System Design & Analysis

LECTURE 05

UML

- Uml stands for Unified Modeling Language
- The question rises, why we need modeling?

Why Model Software System?

- Software development is complex
- Modeling can help build a system easily & quickly
- Useful in simulations
- Can evolve when our understand of system increases
- We can choose the details in the model
- Simple view of the system
- No need to dig into the system for details
- A model can represent real or abstract things in any domain

Model

▶ A model is an abstraction of something that exists in real life

"A model captures the view of a partical system. It is an abstraction of the physical system, with a certain purpose. This purpose determines what is to be included in the model and what is irrelevant. Thus, the model completely describes those aspects of the physical system that are relevant to the purpose of the model, at an appropriate level of detail."

Modelling Language

- Made up of pseudo-code, pictures, comments, diagrams, etc.
- These elements are made up a modelling language called notation
- Code may be used for modelling
  - Too verbose i-e. information overloaded
  - Only models implementation, not design
  - Cannot be easily ported to other implementations
  - Does not give a bigger picture of the system
  - Clients may not understand code

Modeling in Informal Languages

- Informal languages don't have formal notations
- For example, we may use natural language, but
  - Difficult to express
  - Leads to ambiguity
  - No rules
  - Not suited for systems modelling
  - Non-standard notations

What is UML

▶ UML is the "Unified" Modelling Language"

▶ Standard modelling language for software development

▶ Maintained by Object Management Group

▶ Current version is 2.5

▶ Bring together ideas of three modelling methods of Jacobson, Rumbaugh & Booch

What is UML?

- ▶ Made up pf pseudo-code, actual code, pictures, diagrams, etc.
- ▶ Describes the system we want to develop
- ▶ Depends mostly on notations to describe the system

What UML Is Not

- It is not a modelling tool
  - It is just a way to model system

- It is not a programming language

- It is not a methodology or a process
  - Rather it helps document various stages of a process

Why UML

▶ Systems design on large scale is difficult

▶ All kind of applications will have thousands of components that interact with each other

  ▶ Components need to be tracked

  ▶ Usefulness of a component for a system must be ascertained

  ▶ Customer requirements must be considered

▶ Designs need to be shared with stakeholders

▶ UML helps model this kind of complexity

  ▶ Captures, documents and helps communicate important aspects of the system's deign

UML Advantages

- Formal
- Concise
- Comprehensive
- Scalable
- Standard

View Model

▶ You can get a big picture of different UML diagrams through the view model

▶ Defines a coherent set of view that are used in construction of a software architecture

▶ A view represents the whole system from perspective of a related set of concerns

▶ This allows different stakeholder to view the system from their viewpoint

▶ The view model is an effective approach for dealing with the inherent complexity of the large distributed systems

Intent

▶ The intent behind making view model is to:

    ▶ Use a mechanism that separates the different aspects of the software system into different views

    ▶ Different stakeholders have different interest in the system

        ▶ Programmers – Classes

        ▶ Administrators – hardware, network, configuration, etc

        ▶ Customers - usability

View of the model

- UML model diagrams can be broken into different perspective or view
- Can be explained through kruchen's 4+1 Model

| Logical view | Process view |

Use Case View

| Physical view | Development View |
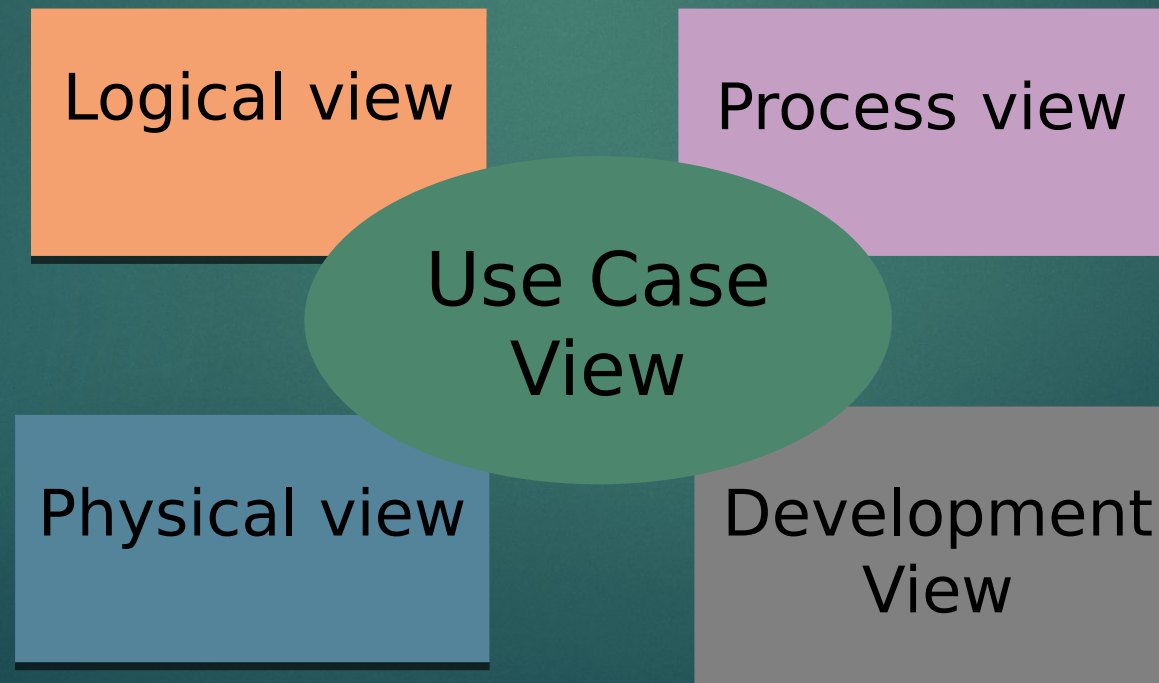
Logical view

▶ The first one is the logical view

▶ The logical view shows the parts that make up the system and how they interact with each other.

▶ It represents the abstractions that are used in the problem domain

   ▶ These abstractions are classes and objects

▶ Different UML diagrams show the logical way such as class diagram state diagram sequence, diagram comm... diagram and object diagram.

Logical view
- class diagram
- state diagram
- Sequence diagram
- communication diagram
- object diagram

Process View

- ▶ Then we have the process view

- ▶ Through this view, we can describe the processes of the system and how they communicate with each other using process

- ▶ Using process view, we can find out what needs to happen to the system

- ▶ So using process view we can understand the overall functioning of the system

- ▶ Activity diagram in UML represents the process view

Process view

- Activity Diagram

Physical View

- Next is physical view
- The physical view is the view that models the execution environment of the system
- Using this view, we can model the software entities onto the hardware that will host and run the entities
- The physical view in UML is represented through deployment diagrams

Physical view

- Deployment Diagrams

Development View

- next view is the development view
- the development view describes the modules are the components of the system.
- This might include packages or libraries.
- It gives a high-level view of the architecture of the system and helps in managing the layers of the system.
- UML provides two diagrams for development view.
  - component Diagram
  - package Diagrams

- All these four views are dependent on Use Case view

Development View
- Component Diagram
- Packages Diagram

Use Case

▶ They use case view illustrates the functionality of the system

▶ using use case we can capture the goals of the user or what the user expects from the system

▶ In UML, Use Cases can be created through use case diagrams or use case descriptions (we will discuss later)

▶ use cases can be created by analysts' architects or even by the users