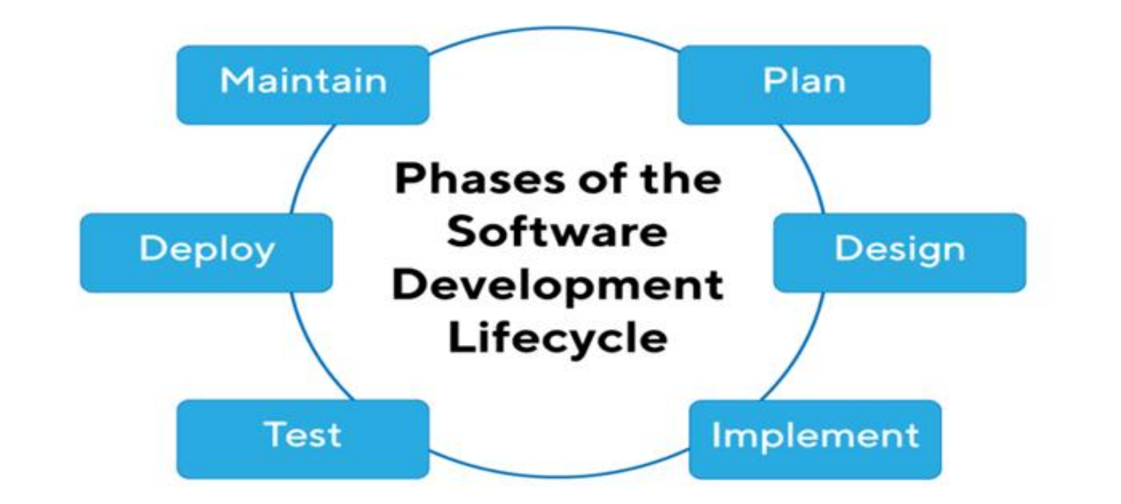
CS3004D Software Engineering

UML Diagrams

Before moving to UML…

SDLC -

Software Development Life Cycle

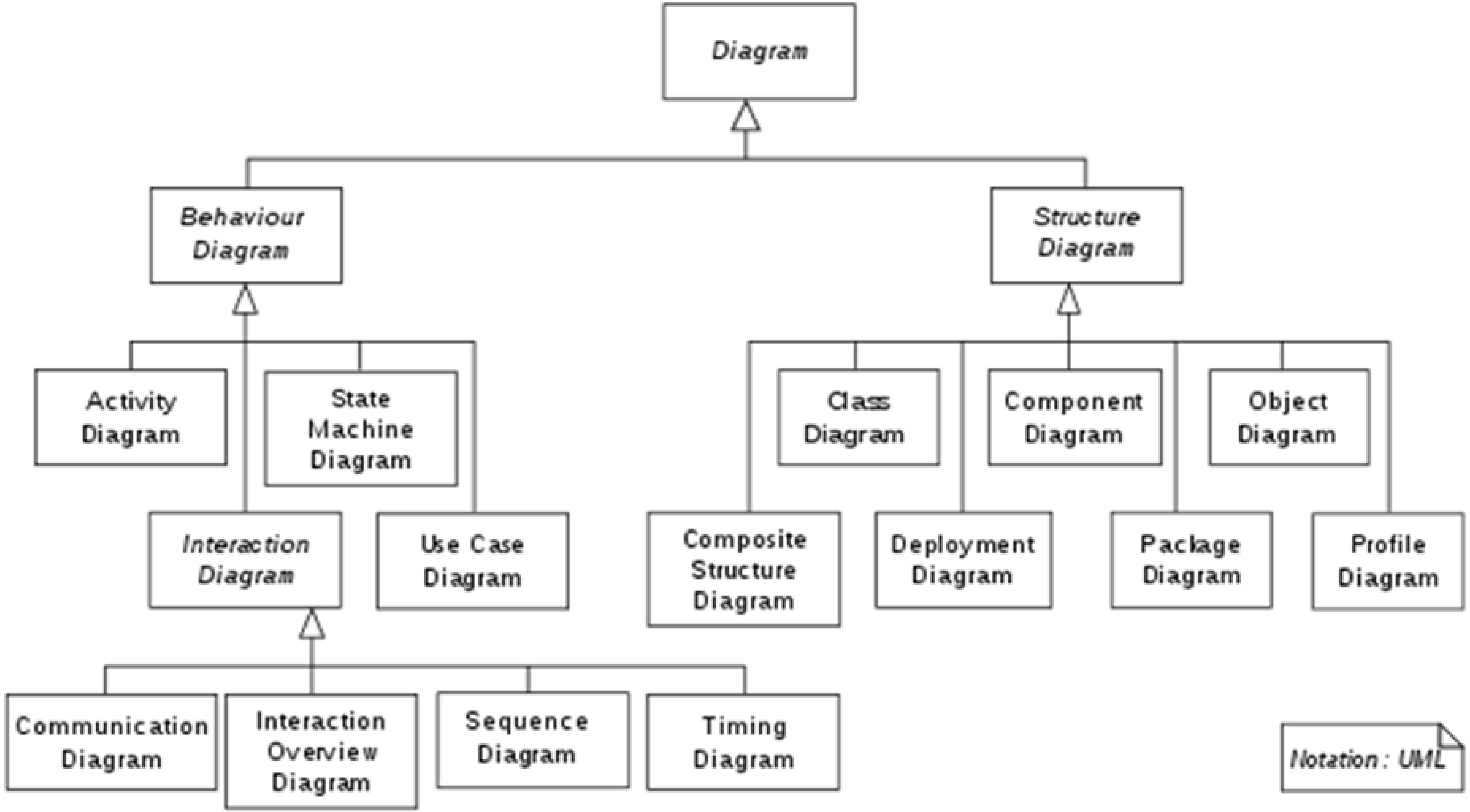


Role of UML (Unified Modelling Language)

> “It is a general-purpose, developmental ***modeling language*** in the field of software engineering that is intended to provide a ***standard way to visualize the design of a system***”

- UML is **NOT a design methodology**
- UML is a language for ***representing/expressing/documenting*** the design created using some methodology

First View



- *UML provides a set of notations, to visualise or to graphically model the system.*
- *It has it’s own syntax (symbols and sentence formation rules) and semantics (meanings of symbols and sentences)*

One more point to remember…

Design can be

Object-oriented design is basically the bottom-up approach

Object Orientation

Identify the classes

- Their instances are Objects
- They interact to capture the functionalities

Simple to Complex

The Bottom Up Approach

UML —

Agree on OOD

Different ways of going about - Different notational systems

Booch - Rumbaugh - Jacobson

Object Management Group (OMG) adopted UML in 1997, ISO in 2005
A standard **representation** of the Object Oriented Design

