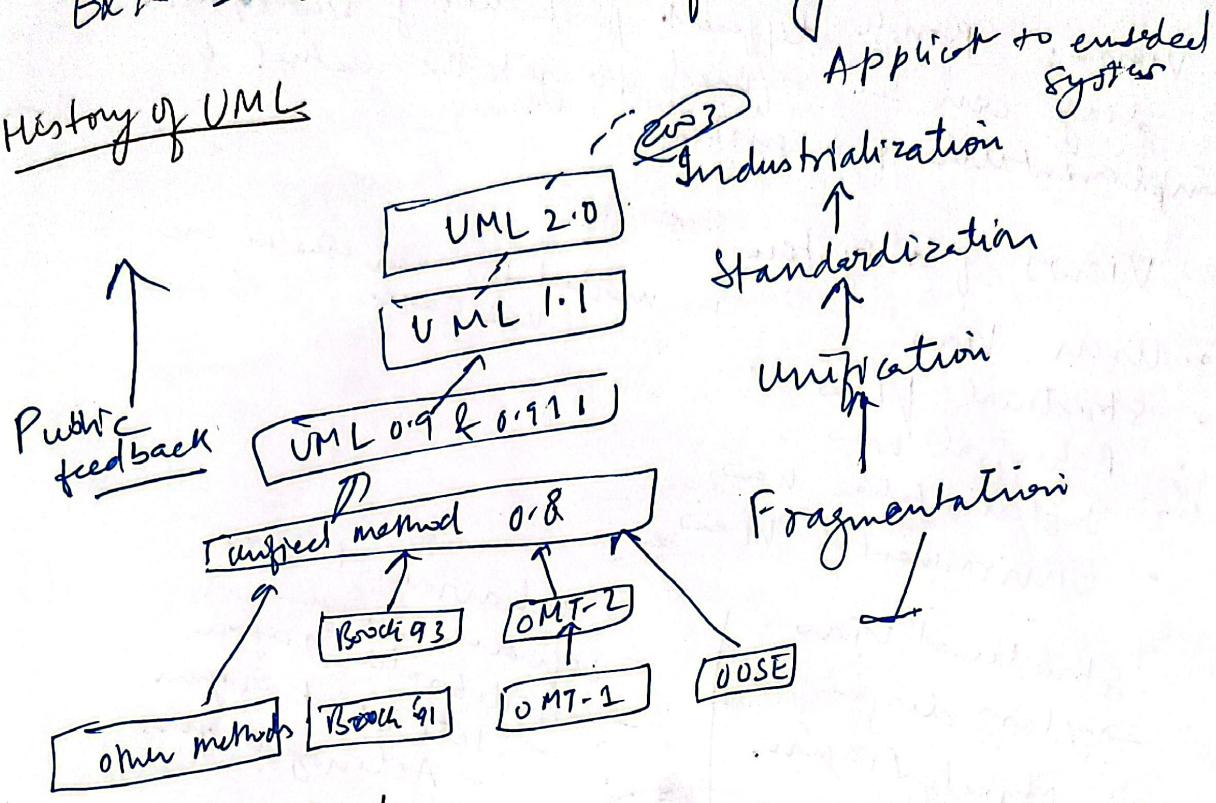
OOS

UML also being used outside S/w development area.
Ex:- Build to order manufacturing.

History of UML



Why we required?

- Modelling is an abstraction mechanism.
  - → capture only important aspect & ignore the rest
- → effective mechanism to handle complexity.
- UML is a graphical Modelling technique
  - Easy to understand & construct the model.

