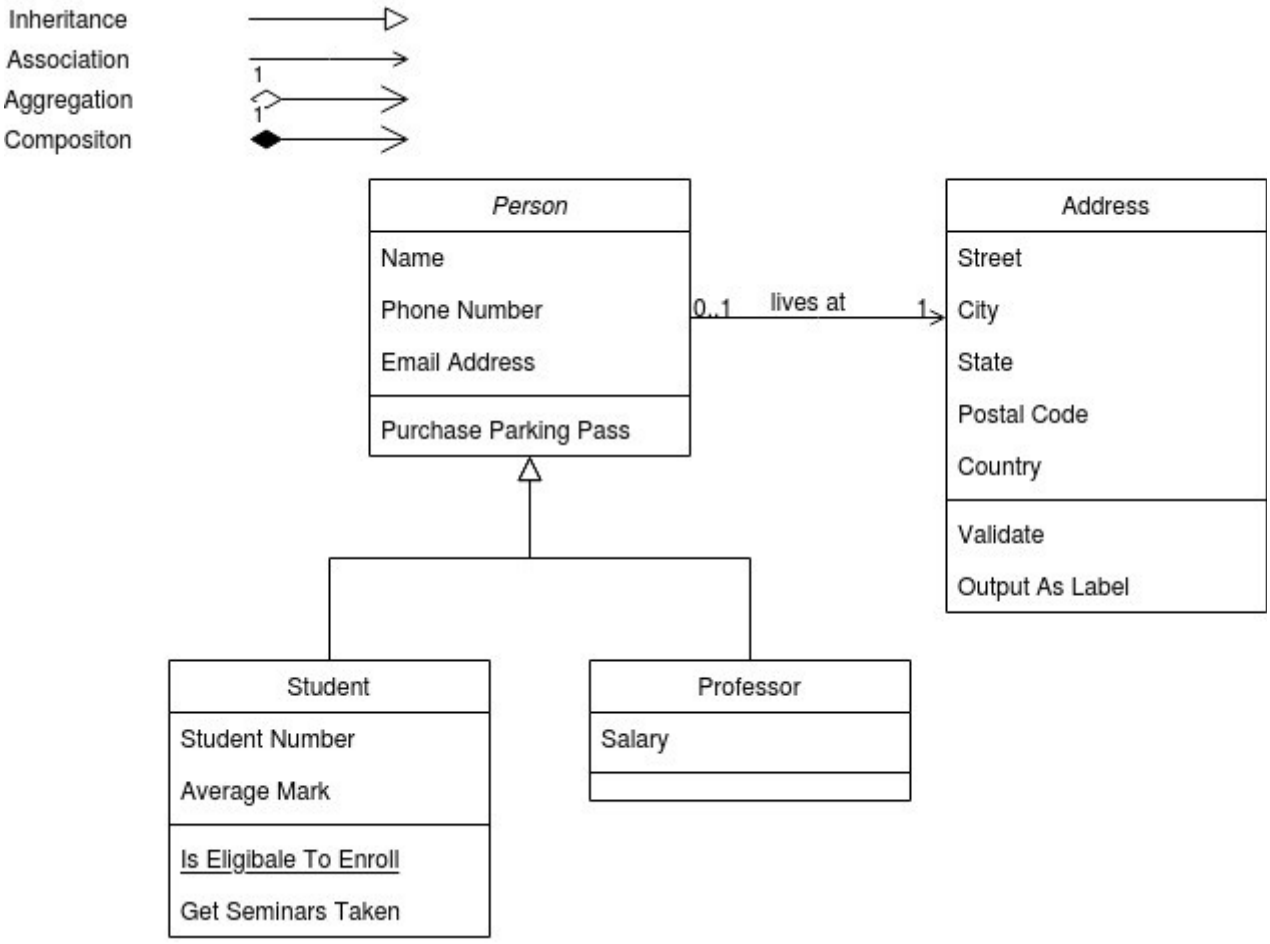
Class notation:

Class diagram is used to construct and visualize the objects oriented system.
It is a blueprint of an object
It encapsulates attributes and methods together. That we call it as class.
eg. Class name: person, Address, Student, Professor.

