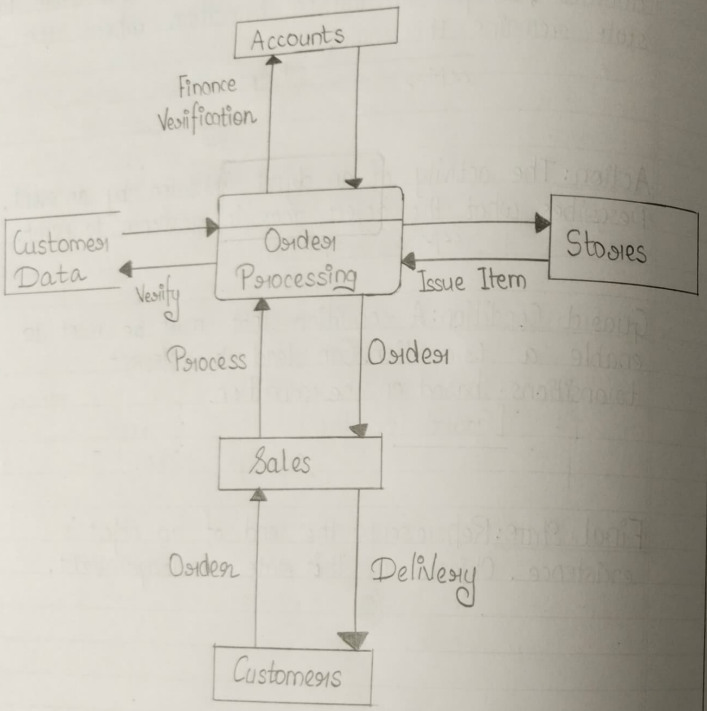
Action: The activity of an object initiated by an event. Describes what the object does in response to event.

<<M>> Event/Action

Guard Condition: A condition that must be meet to enable a transition. Can lead to different transitions based on the condition.

[Guard Condition]

Final State: Represents the end of an object's existence. Objects in this state no longer exist.



EXPERIMENT : No. Practical 6

DATE 09 02 24

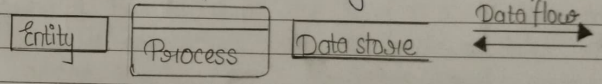
AIM: Study and implementation of Data Flow Diagrams

Solution:

A data flow diagram (or DFD) is a graphical representation of the flow of data through an information system. It shows how information is input to and output from the system, the sources and destinations of that information, and where that information is stored.

Components

DFD can represent Source, destination, storage and flow of data using the following set of components –



Entities - Represent the sources and destinations of data within the system. They are typically represented by rectangles with their respective names.

Process - Represents activities and actions taken on the data within the system. They are usually represented by circles or rounded rectangles.

Data Storage: Represents where data is stored within the system. There are two variants: either as a rectangle with both smaller sides absent or as an open-sided

Teacher's Sign.:

rectangle with one side missing.

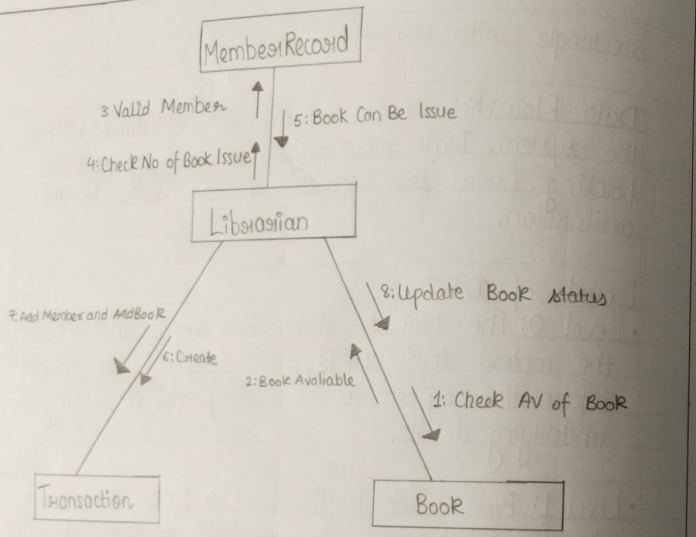
Data Flow: Represents the moment of data within the system. Data movement is shown by arrows pointing from the source of the data to its destination.

Levels of DFD:

- Level 0: The highest abstraction level, also known as the context level DFD, which depicts the entire information system as one diagram, concealing underlying details.

- Level 1: Breaks down the level 0 DFD into more specific components, showing basic modules, data flow among modules, basic processes, and sources of information.

- Level 2: Provides a deeper understanding of how data flows within the modules mentioned in Level 1. Higher-level DFDs can be broken down into more specific lower-level DFDs until the desired level of specification is achieved.



EXPERIMENT : No. Practical 7

DATE 23 02 24

AIM: Study and implementation of Collaborsation Diagrams.

Solution:

Collaborsation diagrsam, also called a communication diagrsam or intersaction diagrsam, is an illustrsation of the srelationships and intersactions among software objects in Unified Modeling Language (UML).

