UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.

UML stands for Unified Modeling Language.

UML is a pictorial language used to make software blue prints.

uml is the object modeling programing language.

-----Modeling-----

the modeling is the consists of the three terminology---

1. Analyze----- identified requerement

2. view ---------->we convert the technical term

3. documenting ------->we combine the all requerement which in technical form as documents

uml----

for desinging view of technical object for user understating

Structural things:

### Grouping things:

Structural things:

The **Structural things** define the static part of the model. They represent physical and conceptual elements. Following are the brief descriptions of the structural things.

**Class:**

Class represents set of objects having similar responsibilities.

class

**Interface:**

Interface defines a set of operations which specify the responsibility of a class.

Interface

**Collaboration:**

Collaboration defines interaction between elements.

Collaboration

**Use case:**

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

Use case

**Component:**

Component describes physical part of a system.

Component

**Node:**

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



### 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 and behavioral things.



## (2) Relationship :

**Relationship** is another most important building block of UML. It shows how elements are associated with each other and this association describes the functionality of an application.

There are four kinds of relationships available.

### Dependency:

Dependency is a relationship between two things in which change in one element also affects the other one.

Dependency

### Association:

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

Association

## (3) UML Diagrams:

UML diagrams are the ultimate output of the entire discussion. All the elements, relationships are used to make a complete UML diagram and the diagram represents a system.

The visual effect of the UML diagram is the most important part of the entire process. All the other elements are used to make it a complete one.

UML includes the following nine diagrams and the details are described in the following chapters.

* Class diagram
* Object diagram
* Use case diagram
* Sequence diagram
* Collaboration diagram
* Activity diagram
* State chart diagram
* Deployment diagram
* Component diagram

UML plays an important role in defining different perspectives of a system. These perspectives are:

* Design
* Implementation
* Process
* Deployment

And the centre is the **Use Case** view which connects all these four. A **Use case** represents the functionality of the system. So the other perspectives are connected with use case.

* **Design** of a system consists of classes, interfaces and collaboration. UML provides class diagram, object diagram to support this.
* **Implementation** defines the components assembled together to make a complete physical system. UML component diagram is used to support implementation perspective.
* **Process** defines the flow of the system. So the same elements as used in *Design* are also used to support this perspective.
* **Deployment** represents the physical nodes of the system that forms the hardware. UML deployment diagram is used to support this perspective.

## Structural Things:

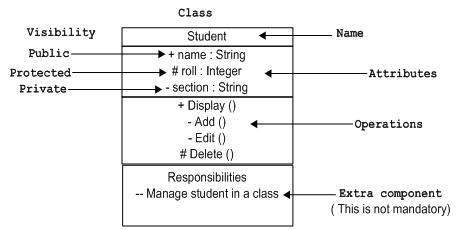
Graphical notations used in structural things are the most widely used in UML. These are considered as the nouns of UML models. Following are the list of structural things.

* Classes
* object
* Interface
* Collaboration
* Use case
* Active classes
* Components
* Nodes

### Class Notation:

UML *class* is represented by the diagram shown below. The diagram is divided into four parts.

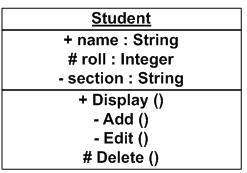
* The top section is used to name the class.
* The second one is used to show the attributes of the class.
* The third section is used to describe the operations performed by the class.
* The fourth section is optional to show any additional components.



Classes are used to represent objects. Objects can be anything having properties and responsibility.

### Object Notation:

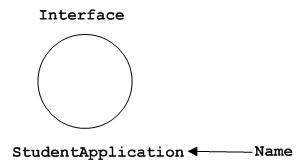
The *object* is represented in the same way as the class. The only difference is the *name* which is underlined as shown below.



As object is the actual implementation of a class which is known as the instance of a class. So it has the same usage as the class.

### Interface Notation:

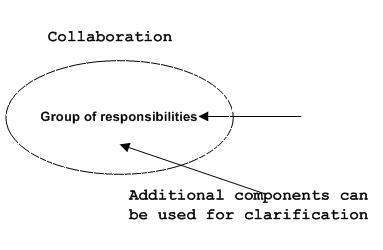
Interface is represented by a circle as shown below. It has a name which is generally written below the circle.



Interface is used to describe functionality without implementation. Interface is the just like a template where you define different functions not the implementation. When a class implements the interface it also implements the functionality as per the requirement.

### Collaboration Notation:

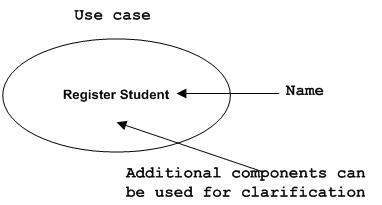
Collaboration is represented by a dotted eclipse as shown below. It has a name written inside the eclipse.



Collaboration represents responsibilities. Generally responsibilities are in a group.

### Use case Notation:

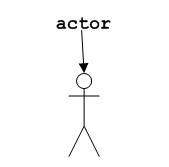
Use case is represented as an eclipse with a name inside it. It may contain additional responsibilities.



Use case is used to capture high level functionalities of a system.

### Actor Notation:

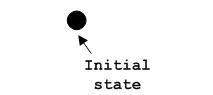
An actor can be defined as some internal or external entity that interacts with the system.