- Class name appear in first partition
- attributes of class appear in second partition
- methods/operations appear in third partition. This are the services that class provides. Return type of method shown after the colon following the parameter name.
- eg. purchasePass(userid:Int):bool
- here purchasePass is a method with one parameter userid of type int and returns the bool value
- Class visibility shown using +(public), -(private), #(protected)
- Diagram can be interpreted from various perespective.
- 1)Conceptual:Represent concept in domain
- 2)Specification: focus is on interfaces of adt in s/w.
- 3)Implementation: describe how classes will implement interfaces.

Relationship between classes are association, Inheritance, aggregation, composition etc.


Inheritance
Association
1
Aggregation
1
Compositon
Person
Name
Phone Number
Email Address
Purchase Parking Pass
0..1
lives at
1
Address
Street
City
State
Postal Code
Country
Validate
Output As Label
Student
Student Number
Average Mark
Is Eligibale To Enroll
Get Seminars Taken
Professor
Salary

Object notation:
The object is represented same way as class. The only difference is the name of object represented using underlined name.
It is actual implementation of class which is also known as instance of class.

- Object name appear in first partition
- attributes of object appear in second partition
- methods/operations appear in third partition.

It has same usage like a class.

Interface Notation:

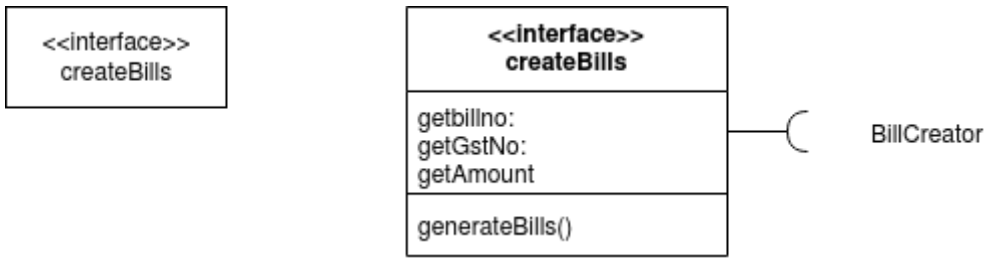
interface is a classifier that declares a set of coherent public features and obligations.

It speifies contract Any instance of a classifier that realizes (implements) the interface must fulfill that contract and thus provides services desscribed by contract.

This are declarations and cannot create instance of this.

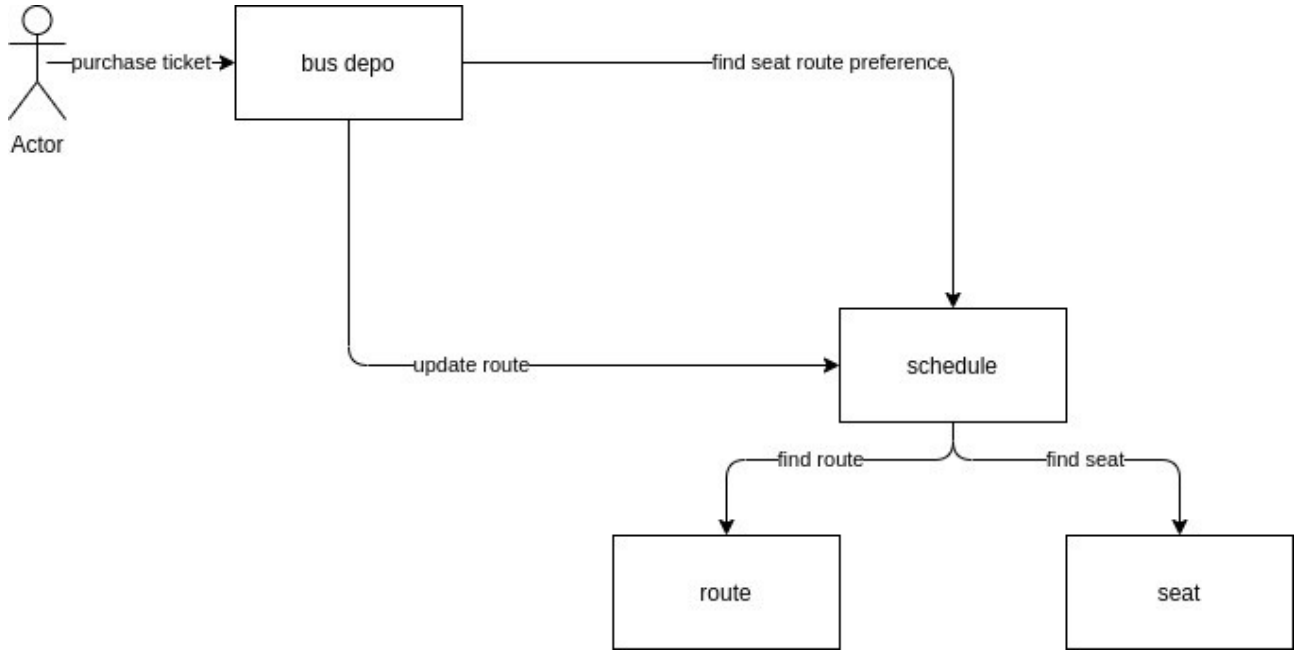
An instance can be shown using rectangle or circle precedding the name.

