

UML (Unified Modelling Language)

graphical / pictorial representation (blueprint).

- it is like before making something we make its model first. & this can be a model we can also show to a person who doesn't understand programming.

modelling vs designing

↓
(we make use
case diagram)

↓
(making use case diagram
but with code)

History of UML:-

UML (1.0) (start) ----- → UML (2.0) last
in 2003

Principle of modelling:-

- choose model wisely
more precise → less error.
- best model is connected to reality
- every model may be expressed different level of precision.
- no single model is sufficient

UML (Unified Modelling Language)

graphical / pictorial representation (blueprint).

- it is like before making something we make its model first. & this can be a model we can also show to a person who doesn't understand programming.

modelling vs designing

↓
(we make use
case diagram)

↓
(making use case diagram
but with code)

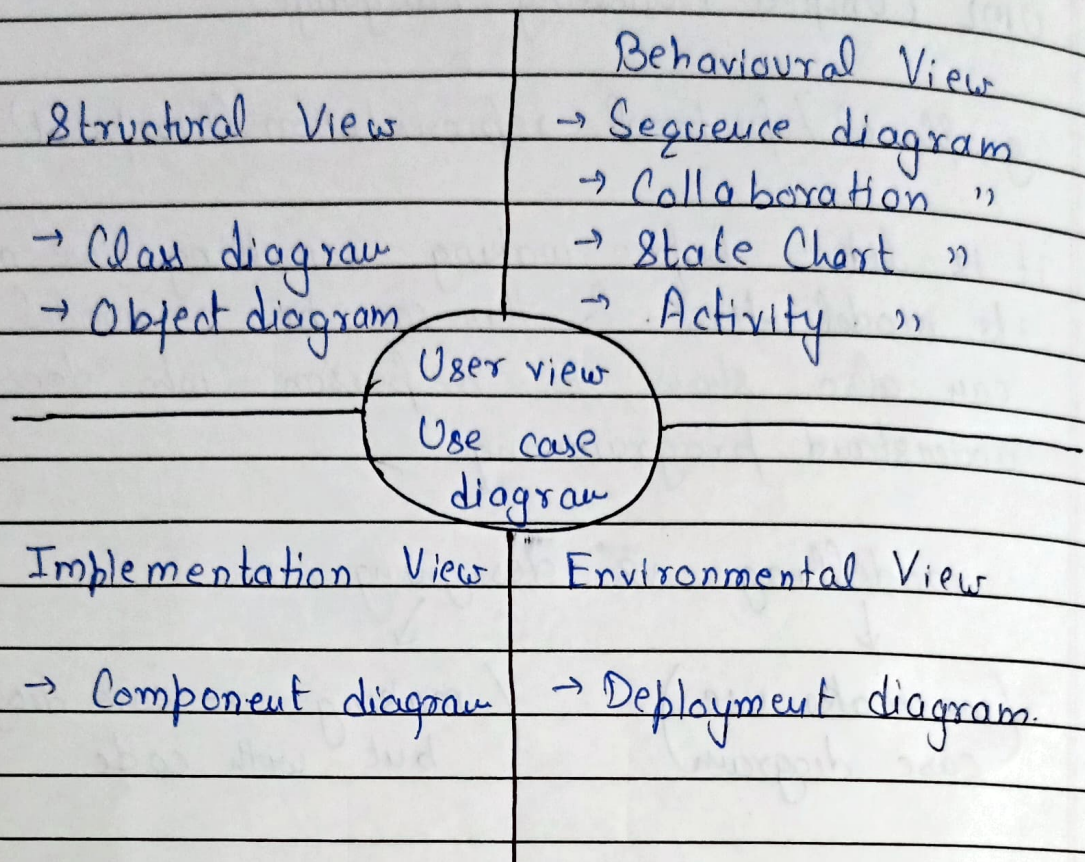
History of UML:-

UML (1.0) (start) - - - - - → UML (2.0) last
in 2003

Principle of modelling:-

- choose model wisely
more precise → less error.
- best model is connected to reality
- every model may be expressed different level of precision.
- no single model is sufficient

View of a system:



UML Building Blocks:-

- (i) Things
- (ii) Relationships
- (iii) Diagrams.