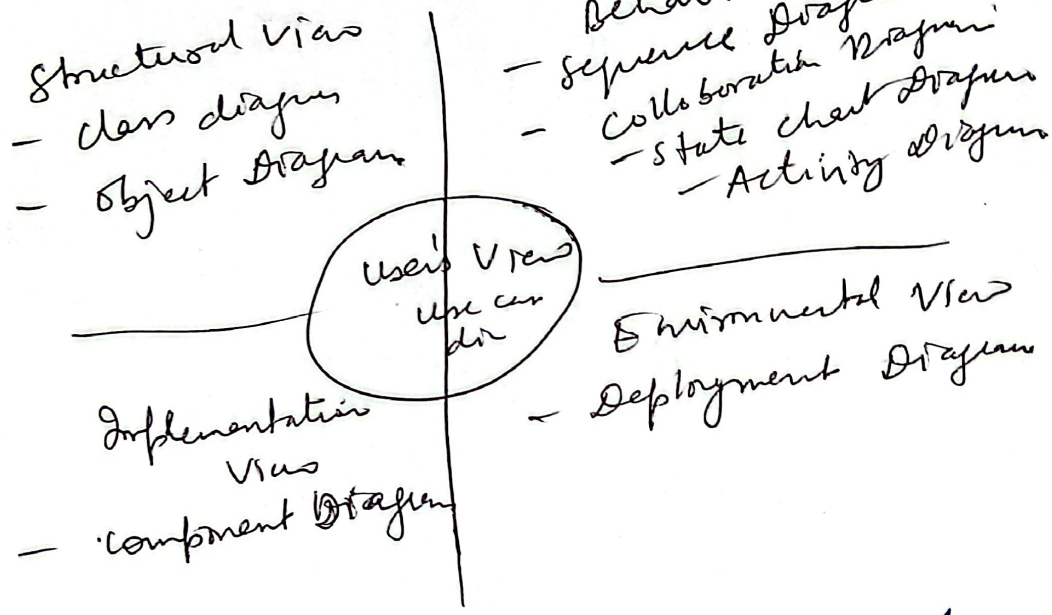
Modelling Vs Designing

- A design is a model of the system
  - But every model is not a design of the system
- From the requirements, our analysis model is created. (Analysis activity) not design model (code)
- Subsequently the analysis model is refined into the design model.

UML Diagrams

- 9 diagrams in UML 1.x:
  - used to capture 5 different Views of a system
- Views: Provide different perspectives of S/W systems
  - diag can be refined to get the actual implementation of a system.

* Views of a system
  - User's View — to model the user view we use use case
  - Structural View
  - Behavioral View
  - Implementation View
  - Environmental View



* Use Case Diagram :-
High level behaviour of the system, user goals, external entities : actors.

- Sequence Diagram
  - Focus on time ordering of messages
- collaboration Diagram
  - Focus on structural organization of objects & msgs
- state chart diagram
  - event driven state changes of system.
- Activity Diagram
  - flow of control b/w activities.

use to document O'O design UML

A case of use:- A way in which a system can be used by the users to achieve specific goals.

- Correspond to a high-level requirement
- Defines external behaviour without revealing internal structure of system.

Importance of Model

- Analyse the Problem-domain
  - Simplify reality
  - Capture requirement
  - Visualize the system in its entirely.
  - Specify the structure / behaviour of the system.

UML: Std lang for specifying, visualizing & constructing & document the s/w system

- Design the sol.
  document the sol - in terms of its structure, behaviour etc.

Principle of Modeling

(1) choose your model well:- the choice of model profoundly impacts the analysis of the problem & the design of the sol.

(2) Every model may be expressed at different level of precision:- The same model can be scaled up (or down) to different granualities.

(3) the best models are connected to reality:= simplify the model, but don't hide important details.

(4) No single model is sufficient:- a set of models is need to solve any non-trivial system

(1) Uses
It provide template that guides you in constructing a system.
Helps to understand complex system part by part

plan & Model create krte h
other lang - S/w ~~kar~~ delmkra-

Unified Modeling lang - pictorial base
lang use to make S/w blueprints

OOS design simplify krte h using pictorial view
& business user, any user.

UML Modelling Types

- It is very imp to distinguish b/w the UML model different diagrams are used for different types of UML modeling.
there are three important ~~types~~ of UML modeling.

~~(1) Structural Modeling~~

- UML is a std lang for specifying, visualizing, constructing & documenting the artifacts of S/w systems.
- UML created by the OMG (object mgt group) & UML 1.0 specification draft was proposed OMG in Jan 1997
- OMG is continuously making efforts to create truly Industry std.
- different from C++, Java, COB-2, etc S/w development
- It pictorial lang of blue print.