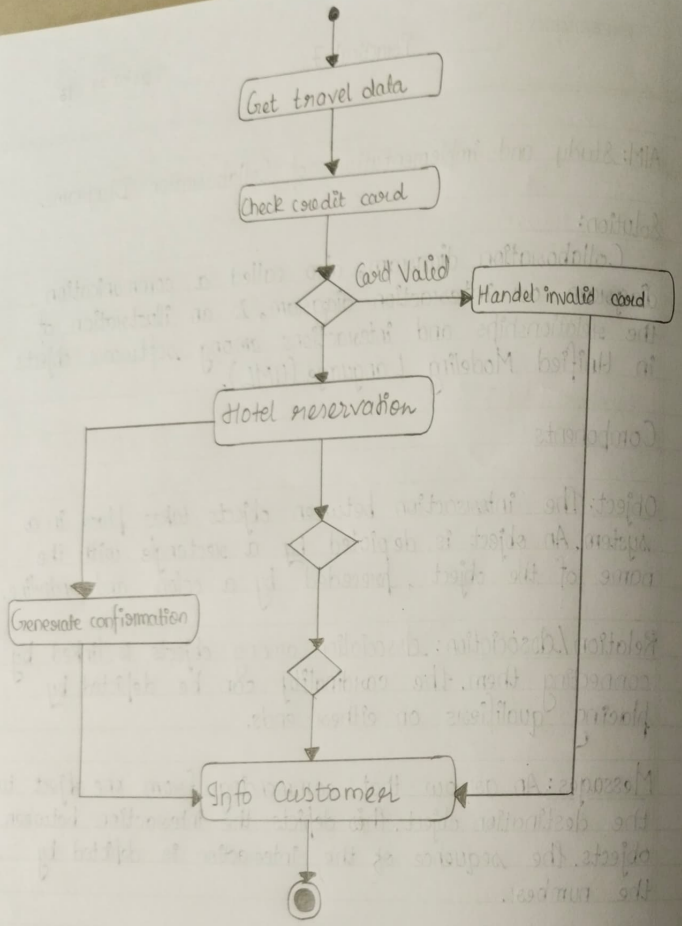
Components

Object: The intersaction between objects takes place in a system. An object is depicted by a srectangle with the name of the object, preceded by a colon and underline.

Relation/Association: Association among objects is linked by connecting them. The cardinality can be depicted by placing qualifiesrs on either ends.

Messages: An arsrow that commencing from one object to the destination object. This depicts the intersaction between objects. The sequence of the intersaction is depicted by the numbesr.

Teacher's Sign.:



EXPERIMENT : No. Practical 8 DATE 14

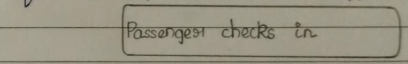
AIM: Study and implementation of Activity Diagrams.

Solution:

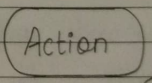
Activity diagram is another important in UML to describe the dynamic aspects of the system. It is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

Components

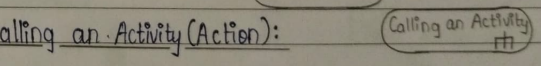
Activity: An activity diagram represents a single business process. It consists of actions and control elements like decisions and merges connected by activity edges forming control flow. Multiple actions can happen in parallel.



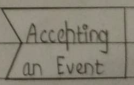
Action: These are individual steps in the process, like doing a calculation or waiting for something to happen. Actions can take in information, do something with it, and produce new information.



Calling an Activity (Action):



Accepting an Event (Action):



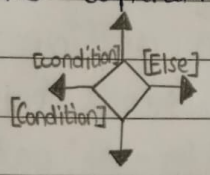
Teacher's Sign.:

Accepting a Time Event (Action): End of month Occurred

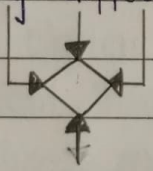
Sending Signals (Action): Sending Signals

Edge (Control Flow): This is the path that the process follows, shown by arrows connecting the actions. Its like following a trail from one action to the next.

Decision Node: These are places where the process can go in different directions depending on conditions.



Merge Node: These bring different path back together after they split.



Initial & Final Node: These show where the process starts and ends.

Initial Node

Final Node

EXPERIMENT : No. Practical 9 DATE 16

AIM: Study and implementation of Component Diagram.

Solution:

Component diagram shows components, provided and required interfaces, ports, and relationships between them. This type of diagram is used in Component-Based Development (CBD) to describe systems with Service-oriented Architecture (SOA).

Components

Component: Think of component like a specialized tool in a toolbox. Its something that performs a specific function or task within a system. Components can talk to each other and share information.
In diagrams, they're often shown as blocks with symbols on them.

Node: They are like bigger machine or systems that house these components. They can be physical hardware like server or virtual environments like software platforms.
In diagrams, nodes are represented by boxes.

Interface: Show input or materials that a component either receives or provides. Interfaces can be represented with textual notes or symbols such as the lollipop, socket and ball-and-socket shapes.

Teacher's Sign.: _____

Port: Symbolized with a small square, it specify a seperate interaction point between the component and the environment.

Package: Packages groups multiple elements of the system together. These are represented by file folders in lucidchart. Just as file folders group together multiple sheets, packages can be drawn around several components.

Dependency: Shows that one part of your system depends on another.
These are represented by dashed lines linking one component (or element) to another.

EXPERIMENT: No. Practical 10 DATE 18

AIM: Study and implementation of Deployment Diagrams.

Solution:

Deployment diagram is a structure diagram which shows architecture of the system as deployment (distribution) of software artifacts to deployment targets. Artifacts represent concrete elements in the physical world that are the result of a development process.

Components

Artifact: A product develop by the software symbolized by a rectangle with the name and the word "artifact" enclosed by double arrows.

Association: A line that indicates a message or other type of communication between nodes.

Component: A rectangle with two tabs that indicates a software element.

Dependency: A dashed line that ends in a arrow, which indicates that one node or component is dependent on ~~athother~~ another.

Node: A hardware or software object, shown by a three-dimensional box.

Teacher's Sign.:

Node as container: A node that contains another node inside of it, such as that the nodes contain components.

Interface: A circle that indicates a contractual relationship; those objects that relize the interface must complete some sort of obligation.

Stereotype: A device contained within the node, presented at the top of the node, with name bracketed by double arrows.