Actor is used in a use case diagram to describe the internal or external entities.

### Initial State Notation:

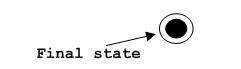
Initial state is defined to show the start of a process. This notation is used in almost all diagrams.



The usage of Initial State Notation is to show the starting point of a process.

### Final State Notation:

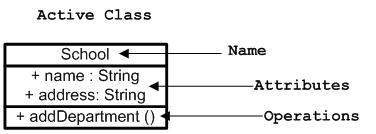
Final state is used to show the end of a process. This notation is also used in almost all diagrams to describe the end.



The usage of Final State Notation is to show the termination point of a process.

### Active class Notation:

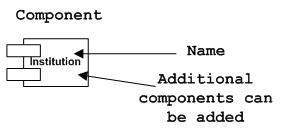
Active class looks similar to a class with a solid border. Active class is generally used to describe concurrent behaviour of a system.



Active class is used to represent concurrency in a system.

### Component Notation:

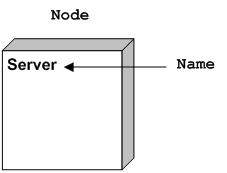
A component in UML is shown as below with a name inside. Additional elements can be added wherever required.



Component is used to represent any part of a system for which UML diagrams are made.

### Node Notation:

A node in UML is represented by a square box as shown below with a name. A node represents a physical component of the system.



Node is used to represent physical part of a system like server, network etc.

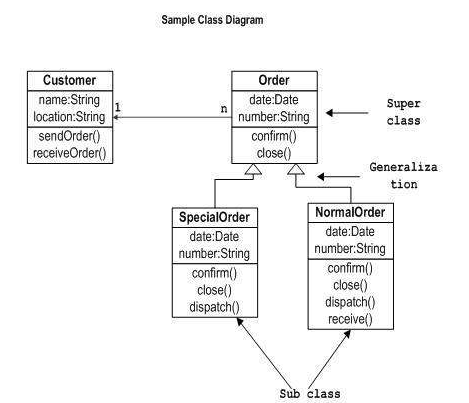
## Behavioral Things:

Dynamic parts are one of the most important elements in UML. UML has a set of powerful features to represent the dynamic part of software and non software systems. These features include *interactions* and *state machines*.

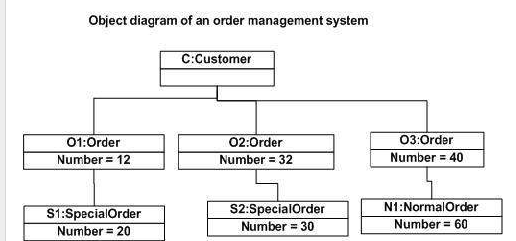
Interactions can be of two types:

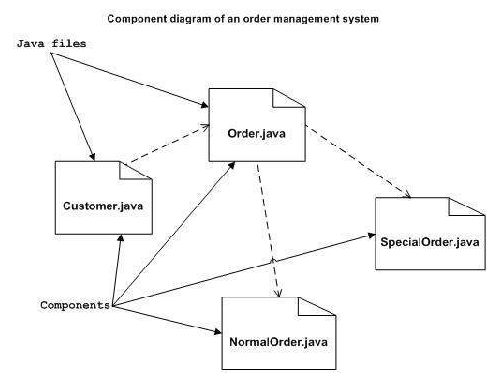
* Sequential (Represented by sequence diagram)
* Collaborative (Represented by collaboration diagram)

Go to project uml diagrams

Class diagram: The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints. It is also known as a*structural diagram*. 

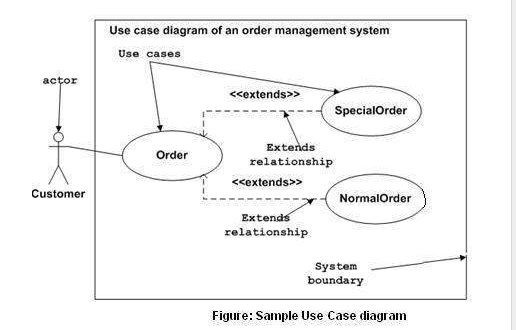
Object diagrams represent the static view of a system but this static view is a snapshot of the system at a particular moment.

Object diagrams are used to render a set of objects and their relationships as an instance. 

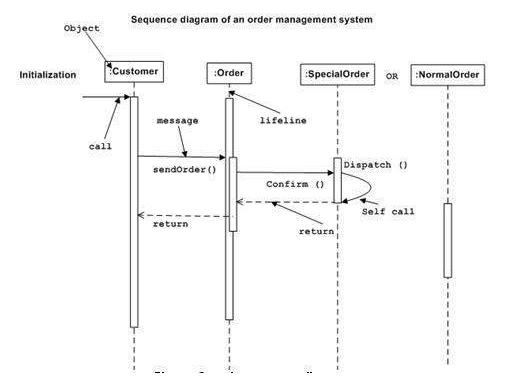
component diagrams are used to visualize the organization and relationships among components in a system. 

use case diagram

The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system.



## The Sequence Diagram: The sequence diagram is having four objects (Customer, Order, SpecialOrder and NormalOrder).



## The Collaboration Diagram:

Here in collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. 