Chapter 3

Ch#3: System Modeling and Architectural Design

- 3.1 System Modeling(Self Study)
- 3.2 UML Diagram Types(Self Study)
- 3.3 Graphical Models(Self Study)
  - 3.3.1 Context Model(Self Study)
  - 3.3.2 Interaction Models(Self Study)
  - 3.3.3 Structural Models(Self Study)

Ch#4: Design and Implementation

- 4.1 Design and Implementation (Self Study)
- 4.2 An Object-Oriented Design Process(Self Study)
- 4.3 Context And Interaction Models (Self Study)
  - Process stages(Self Study)

System Modeling

System modeling is the process of developing abstract models of a system, with each model presenting a different view or perspective of that system.

System modelling helps the analyst to understand the functionality of the system and models are used to communicate with customers.

System modeling has now come to mean representing a system using some kind of **graphical notation**, which is now almost always based on notations in the **Unified Modeling Language (UML).**

UML Diagram Types

1. **Activity diagrams** which show the activities involved in a process or in data processing.
2. **Use case diagrams** which show the interactions between a system and its environment.
3. **Sequence diagrams** which show interactions between actors and the system and between system components.
4. **Class diagrams** which show the object classes in the system and the associations between these classes.
5. **State diagrams** which show how the system reacts to internal and external events.

Graphical Models

1. **Context Model:** in this model, context diagram and activity diagram is used.
2. **Interaction Model:** in this model, use-case diagram and sequence diagram is used.
3. **Structural Model:** in this model, class diagram and object diagram is used.

Context Models

Context models are used to illustrate the operational context of a system they show what lies outside the system boundaries.

Social and organizational concerns may affect the decision on where to position system boundaries.

Architectural models show the system and its relationship with other systems.

Activity diagrams are intended to show the activities that make up a system process and the flow of control from one activity to another.

The start of a process is indicated by a filled circle; the end by a filled circle inside another circle.

Rectangles with round corners represent activities, that is, the specific subprocesses that must be carried out.

You may include objects in activity charts.

UML activity diagrams are used to define major **business** process models.

Interaction Models

Modeling user interaction is important as it helps to identify user requirements.

Modeling system-to-system interaction highlights the communication problems that may arise.

Modeling component interaction helps us understand if a proposed system structure is likely to deliver the required system performance and dependability.

Use case diagrams and sequence diagrams may be used **for interaction modeling**.

Use Case Modeling

Use cases were developed originally to support requirements elicitation and now incorporated into the UML.

Each use case represents a discrete task that involves external interaction with a system.

Actors in a use case may be people or other systems.

Use case diagrams give a fairly simple overview of an interaction so you have to provide more detail to understand what is involved. This detail can either be a simple textual description, a structured description in a table, or a sequence diagram.

Sequence Diagrams

Sequence diagrams are **part of the UML** and are used to model the interactions between the actors and the objects within a system.

A sequence diagram shows the sequence of interactions that take place during a particular use case or use case instance.

The objects and actors involved are listed along the top of the diagram, with a dotted line drawn vertically from these.

Interactions between objects are indicated by annotated arrows.

Structural Models

Structural models of software display the organization of a system in terms of the components that make up that system and their relationships.

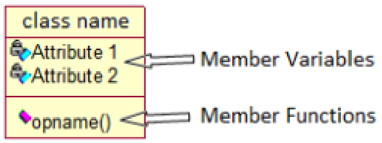
Structural models may be static models, which show the structure of the system design, or dynamic models, which show the organization of the system when it is executing.

You create structural models of a system when you are discussing and designing the system architecture.

Class Diagrams

Class diagrams are used when developing an object-oriented system model to show the classes in a system and the associations between these classes.

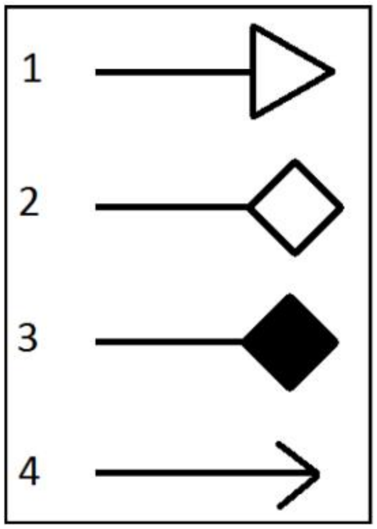
Class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modeling Language (UML).



Class Diagram Notations: UML class is represented by the diagram shown below that is divided into three sections.

1. The first section is used to **name the class**.
2. The second section is used to show the **attributes** / member **variable** of the class.
3. The third section is used to describe the **operations performed by the class**.

Types of Relationship Between the Classes



1. Shows **Generalization** between two classes to maintain parent-child relationship. (***is-a relation***)
2. Shows **Aggregation** between two classes to maintain containership. (***has-a relation***)
3. Shows **Composition** between two classes to maintain containership. (***has-a relation***)
4. Shows **Association** between two classes to maintain a simple connectivity.

Difference between Aggregation and Composition: (Has-a Relationship)

1. In **aggregation** contained class exist even if main class is deleted. Eg: Car has Engine, If car is broken still engine exist and can be used in another car.
2. In **composition** contained class live or die on the basis of main class. Eg: Folder has file, If you delete a folder all files will be deleted as well.

Generalization: (Is-a Relationship)

Generalization is used in class diagrams to deal with most powerful concept of object orientation that is **Inheritance**. Generalization is drawn between two classes to show Parent-Child relation.

In modeling systems, it is often useful to examine the classes in a system to see if there is scope for generalization. If changes are proposed, then you do not have to look at all classes in the system to see if they are affected by the change.

The lower-level classes are subclasses that inherit the attributes and operations from their superclass. These lower-level classes then add more specific attributes and operations.

Chapter 4

Design and Implementation

Software design and implementation is the stage in the software engineering process at which an **executable software system is developed**.

Software design and implementation activities are invariably inter-leaved.

- Software design is a **creative** activity in which you identify software components and their relationships, based on a customer's requirements.
- Implementation is the process of realizing the design as a program.

Object-Oriented Design Process

Structured object-oriented design processes involve developing a number of different system models.

They require a lot of effort for development and maintenance of these models and, for small systems, this may **not be cost-effective**.

for large systems developed by different groups design models are an important communication mechanism.

Process Stages

There are a variety of different object-oriented design processes that depend on the organization using the process.

Common activities in these processes include:

- Define the context and modes of use of the system.
- Design the system architecture.
- Identify the principal system objects.
- Develop design models.
- Specify object interfaces.

Context and Interaction Models

A system context model is a structural model that demonstrates the other systems in the environment of the system being developed.

An interaction model is a dynamic model that shows how the system interacts with its environment as it is used.