The obliagtions may associated withinterface in the form of constraints like precondition postcondition protocol specification



Collaboration notation:

It is a collection of named object and actors with a link connecting them. They collaborate on performing some task.
A Collaboration defines a set of participants and relationships that are meaningful for a given set of purposes
a collaboration diagram shows the relationships among the objects
This contains objects, Actors, Links and some messages.
Objects shown using rectangle with name inside it. Naming convention follows is objName:className
Actors are instances that invoke the interaction in diagram. Each actor has role, name which initialize the whole use case.
Links connects the object with actors with solid line between two elements
Messages between objects are shown as a labelled arrow placed near link
This can be use when...
1) modelling collaboration, mechanism or structural organization within system
2) Providing overview of objects within an object oriented system..
3)Demonstrate reverse and forward engg.
4) Visualizing the logic behind an operation.



Dynamic behavior of object can in addition of sequence diagram also represented using collaboration notation.

Transoformation from sequence diagram into collaboration diagram is bi directional
It emphasizes more the structure than the sequence of interactions
in above example purchase ticket, find seat route preference, update route, find route, find seat are the messages,
bus dept schedule, route, seat are the objects in the flow, and user is an actor

Use case notation:

This are the descriptions of the functionality of a system from user perespective.

It depict the behavior of the system. As it appears to the outside user it tells about the functionality of the system.

Relationship between the actors that use the system, the relationship between different use cases.

It tells the scope of the system. Because of this developer understand the user requirements

Use case models are developed at differenct level of abstractions.

System, system components, or a class.

It is iterative and incremental process.

If user requirement change, changes have to be done in all affected areas in document.

Use case is a description of set of interactions between user and the system.

Components use here are

1) actor:

actor is a someone that must interact with the system.

It can be hunan or automated system.

This are not part of system

it has 2 types-1)Primary actor 2)Secondary actor.

Primary actor: acts on the system.

Initiate interaction with the system

uses system to fulfil his goals

Secondary actor: it act on invoked by system.

Helps system to fulfil his goals

2)Use case:

it is a pattern or behavior of the system

the use cases are sequences of actions that the user takes on the system to get perticular target.

It is a dialogue between actor and system.

3) System boundry:

it is shown as rectangle

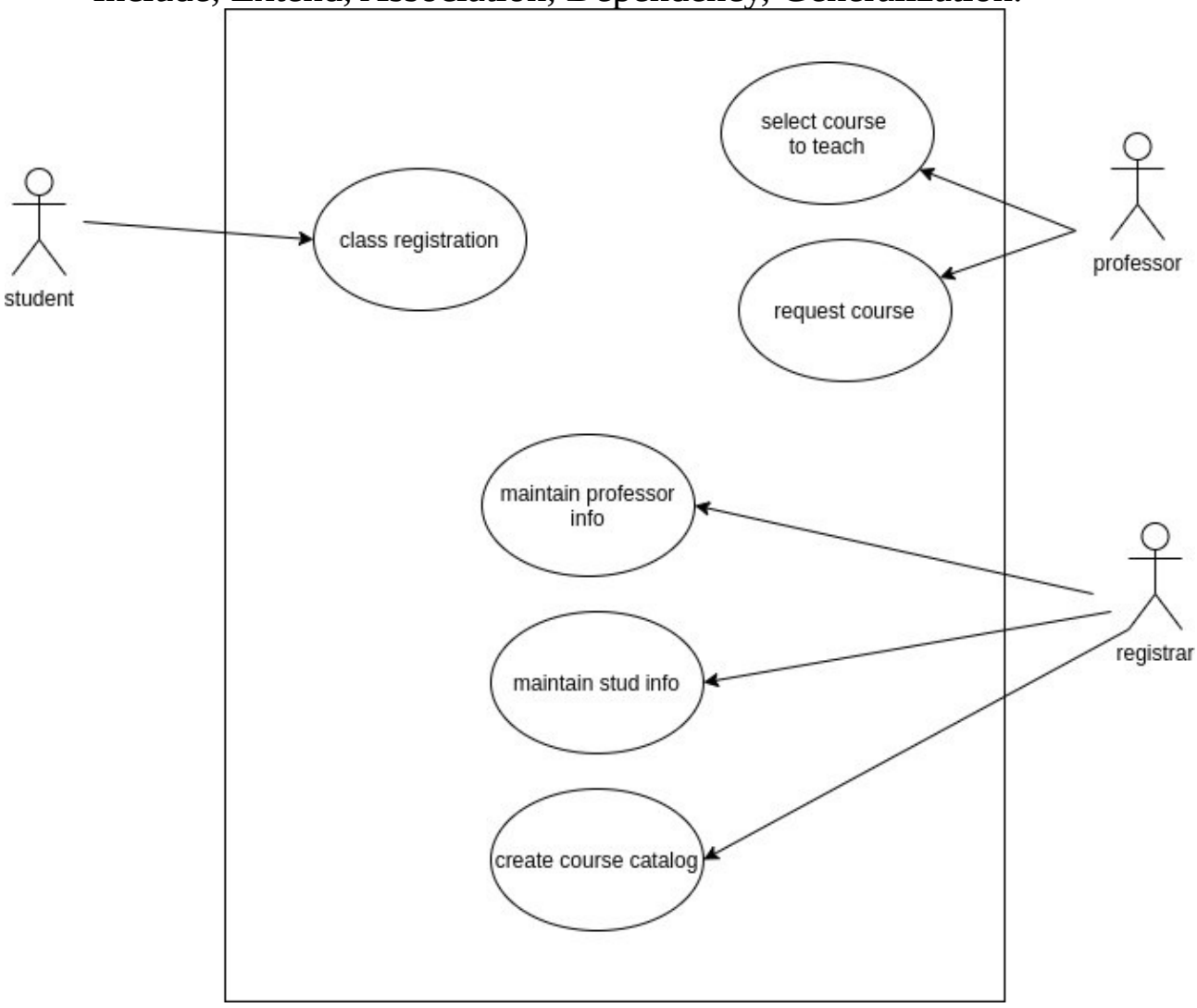
helps to identify what is external vs what is internal and what responsibilities of system are

4)Relationship:

it is an association between use case and actor.

There are several use cases.

Include, Extend, Association, Dependency, Generalization.



Actor notation:
actor is a someone that must interact with the system.
It can be hunan or automated system.
This are not part of system
it has 2 types-1)Primary actor 2)Secondary actor.
Primary actor: acts on the system.

Initiate interaction with the system

uses system to fulfil his goals

Secondary actor: it act on invoked by system.

Helps system to fulfil his goals

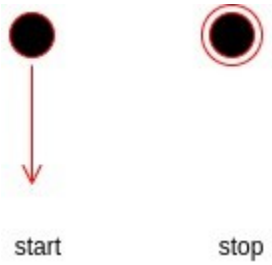
Initial state notation:

State mahine diagram is a behavior and used to specify doscrete system through finite state transitions.

