Analysis [Unit -7]

Utsav Baral [**ROLL 10**]

Kushal Dhakal [**ROLL 9**]

Introduction

- Aim of analysis phase is to **analyze, specify and define** the system which is to be built.
- Two different models are developed in analysis; **the requirement model** and **the analysis model** which will describe what the system is to do.
- The models are problem oriented and no attention is paid to the real implementation environment thus they are fairly straightforward to develop from a functionality viewpoint.

The Requirements Model

- This model aims at delimiting the system and defining what functionality the system should offer.
- Could function as a contract between the developer and customer and thus forms the developers view of what the customer wants.
- Thus it is essential that this model should be readable also for non-OOSE practitioners.
- This model will govern the development of all other models, so this model is the central one throughout the whole system development.

- The requirement model will be:
    - Structured by analysis model
    - Realized by design model
    - Implemented by implementation model
    - And tested by testing model
- The requirement model consists of three parts:
    - Use-case model
    - Problem domain object model
    - User interface descriptions

- **Use-case model**
  - It is a model of how different types of users interact with system to solve a problem.
  - It describes the goals of the users, the interaction between the user and system and the required behavior of the system in satisfying these goals.
- **Problem domain object model**
  - Describes in an object-oriented style that concerns the system to be developed.
  - Eg: in the case the system should handle a lift, the Problem domain object model should describe how the lift works, in which ways the call can be made, which policy must be followed to handle the calls and what has to be done for the safety of the users.

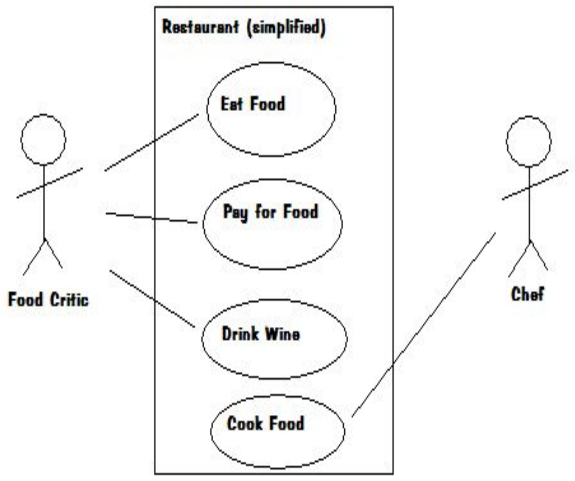


Fig: Use-case Model

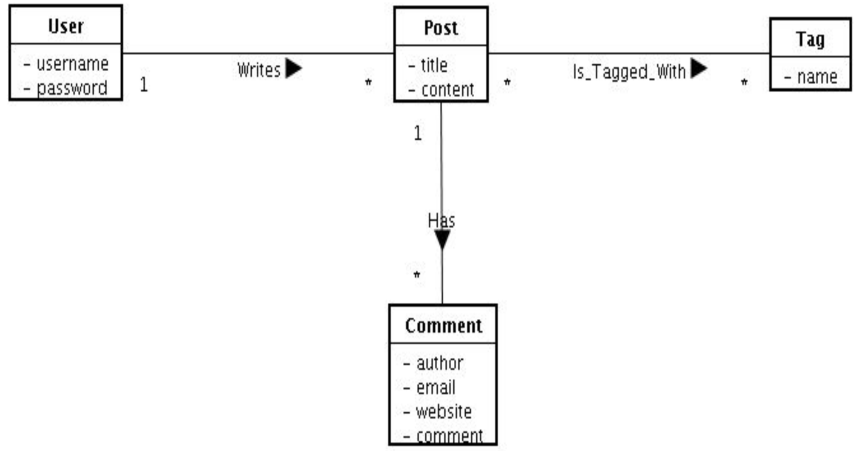


Fig: Problem domain object model

- **User interface descriptions**
    - We can use sketches of what the user will see on the screen when performing the use case or more sophisticated simulations using UIMS (User Interface Management System).
    - We can thus liven up the use case descriptions with real computer interaction by the potential users.
    - This will eliminate several possibilities of misunderstandings.