• Process flows in manufacturing unit. Example S/w design.

goals:-

1) define an general purpose modelling lang, for business users, common people, anybody interested in the S/w system.

Conceptual Model in UML

- A conceptual model can be defined as a model which is made of concepts & their relationships
- A conceptual model is the first step before drawing a UML diagram. It helps to understand the entities, in the real world & how they interact with each other.
- As UML describes the real-time systems, it is very important to make a conceptual model & then proceed gradually.

Elements of Conceptual Model of UML

- UML building blocks
- Rules to connect the building blocks
- common mechanisms of UML.

Role of UML in OOP

- UML is a modeling lang used to model S/w & non S/w systems.
- OOS transfer to UML design.
- ~~the oo design is~~ before understanding the UML in detail. the OO concepts should be learned properly
- once the OO analysis & design is done next is the i/p from OO analysis & design. the i/p to UML diagrams

UML building blocks

- the building blocks of UML can be defined as:
- Things
- Relationships
- Diagrams.

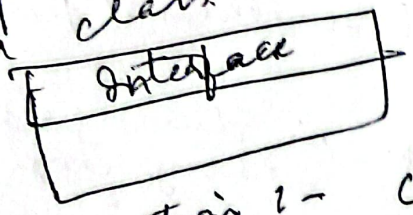
Things :- things are the most important building blocks of UML. things can be

- Structural
- Behavioural
- Grouping
- Annotational

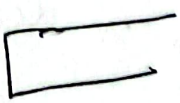
1) Structural :- Structural things define the static part of the model. they represent the physical & conceptual elements. following are the brief description of the structural things

class - class represents a set of objects having similar responsibilities.

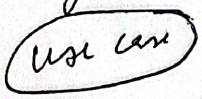
Interface :- Interface define a set of operation. which specify the responsibility of a class



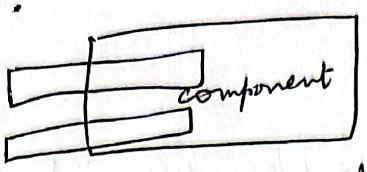
Collaboration :- Collaboration defines an interaction b/w elements.



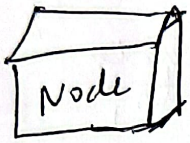
Use case - Use case represents a set of actions performed by a system for a specific goal.



Component - Component describes the physical parts of a system.



Node :- A node can be defined as a physical element that exists at run time.



Behavioral things

A behavioral thing consists of the dynamic parts of UML models. Following are the behavioral things -

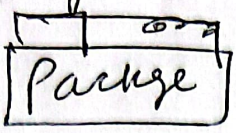
Interaction :- Interaction is defined as a behavior that consists of a group of message exchanged among elements to accomplish a specific task.

State M/c :- State M/c is useful when the state of an object in its life cycle is imp. It defines the sequence of states an objects through in response to events. events are external factors responsible for state change.

grouping things

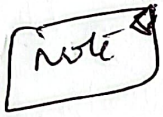
grouping things can be defined as a mechanism to group elements of a UML model together. There is only one grouping thing available

Package:- package is the only one grouping thing available for gathering structural & behavioral things.

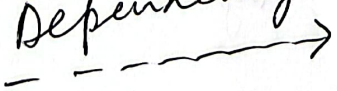


* Annotational things:-
Annotational things can be defined as a mechanism to capture remarks, description & comments of UML model elements.
Note:- It is the only one Annotational thing available.

- A note is used to render ~~comments~~ comments, ~~texts~~ constraints, etc. of a UML element



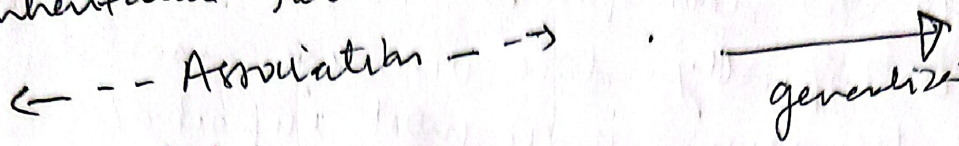
Relationship
Relationship is another most imp building block of UML. It shows how the elements are associated with each other & this association describe the functionality of an application.
There are 4 kinds of relationship available.
dependency.
Dependency is a relationship b/w two things in which change in the one element also affects the other.