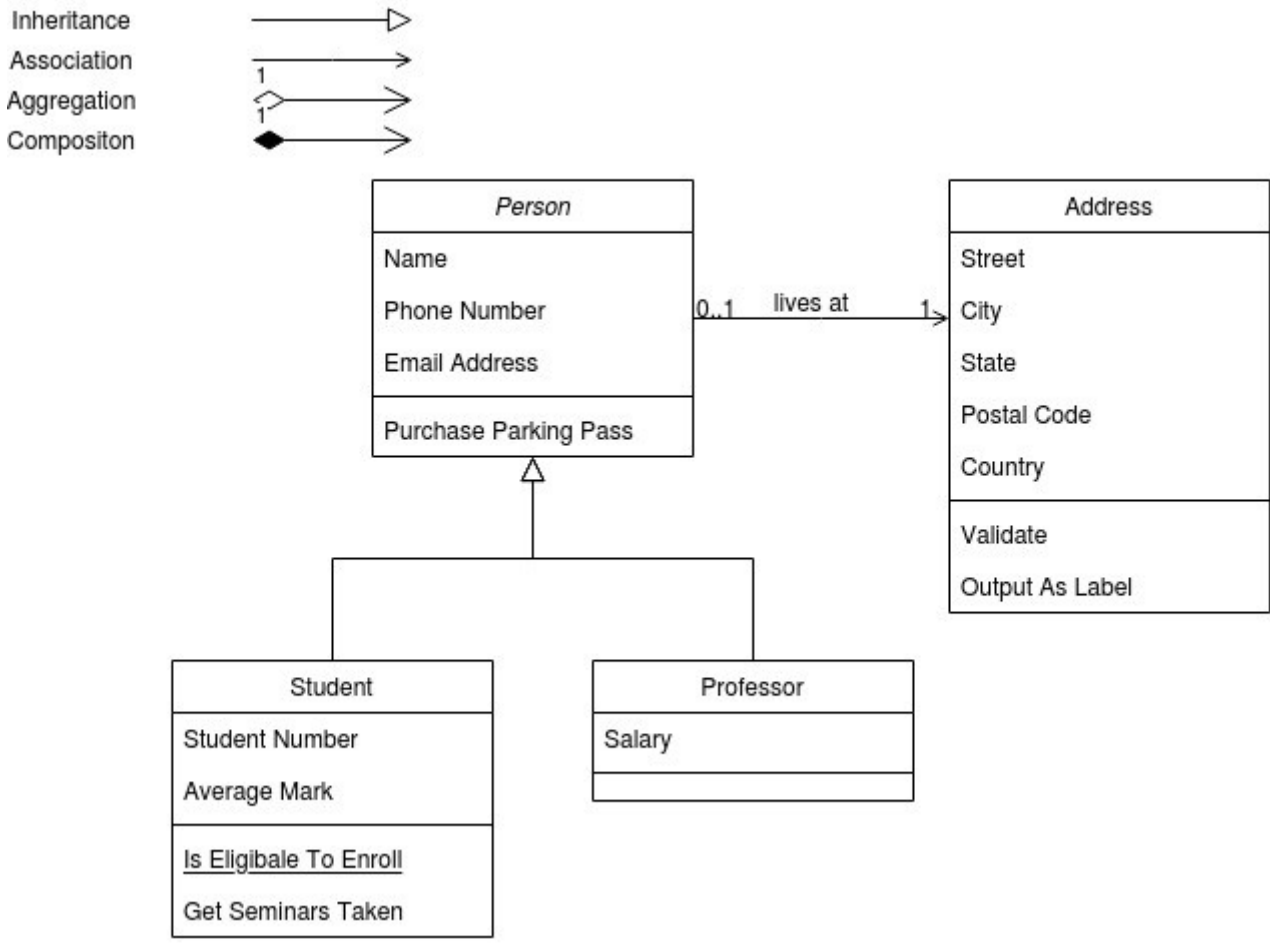
black filled circle is used to represent the initial state of system or a class

Final state notation:

we use filled circle within the another circle notation to represent the final state of state machine diagram



Active class notation:
Active class is look similer to a class with solid border.
It is used to describe the concurrent behavior of system.



Component notatioin:

in uml component notation used for modelling large systems into smaller subsystems which can be easily managed.