O.O.A

OBJECT ORIENTED ANALYSIS

[Modelling **real world Object** based on their **Description**]

Round Object, Used for Playing Games, Played on Ground

The Limitations of the Human Capacity for Dealing with Complexity

EITHER WE

OVERESTIMATE

or

WE **UNDERESTIMATE**

`OO Analysis → OO Design → OO implementation using OO languages`

WE ARE **HERE !**

Main purpose

1. How to **characterized a system.**
2. To know what are different **Relevant Object**.
3. How do **they relate to each Other.**
4. How to **specify or Model a problem** to create a **effective Design.**
5. **Examine** requirement , **Analyse** their Implications

What We Intend to Do.

To **Define** all the **classes** that are Relevant to the problem to be solved.
[To do that, numbers of task must occur]

1. Basic User requirements must be Known.
2. Classes must be Identified.
3. A class Hierarchy must be specified.
4. Object to Object relationship should be Represented.
5. Object Behaviour Must be Modelled.
6. Task 1-5 is reapplied iteratively until the model is Completed.

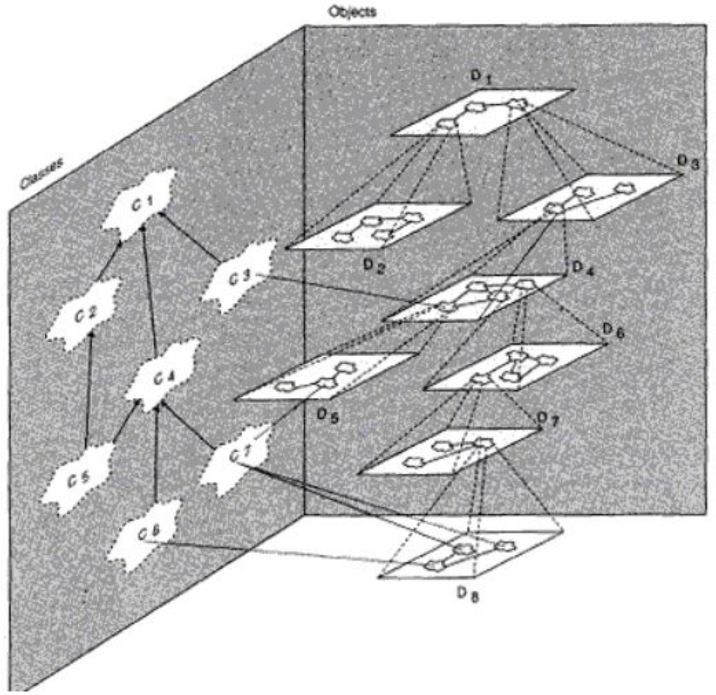


Figure 1-1
The Canonical Form of a Complex System

- The **discovery** of **common abstractions** and **mechanisms** greatly facilitates our **understanding of complex systems**

Object Oriented Decomposition [OOD]

- we **view** the **world** as a **set of autonomous agents** that **collaborate to perform some higher level behavior.**

-Object-oriented decomposition yields **smaller systems** through **the reuse of common mechanisms [OOP],** thus providing an important economy of expression.

