The Unified Modeling Language (UML) is a graphical language for OOAD that gives a standard way to write a software system’s blueprint. It helps to visualize, specify, construct, and document the artifacts of an object-oriented system. It is used to depict the structures and the relationships in a complex system.

**Brief History**

It was developed in 1990s as an amalgamation of several techniques, prominently OOAD technique by Grady Booch, OMT Obje ctModelingTechniqu e by James Rumbaugh, and OOSE ObjectOrientedSoftwareEngineering by Ivar Jacobson. UML attempted to standardize semantic models, syntactic notations, and diagrams of OOAD.

**Systems and Models in UML**

**System**: A set of elements organized to achieve certain objectives form a system. Systems are often divided into subsystems and described by a set of models.

**Model**: Model is a simplified, complete, and consistent abstraction of a system, created for better understanding of the system.

**View**: A view is a projection of a system’s model from a specific perspective.

**Conceptual Model of UML**

The Conceptual Model of UML encompasses three major elements:

- Basic building blocks
- Rules
- Common mechanisms

**Basic Building Blocks**

The three building blocks of UML are:

- Things
- Relationships
- Diagrams

**a Things:**

There are four kinds of things in UML, namely:

- **Structural Things**: These are the nouns of the UML models representing the static elements that may be either physical or conceptual. The structural things are class, interface, collaboration, use case, active class, components, and nodes.

- **Behavioral Things**: These are the verbs of the UML models representing the dynamic behavior over time and space. The two types of behavioral things are interaction and state machine.

- **Grouping Things**: They comprise the organizational parts of the UML models. There is only one kind of grouping thing, i.e., package.

- **Annotational Things**: These are the explanations in the UML models representing the comments applied to describe elements.

**b Relationships:**
Relationships are the connection between things. The four types of relationships that can be represented in UML are:

- **Dependency**: This is a semantic relationship between two things such that a change in one thing brings a change in the other. The former is the independent thing, while the latter is the dependent thing.

- **Association**: This is a structural relationship that represents a group of links having common structure and common behavior.

- **Generalization**: This represents a generalization/specialization relationship in which subclasses inherit structure and behavior from super-classes.

- **Realization**: This is a semantic relationship between two or more classifiers such that one classifier lays down a contract that the other classifiers ensure to abide by.

**Diagrams**: A diagram is a graphical representation of a system. It comprises of a group of elements generally in the form of a graph. UML includes nine diagrams in all, namely:

- Class Diagram
- Object Diagram
- Use Case Diagram
- Sequence Diagram
- Collaboration Diagram
- State Chart Diagram
- Activity Diagram
- Component Diagram
- Deployment Diagram

**Rules**

UML has a number of rules so that the models are semantically self-consistent and related to other models in the system harmoniously. UML has semantic rules for the following:

- Names
- Scope
- Visibility
- Integrity
- Execution

**Common Mechanisms**

UML has four common mechanisms:

- Specifications
- Adornments
- Common Divisions
- Extensibility Mechanisms

**Specifications**

In UML, behind each graphical notation, there is a textual statement denoting the syntax and semantics. These are the specifications. The specifications provide a semantic backplane that contains all the parts of a system and the relationship among the different paths.
Adornments

Each element in UML has a unique graphical notation. Besides, there are notations to represent the important aspects of an element like name, scope, visibility, etc.

Common Divisions

Object-oriented systems can be divided in many ways. The two common ways of division are:

- **Division of classes and objects**: A class is an abstraction of a group of similar objects. An object is the concrete instance that has actual existence in the system.

- **Division of Interface and Implementation**: An interface defines the rules for interaction. Implementation is the concrete realization of the rules defined in the interface.

Extensibility Mechanisms

UML is an open-ended language. It is possible to extend the capabilities of UML in a controlled manner to suit the requirements of a system. The extensibility mechanisms are:

- **Stereotypes**: It extends the vocabulary of the UML, through which new building blocks can be created out of existing ones.

- **Tagged Values**: It extends the properties of UML building blocks.

- **Constraints**: It extends the semantics of UML building blocks.