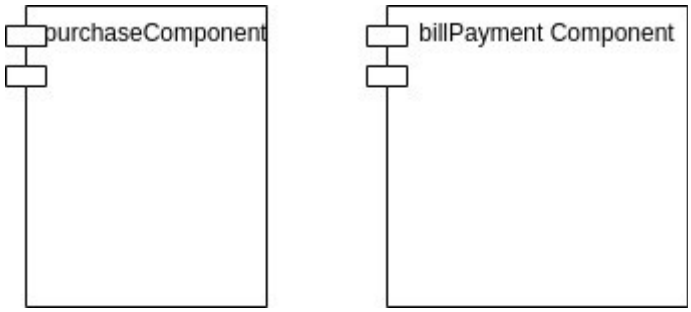
This are used at implementation phase

Component is replacable and executable piece of system whose implementaion details are hidden. Provides set of interfaces that a component realizes or implements.

It also requires interfaces to carry information.

It does not describe the functionality of the system but it describes the components used to make those functionalities.

component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc.



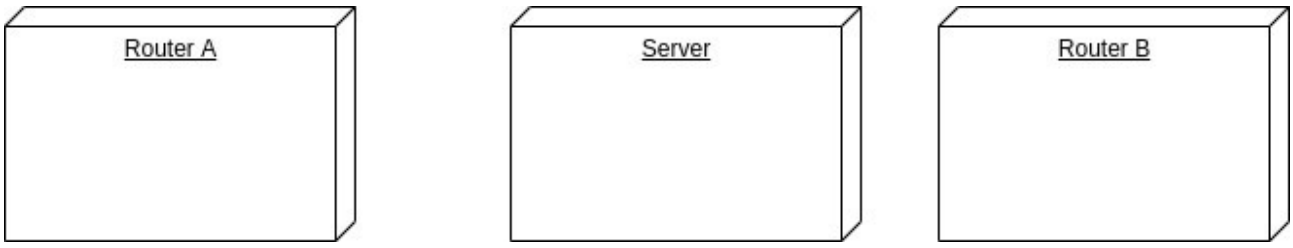
Benifits of components diagram

Imagine the system’s physical structure

system components and how they relate

emplasize the service behavior as it relates to interface

Node notation:
A node in UML is represented by a square box
A node represents the physical component of the system.
Node is used to represent the physical part of a system such as the server, network, etc.



Interaction notation:
Interaction diagram is nothing but a subset of behavioral diagrams. It is used to visualize the flow between various use case elements of a system. Interaction diagrams are used to show an interaction between two entities and how data flows within them.
Interaction diagrams in UML

1) Timing diagram
2) Sequence diagram
3) Collaboration diagram

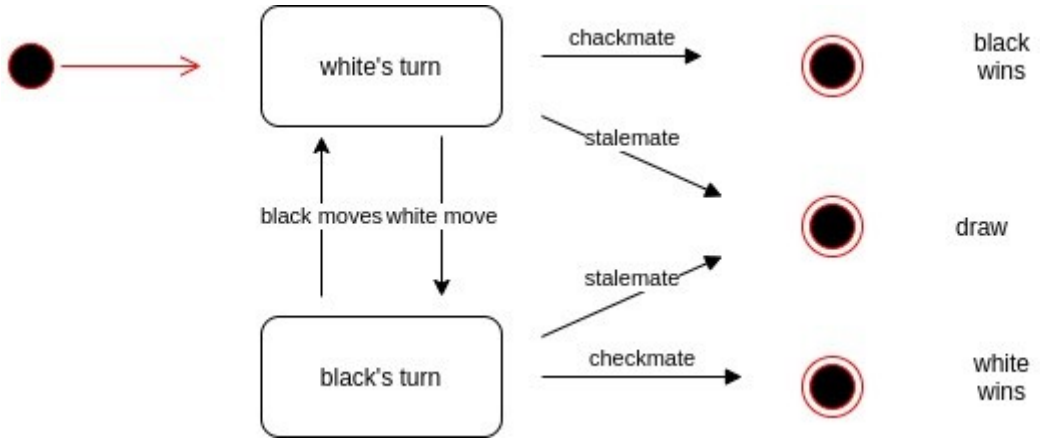
purpose of interaction diagrams is to visualize the interactive behavior of the system.
Visualizing the interaction is a difficult task. Hence, the solution is to use different types of models to capture the different aspects of the interaction.

To capture the dynamic behaviour of a system.
To describe the message flow in the system
To describe the interaction among objects.