-OOD is **flexible to changes** and can **better evolve over time.**

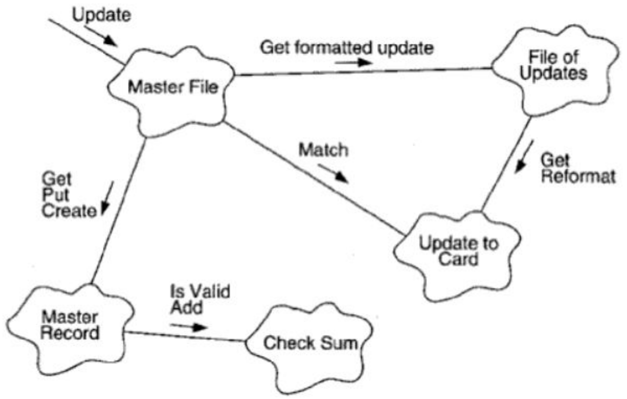


Figure 1-3
Object-Oriented Decomposition

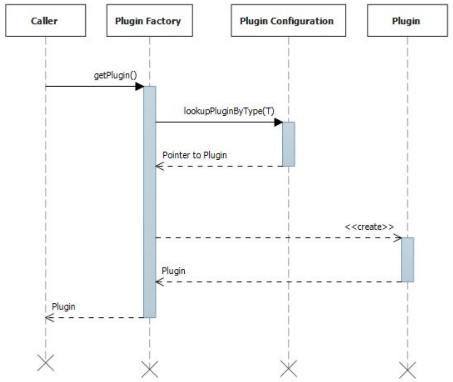
Conventional vs Object Oriented Analysis

- Conventional Analysis treats **Data** and **Process** as **two Separate Modules** but Object Oriented Analysis combines **Data** and **Process** that acts into **Object**.

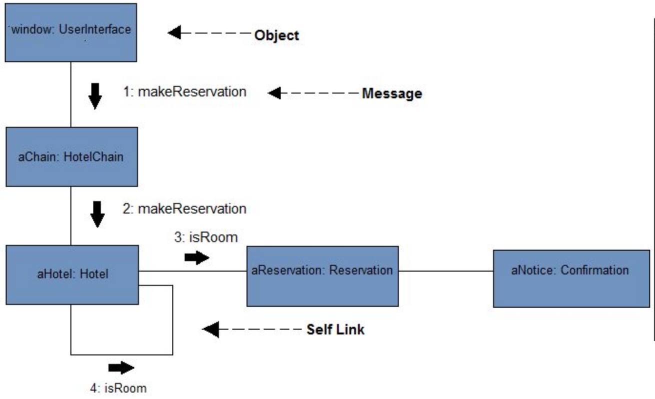
Conventional vs Object Oriented Analysis

Data Flow Diagrams	Use Case Model
Decision Table\Tree	Object Model 1. Find Classes and Class Relations
Entity Relation [E.R] Analysis	-Object relations 1. Sequence Diagram 2. Collaboration Diagram 3. State Machine Diagram

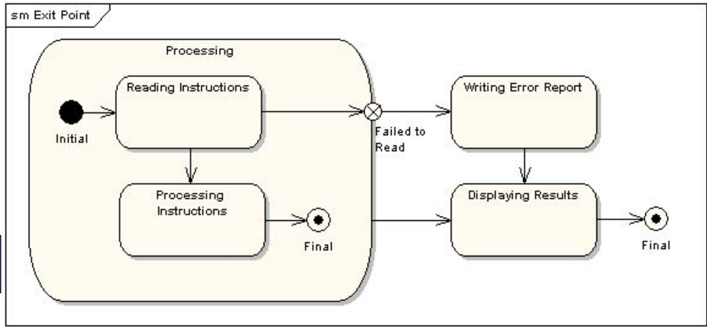
Sequence Diagram



Collaboration Diagram



State Machine Diagram



OOA Landscape

Different OOA Method.

- BOOCH METHOD
- RUMBAUGH METHOD
- JACOBSON METHOD
- COAD & YOURDON METHOD
- WIRFS-BROCK METHOD

Unified Approach To OOA. [a.ka. **UML**]

Grady Booch , James Rumbaugh , and Ivar Jacobson collaborated to combine the Best features of their individual Object Oriented Analysis method into a unified method. Today we know it as **UML [Unified Modelling Language]**

