# Chapter 7

# Object Oriented Analysis and Design

## 7.1 Object-Oriented Development Life Cycle:

1. **The Object-Oriented Analysis Phase**
    1. Requirements Model
    2. Object Model
2. **The Object-Oriented Design Phase**
    1. Result is a plan of *how* the system will do what the Requirements Analysis asks for
3. **The Construction Phase**
    1. Coding and testing
    2. Deployment and user training
4. **The Object-Oriented Testing Phase**
    1. Complete the unit testing of individual classes and programs, then system testing.
5. **The Maintenance Phase**
    1. Bug fixes
    2. Enhancements

## 7.2 The Unified Modeling Language

UML - Unified Modeling language . UML is a modeling language . Developed by Grady Booch, James Rumbaugh and Ivar Jacobson. Accepted as a standard by the Object Management Group (OMG), in 1997.

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

Visualizing - *a picture is worth a thousand words; a graphical notation articulates and unambiguously communicates the overall view of the system* .

Specifying - *UML provides the means to model precisely, unambiguously and completely, the system in question.*

Constructing - *models built with UML have a "design" dimension to it; these are language independent and can be implemented in any programming language.*

Documenting - *every software project involves a lot of documentation - from the inception phase to the deliverables.*

Use graphical notation to communicate more clearly than natural language and code.

UML is *not* dependent on any one language or technology.

## 7.3 Use-Case Modeling

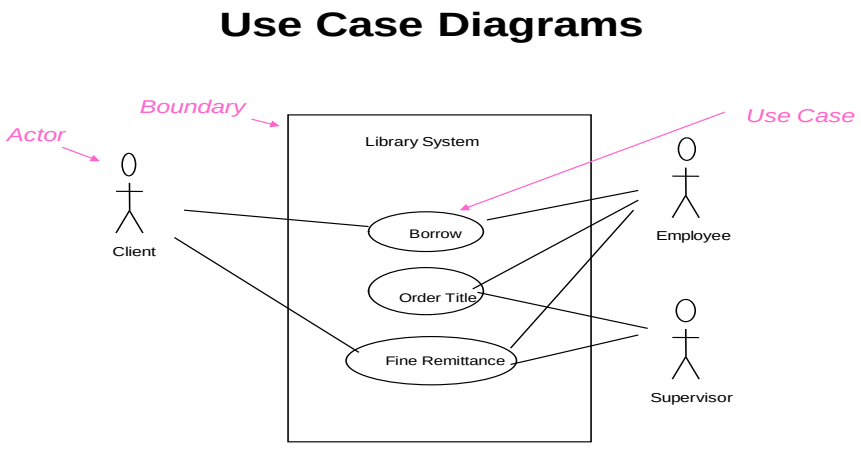
- **The User Interaction or Use Case Model**- describes the boundary and interaction between the system and users.
- Mainly used for capturing user requirements
- Work like a **contract** between end user and software developers

**Use Case Diagram (core components)**

**Actors:** A role that a user plays with respect to the system, including human users and other systems.

**Use case:** A set of scenarios that describing an interaction between a user and a system, including alternatives.


**System boundary**: rectangle diagram representing the boundary between the actors and the system.




- A generalized description of how a system will be used.
- Provides an overview of the intended functionality of the system

Association: communication between an actor and a use case; Represented by a solid line.



Generalization: relationship between one general use case and a special use case. Represented by a line with a triangular arrow head toward the parent use case.

Include: a dotted line labeled <<include>> beginning at base use case and ending with an arrows pointing to the include use case.

<<include>>

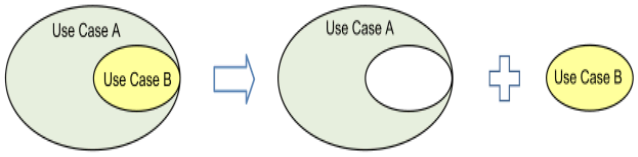


Extend: a dotted line labeled <<extend>> with an arrow toward the base case. The extending use case may add behavior to the base use case. The base class declares "extension points".