Following things are to be identified clearly before drawing the interaction diagram

- Objects taking part in the interaction.
- Message flows among the objects.
- The sequence in which the messages are flowing.
- Object organization.

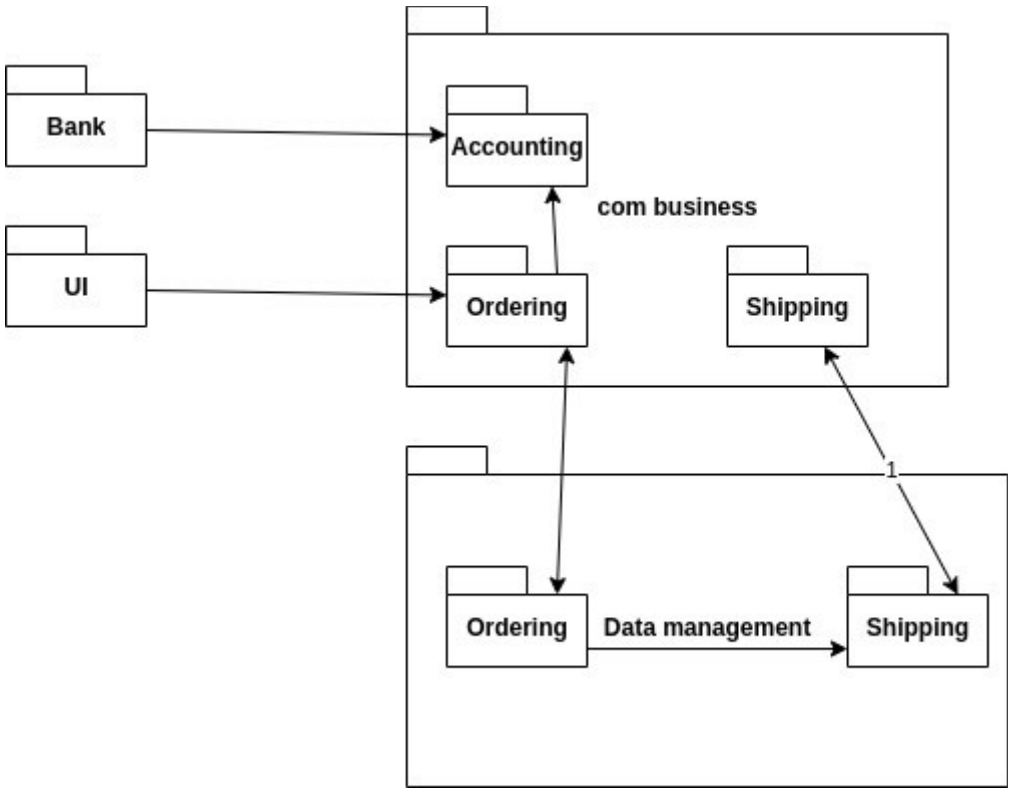
State machine notation:
It is a behavior diagram which shows discrete behavior of part of design system
through finite state transitions.
Used to expressusage protocol of system.
It defines the number of states it has. The events under which an object changes state.
Notations used in state machines : state, event, guard, trasition, action
Applications:
depicts event driven object inreactive system.
Illustrate use case scenarios in business contact
describing how object moves through various stages within lifetime.
Shows overall behavior of system.



Package notation:
Package diagramisUML structural diagram which shows pakages and dependencies between the packages.

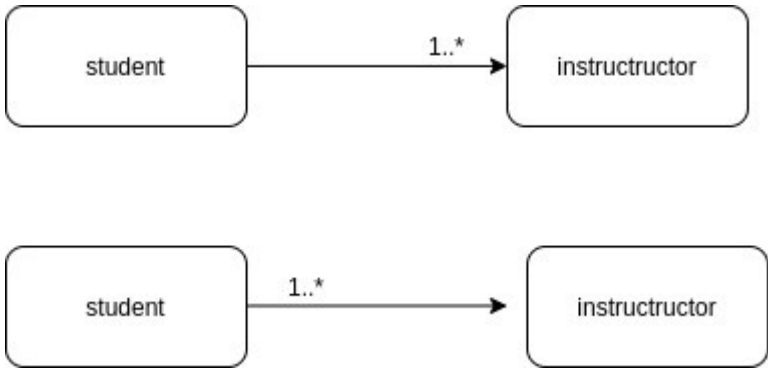
Package can be used as a template for other packages.
Packageable element can be used as a template parameter. A package template parameter may refer to any element owned or used by the package template, or templates nested within it.