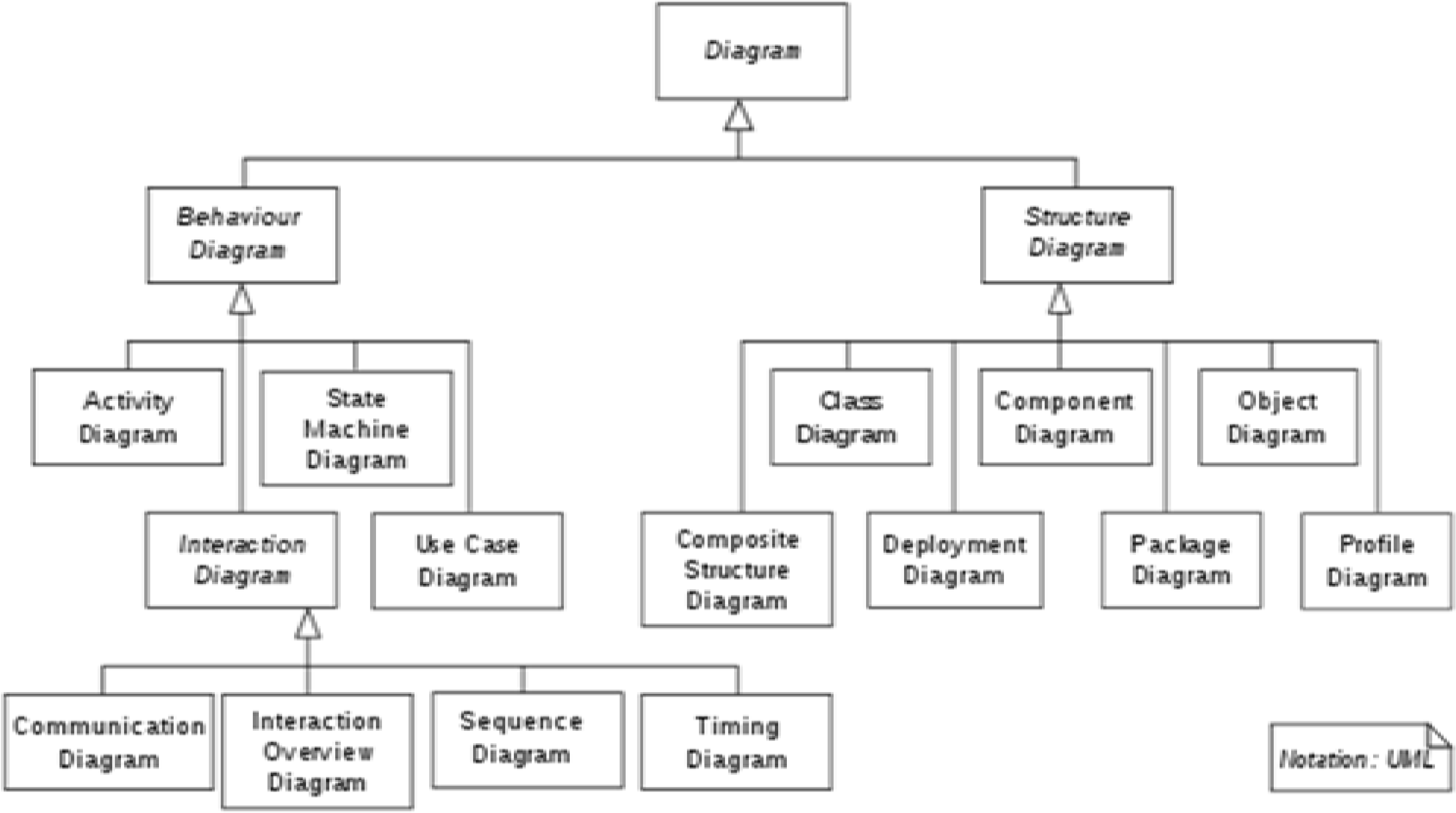


Diagram
Behaviour Diagram
Structure Diagram
Activity Diagram
State Machine Diagram
Interaction Diagram
Use Case Diagram
Communication Diagram
Interaction Overview Diagram
Sequence Diagram
Timing Diagram
Class Diagram
Component Diagram
Object Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Profile Diagram
Notation: UML

Different views from Different UML Diagrams

- User view
- Behavioral view
- Structural view
- Implementation view
- Environment view

Standard Reference Link

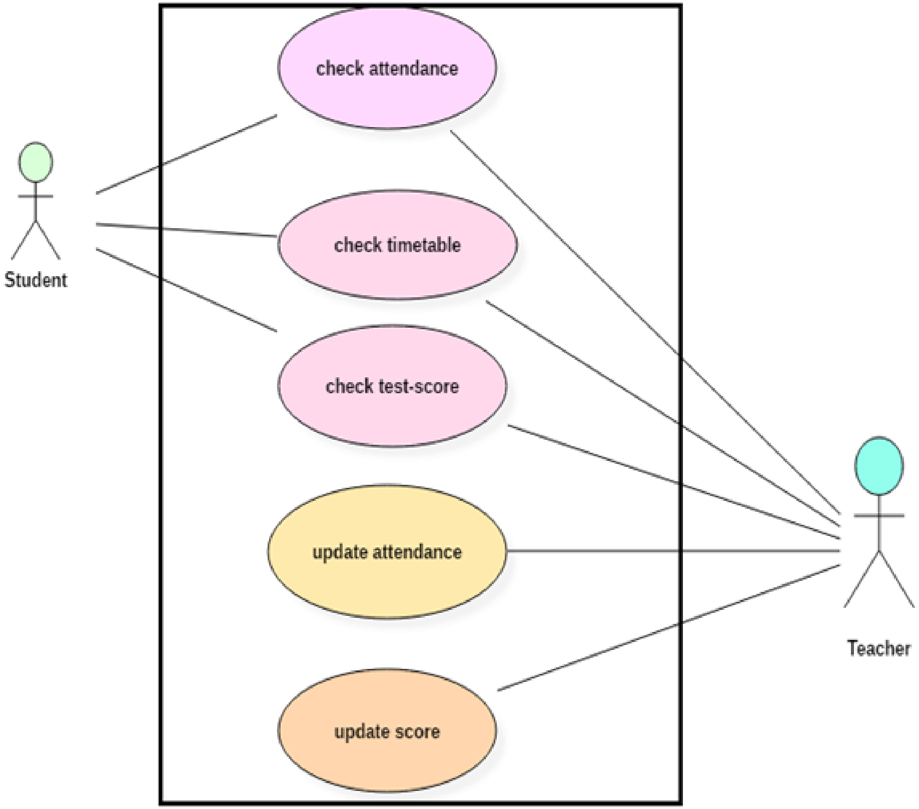
In case of conflicts, we will use this for resolving them:

uml-diagrams.org

User View - Usecase Diagram

- Represents the **user perception** of the system
- *How the user is going to use the system?*

- Between user and system (**user interactions to the system**)
- Represents the different usage scenarios of the system
- Created as part of **usecase analysis** in requirements phase



Basic Components - Usecase Diagram

- Actors (stick figure)
- Usecases (Ellipse)
- System Boundary
- Relationship (lines and arrows)

Actor - Usecase Diagram

An actor is a **role** that a user or some other system plays when interacting with your system

Eg: Learning Management System

Actors :- Student, Teacher, Admin

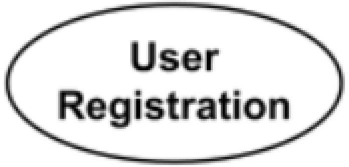
Eg: Library System

Actors :- Borrower, Clerk

Usecases -

The tasks that each actor will need to do with the system

A *use case* is a typical sequence of actions that a user performs in order to complete a given task


User
Registration