UML – UNIFIED MODELLING LANGUAGE

DEFINITION
- Unified
  - Combines the best from existing object oriented software modelling methodologies.
  - Grady Booch, James Rumbaugh, and Ivor Jacobson are the primary contributors to UML.
- Modelling
  - Used to present a simplified view of reality in order to facilitate the design and implementation of object-oriented software systems.
- Language
  - UML is primarily a graphical language that follows a precise syntax.

HISTORY
- OO modelling languages made their appearance in the late 70's.
- As the usefulness of OO programming became undeniable, more OO modelling languages began to appear.
- By the start of the 90's there was a flood of modelling languages, each with its own strengths and weaknesses.
- In 1994 the UML effort officially began as a collaborative effort between Booch and Rumbaugh. Jacobson was soon after included in the effort.
- The goal of UML is to be a comprehensive modelling language (all things to all people) that will facilitate communication between all members of the development effort.

OVERVIEW
- UML is a language.
  - Conforms to specific rules.
  - Allows the creation of various models.
- UML is a language for visualizing.
  - UML is a graphical language.
  - Pictures often facilitate communication (a picture is worth a thousand words).
- UML uses concepts from functional, state based and event based modeling approaches.
  - UML supports both forward and reverse engineering.
  - UML is semi-formal; its formal basis is currently a research topic.
- UML is a language for documenting design
  - Provides a record of what has been built.
  - Useful for bringing new programmers up to speed.
  - Useful when developing new releases of a product.
- UML is intended primarily for software intensive systems

A CONCEPTUAL MODEL

A conceptual model needs to be formed by an individual to understand UML. UML contains three types of building blocks: things, relationships, and diagrams.

- Things
  - Structural things
ARCHITECTURE

Architecture refers to the different perspectives from which a complex system can be viewed. The architecture of a software-intensive system is best described by five interlocking views:

- Use case view: system as seen by users, analysts and testers.
- Design view: classes, interfaces and collaborations that make up the system.
- Process view: active classes (threads).
- Implementation view: files that comprise the system.
- Deployment view: nodes on which software resides.

UML AND SDLC

UML is involved in each phase of the software development life cycle. The UML development process is

- Use case driven
- Architecture-centric
- Iterative and incremental