Package diagram, a kind of structural diagram, shows the arrangement and organization of model elements in middle to large scale project. Package diagram can show both structure and dependencies between sub-systems or modules, showing different views of a system



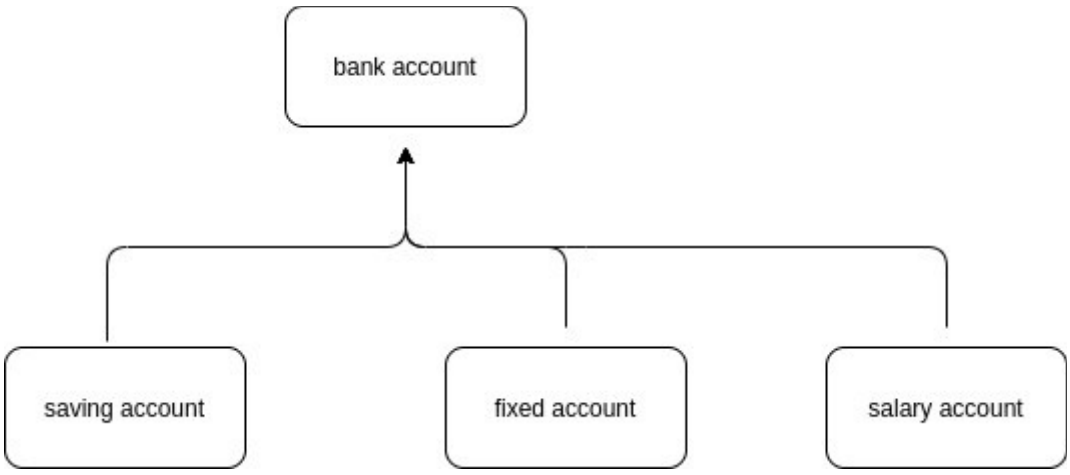
Association Notation:
If two classes in a model need to communicate with each other, there must be a link between them, and that can be represented by an association

We can indicate the multiplicity of an association by adding multiplicity adornments to the line denoting the association. The example indicates that a Student has one or more Instructors:



Generalization Notation:

A **generalization** is shown as a line with a hollow triangle as an arrowhead between the symbols representing the involved classifiers
The arrowhead points to the symbol representing the general classifier.
This notation is referred to as the **"separate target style."**
Generalization is the activity of identifying commonality among concepts and defining superclass (general concept) and subclass (specialized concept) relationships. It is a way to construct taxonomic classifications among concepts which are then illustrated in class hierarchies
Identifying a superclass and subclasses is of value in a domain model because their presence allows us to understand concepts in more general, refined and abstract terms



Extensibility Notation:
UML defines three extensibility mechanisms to allow modelers to add extensions without having to modify the underlying modeling language. These three mechanisms are:
1)sterotypes:

A *stereotype* extends the vocabulary of the UML, allowing you to create new kinds of building
2)Constraits:

A constraint is an extension of the semantics of a UML element, allowing you to add new rules or to modify existing ones.
A constraint specifies conditions that must be held true for the model to be well-formed. This notation can also be used to adorn a model element's basic notation, in order to visualize parts of an element's specification that have no graphical cue. For example, you can use constraint notation to provide some properties of associations, such as order and changeability
3)tagged values:

A tagged value extends the properties of a UML building block, allowing you to create new information in that element's specification. They are properties for specifying keyword-value pairs of model elements, where the keywords are attributes. They allow you to extend the properties of a UML building block so that you create new information in the specification of that element. Tagged Values can be defined for existing model elements, or for individual stereotypes so that everything with that stereotype has that tagged value. It is important to mention that a tagged value is not equal to a class attribute.
Graphically, a tagged value is rendered as a string enclosed by brackets, which is placed below the name of another model element. The string consists of a name (the tag), a separator (the symbol =), and a value (of the tag)