

Systems Development: Object Oriented Analysis and Design

(H172 35)

The 9 UML Diagrams

Structural Diagrams:

The structural diagrams represent the **static** aspect of a system

These static aspects represent those parts of a diagram which forms the main structure (which is fixed or stable)

These static parts are represented by classes, interfaces, objects, components and nodes

The **four** **structural** diagrams are:

* Class diagram
* Object diagram
* Component diagram
* Deployment diagram

The Class Diagram:

Class diagrams are the most common diagrams used in UML

Class diagrams consist of classes, interfaces, associations and collaboration

Fundamentally, Class diagrams represent the object oriented view of a system which is static in nature

As the Class diagram represents the object orientation of a system, it is generally used for development purposes

It is the most widely used diagram at the time of system construction

The Object Diagram:

An Object diagram can be described as an instance of a Class diagram

These diagrams are closer to the real life scenarios from which we implement the system

Object diagrams represent a set of objects together with their relationships

(just like Class diagrams) and also represent the static view of the system

Their use is similar to that of Class diagrams but they are used to build a prototype of a system from the practical perspective

The Component Diagram:

Component diagrams represent a set of components and their relationships

These components consist of classes, interfaces or collaborations, so Component diagrams represent the implementation view of a system

During the design phase, software artefacts (classes, interfaces etc.) of a system are arranged in different groups depending upon their relationship

Now these groups are known as components

Component diagrams are used to visualize the implementation

The Deployment Diagram:

Deployment diagrams are a set of nodes and their relationships

These nodes are physical entities where the components are deployed

Deployment diagrams are used for visualizing a deployment view of a system

Generally used by the deployment team

Note:

If the above descriptions and usages are observed carefully then it is very clear that all the diagrams have some relationship with one another

Component diagrams are dependent upon the classes, interfaces etc. which are part of Class / Object diagram

In addition, the Deployment diagram is dependent upon the components which are used to make Component diagrams

Behavioural Diagrams:

The behavioural diagrams represent the **dynamic** aspect of the system

Any system can have two aspects, static and dynamic

So a model is considered as complete when both the aspects are covered fully

The dynamic aspect can be further described as the changing / moving parts of a system

UML has the following **five** types of **behavioural** diagrams:

* Use case diagram
* Sequence diagram
* Collaboration diagram
* Statechart diagram
* Activity diagram

The Use case Diagram:

Use case diagrams are a set of use cases, actors and their relationships

They represent the use case view of a system

A use case represents a particular functionality of a system

So the Use case diagram is used to describe the relationships among the functionalities and their internal / external controllers

These controllers are known as actors

The Sequence Diagram:

A Sequence diagram is an interaction diagram

From it’s name it is clear that the diagram deals with the sequence of messages flowing from one object to another

Interaction among the components of a system is very important from an implementation and execution perspective

The Sequence diagram is used to visualize the sequence of calls in a system to perform a specific functionality

The Collaboration Diagram:

The Collaboration diagram is yet another form of interaction diagram

It represents the structural organization of a system and the messages sent / received

A structural organization consists of objects and links

The purpose of the Collaboration diagram is similar to the Sequence diagram, but the specific purpose of the Collaboration diagram is to visualize the organization of objects and their interaction

The Statechart Diagram:

Any real time system is expected to be able to react to internal / external events

These events are responsible for the state change of the system

The Statechart diagram is used to represent the event driven state change of a system

It basically describes the state change of a class, interface etc.

Statechart diagrams are used to visualize the reaction of a system caused by

internal / external factors

The Activity Diagram:

The Activity diagram describes / visualizes the flow of control in a system

It consists of activities and links

The flow can be sequential, concurrent or branched

Activities are nothing but the functions of a system

Numbers of Activity diagrams are prepared to capture the entire flow in a system

and to visualise how the system will work when executed

Note:

The dynamic nature of a system can be very difficult to capture

So UML has provided features to capture the dynamics of a system from different angles

Sequence diagrams and Collaboration diagrams are ‘isomorphic’ so they can be converted from one another without losing any information

This is also true for the Statechart and Activity diagrams