Association:-
Association is basically a set of links that connects the elements of a UML model. It also describes how many objects are taking part in that relationship.

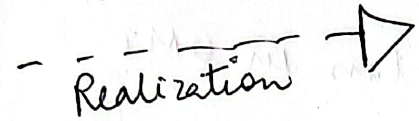
Generalization:-

Generalization can be defined as a relationship which connects a specialized element with a generalized element. It basically describes the inheritance relationship in the world of objects.



* Realization:-

Realization can be defined as a relationship in which two elements are connected. One element describes some responsibility, which is not implemented & the other one implements them. This relationship exists in case of interfaces.



UML Diagrams

- All the elements, relationships are used to make a complete UML diagram & the diagram represents a system. UML included the following nine diagrams.
- class diagram
- Sequence diagram
- Activity diagram
- Deployment diagram
- object diagram (instance of class)
- collaboration diagram
- State Chart diagram
- Component diagram
- Use case diagram

Encapsulation

- Use for hiding unwanted data & giving relevant results
- One can use abstraction using interface & Abstract class.
- Abstraction solve the problem in design level
- For simplicity, we can say abstraction means hiding implementation using Abstract class & interface

Abstraction

- Hiding the code & data into single unit to protect data from the outside world.
- One can implement encapsulation using Access modifier
- Encapsulation solve the problem in implementation level
- For simplicity encapsulation means hiding data using getter & setter

Object oriented Modeling

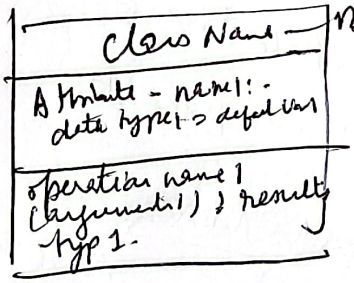
It is an approach to modeling an application that is used at the beginning of the S/w life cycle when using an object-oriented approach to S/w development. Object oriented modelling is typically done via use cases & abstract definitions of the most important objects. the most common language used to do object-oriented modelling (UML)

- UML. a model is structural, emphasing the organization of the system or may be behavioural model. dynamic of the system.

How to represent class in OMT

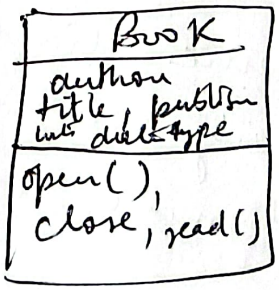
- the classes are represented by a rectangular box. which may be divided into 3 parts.
  - top part contains the name of the class written in bold.
  - Middle part contains a list of attributes
  - Bottom part contains a list of operations

Eg 1-



Book

Eg 2-



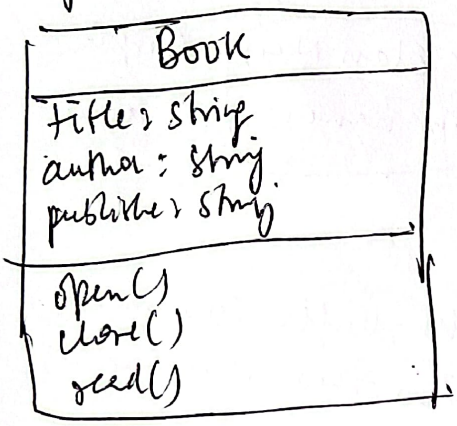
Attributes of class.