1. Things:-

Things are the most important building block of UML. Things can be structural, behavioural, grouping, annotational.

a) Structural:-

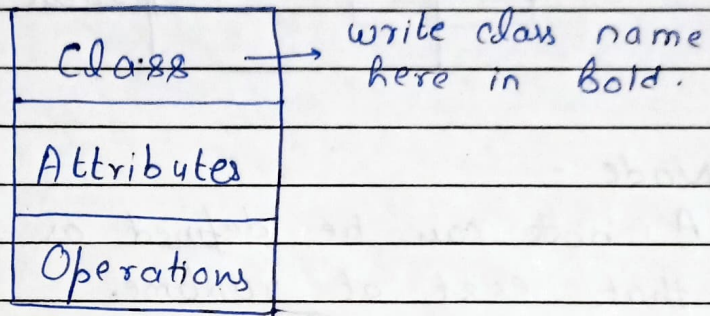
Structural things define the static part of the model. They represent the physical & conceptual elements.

Following are the brief description of the

structural things :-

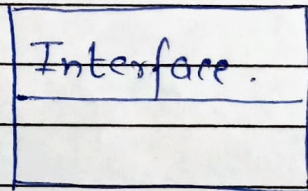
(i) Class :-

Class represent a set of object having similar responsibility.



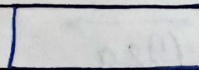
(ii) Interface :-

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



(iii) Collaboration :-

Collaboration define an interaction b/w elements



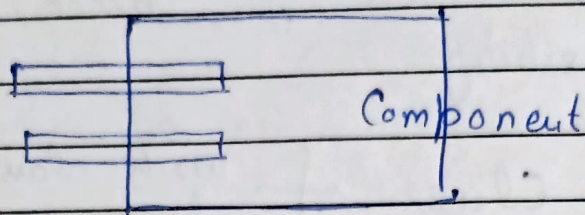
(iv) Use Case :-

Use case represent a set of actions performs by a system for a specific goal.

Use case

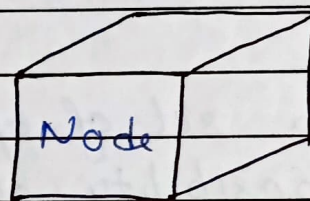
(v) Component :-

Component describe the physical part of the system



(vi) Node :-

A node can be defined as a physical element that exist at runtime.



b) behaviour things

A behavioural things consist of the dynamic parts of UML models.

Following are the behavioural things

(i) Interaction: →

$\overrightarrow{\text{msg}}$

→ Interaction is defined as a behaviour that consist of a group of message exchange among elements to accomplish a specific task.

(ii) State machines :- state

- State m/c is useful when the state of an object in its life cycle is important. It defines the sequence of states an object ~~through/throughs~~ in response to events, events are ext. factors responsible for state change.

c) ~~Grouping~~ 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 i.e. package.

(i) Package Package name Package

Package is the only one grouping thing available for gathering structural & behavioural things.

d) Annotation things

Annotation things can be defined as a mechanism to capture remarks, description and comments of UML model elements.

(i) Note:- Note. 4

- It is the only one annotational thing available.
- A note is used to render comments, constraints, etc. of a UML elements.

2. Relationship:-

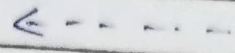
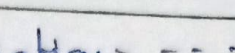
Relationship is another most imp building block of UML. It shows how the elements are associated with each other & this association describe the functionality of an application.

There are 4 kinds of Relationship are:-


- (i) Dependency
- (ii) Association
- (iii) Generalization
- (iv) Realization.

(i) Dependency:- ----->


It is a relationship b/w two things in which change in one element affects the other.

(ii) Association :-  Association 

- It 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.

(iii) 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.

(iv) Realization 

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