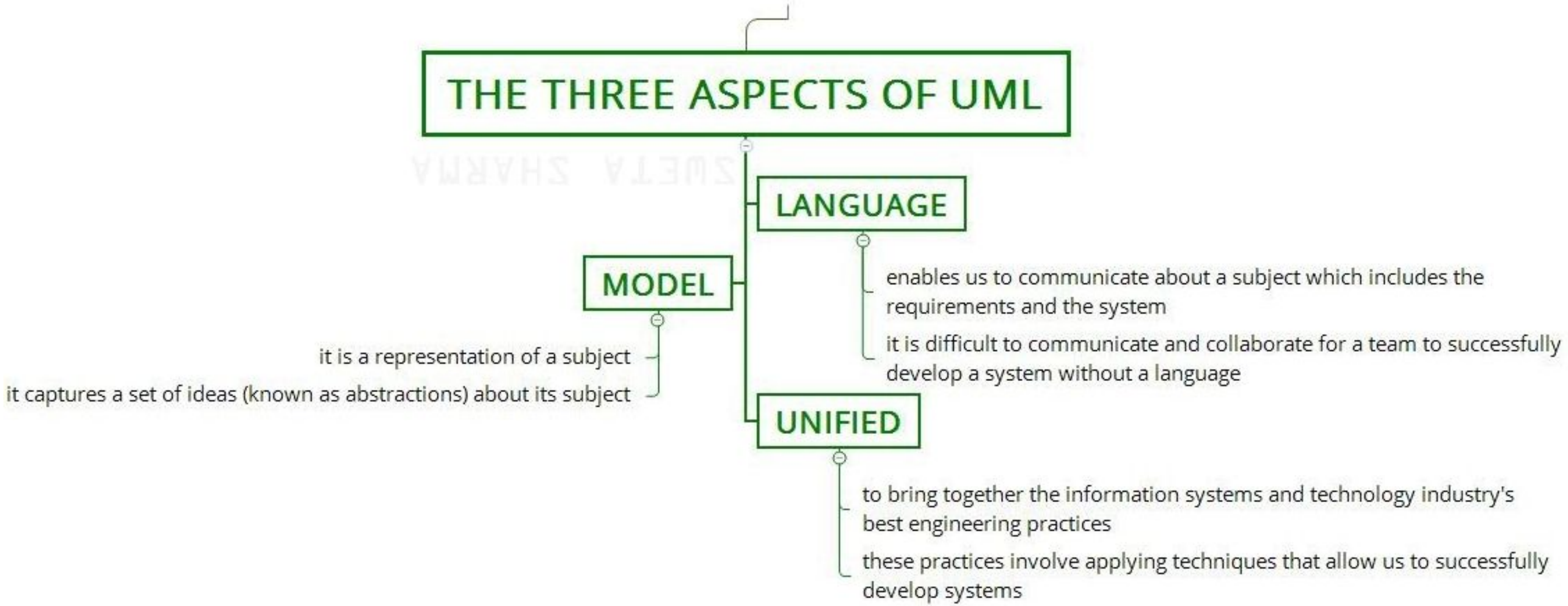
UML [A picture is worth a thousand words**]**

UML is a **standard language** for **specifying, visualizing, constructing, and documenting** the **artifacts of software systems**.

UML is a **pictorial language** used to make **software blueprints.**

A general purpose visual modeling language

Not **Limited just for, Software System. [eg.** Modelling WaterDamn, Aerospace Rocket Designing, Compiler Architecturing etc. and many more **]**



THE THREE ASPECTS OF UML

LANGUAGE

- enables us to communicate about a subject which includes the requirements and the system
- it is difficult to communicate and collaborate for a team to successfully develop a system without a language

MODEL

- it is a representation of a subject
- it captures a set of ideas (known as abstractions) about its subject