- An attribute is a data value held by objects in a class
- Each attribute has a value for each object instance
- this value should be a pure data value, not an object
- Attributes are listed in the second part of the class box
- Attributes may or may not be shown :, It depends on level of detail desired.
- Each attribute name may be followed by the details such as type & default value

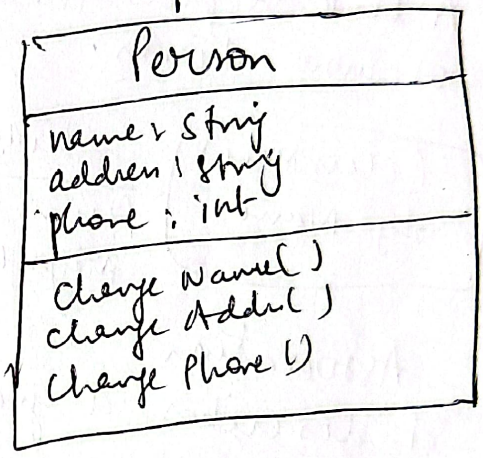
Operation of class

1) function or transformation that may be applied to or by object in a class.
- Operations are listed in the third part of the class box.
- Operations may or may not be shown: it depends on the level of details desired.
- Each operation may be followed by optional details such as argument list & result type.
- the name & type of each argument may be given.
- An empty argument list in parentheses shows explicitly that there are no arguments.
- All objects in a class share the same operations.

eg:-

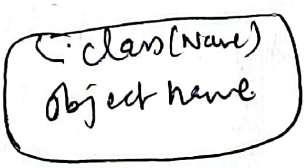


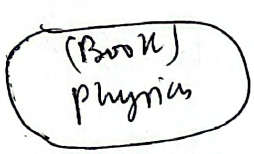
eg, 2:-

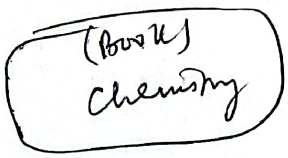


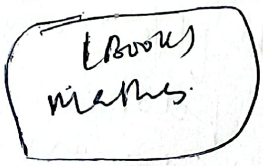
How to represent object in OMT

- An object is an instance of an object class
- Rounded box represented an object instance in OMT
- Object instance is a particular object from an object.
- Obj. instance can be used in instance ~~change~~ diagram which are useful for documenting test cases.









Class Person

(Person) Mayank

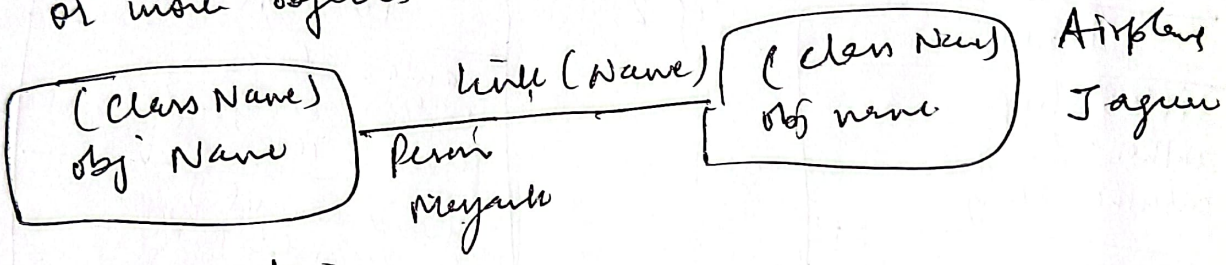
(Person) Kartik

(Person) Kavita

Link & Association

Link :- 1) A link represents a connection through which an obj collaborates with other objects.

- Rumbaugh defined it as :- "a physical or conceptual connection b/w objects."
- through a link, one obj may invoke the methods or navigate through another object.
- A link depicts the ~~relation~~ relationship b/w two or more objects.



Association

- It describes a group of links with common structure & common semantics b/w two or more classes.
- It is represented by a line labled with the association name in italics.
- Association name is optional. If the association is given a name, it should be written above the line.
- Association name are Italized.

Binary :- the name reads as a particular direction (ie, left to right), but the binary association can be traversed in either direction.

- eg:- a pilot flies an airplane or an airplane is flown by a pilot.
- All the links on an association connect objects from the same classes.
- Associations are bidirectional in nature.