UNIFIED

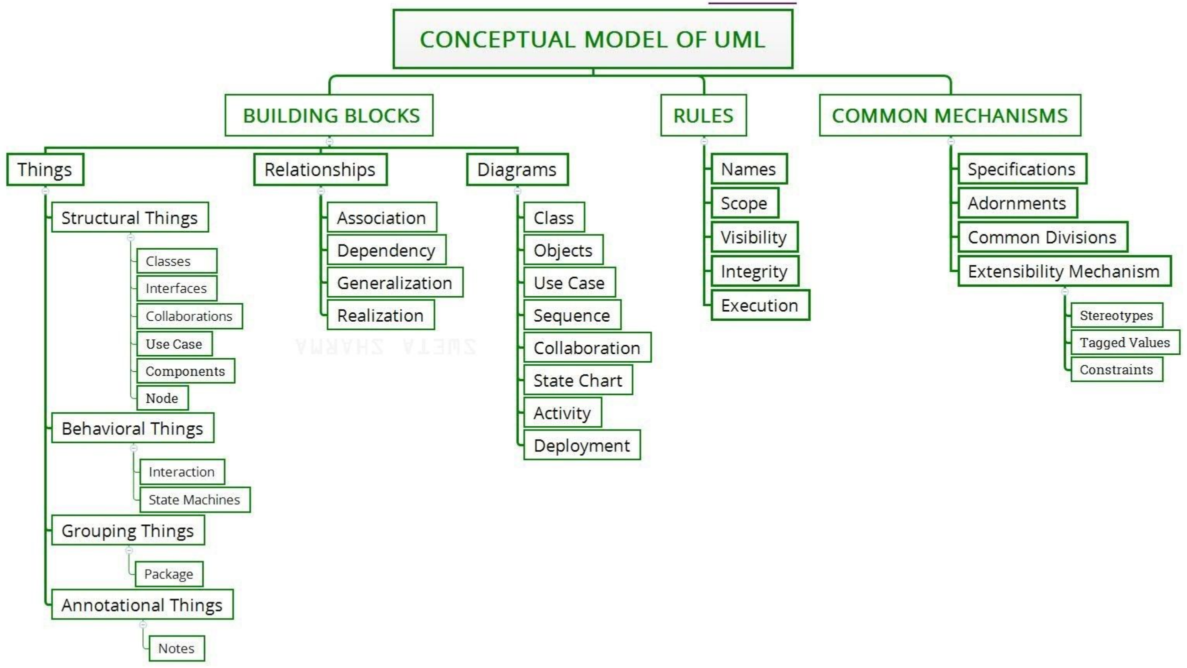
- to bring together the information systems and technology industry's best engineering practices
- these practices involve applying techniques that allow us to successfully develop systems

A Conceptual Model of UML

The **conceptual model** of UML can be **mastered** by learning the following **three** major elements –

- **UML building blocks**
- **Rules to connect the building blocks**
- **Common mechanisms of UML**

A Conceptual Model of UML



View of *System* In UML

- **USER MODEL VIEW**
  This view represent the system [**Product**] From the User’s [**Actors**] Perspective.

- **STRUCTURAL MODEL VIEW**
  Data and Functionality are Viewed from Inside the System.

- **BEHAVIOURAL MODEL VIEW**
  This part of OOA represent the Dynamic of behavioural aspect of the system.

- **IMPLEMENTATION MODEL VIEW**
  Implementation of Structural and behavioural Model is done in this Phase.

- **ENVIRONMENT MODEL VIEW**
  Structural and behavioural aspect of the Environment in which the system is to be implemented are represented

Generic Component Of O.O Analysis **Model**

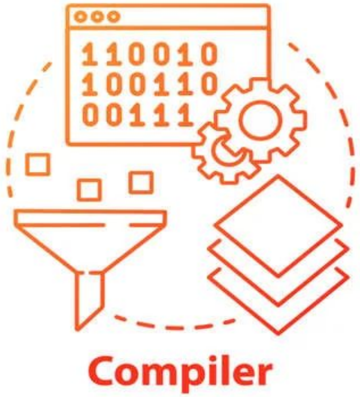
- Static Component
  - These are Structural in nature
  - Indicates Characteristics that holds throughout the Operational Life of an Application
- Dynamic Component
  - Focus on control and are Sensitive to timing and event processing.
  - They defines how one object interacts with other objects over time.

Object Modelling Techniques [OMT]

OMT is a **methodology** of object oriented analysis, design and implementation that focuses on creating a **model of objects from the real world** and then to **use this model** to **develop object–oriented software**

Object Modelling Techniques [OMT]

- OMT has proven itself **easy to understand, to draw and to use.**
- It is very successful in many application domains: telecommunication, transportation, compilers etc.



Phases of [OMT]

[The OMT methodology covers the full software development life cycle. The methodology has the following phase]

1. **Analysis**
2. **System design**
3. **Object design**
4. **Implementation**

THANK YOU :)