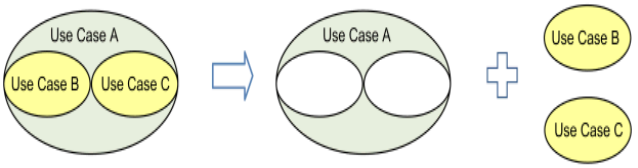
<<extend>>

The **include** relationship could be used:

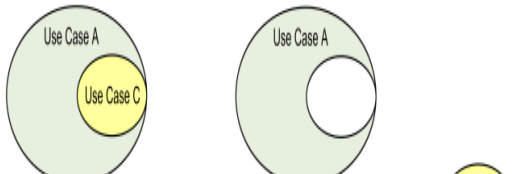
- to simplify large use case by splitting it into several use cases,
- to extract **common parts** of the behaviors of two or more use cases.

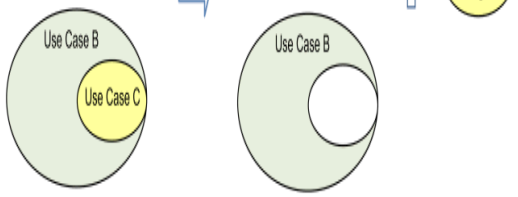



*Use case B is extracted from larger use case A into a separate use case.*

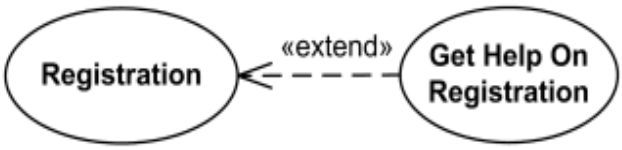



*Use cases B and C are extracted from larger use case A into separate use cases.*

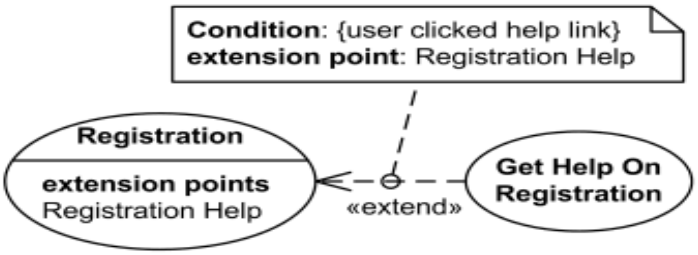



*Use case C is extracted from use cases A and B to be reused by both use cases using UML include relationship.*

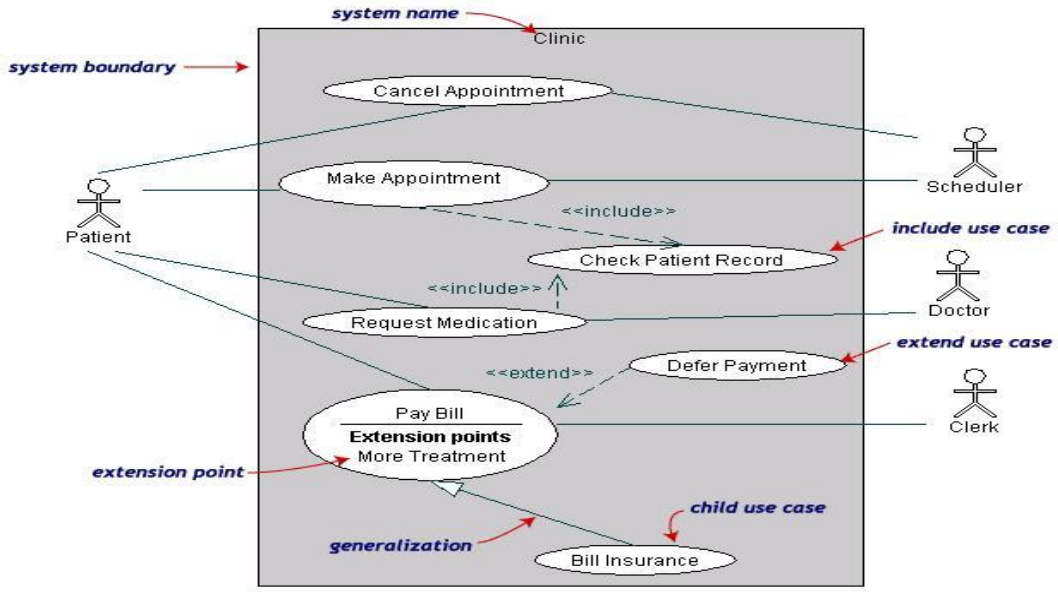



***Registration** use case is complete and meaningful on its own.*
*It could be extended with optional **Get Help On Registration** use case.*




*Registration use case is conditionally extended by Get Help On Registration use case in extension point Registration Help.*

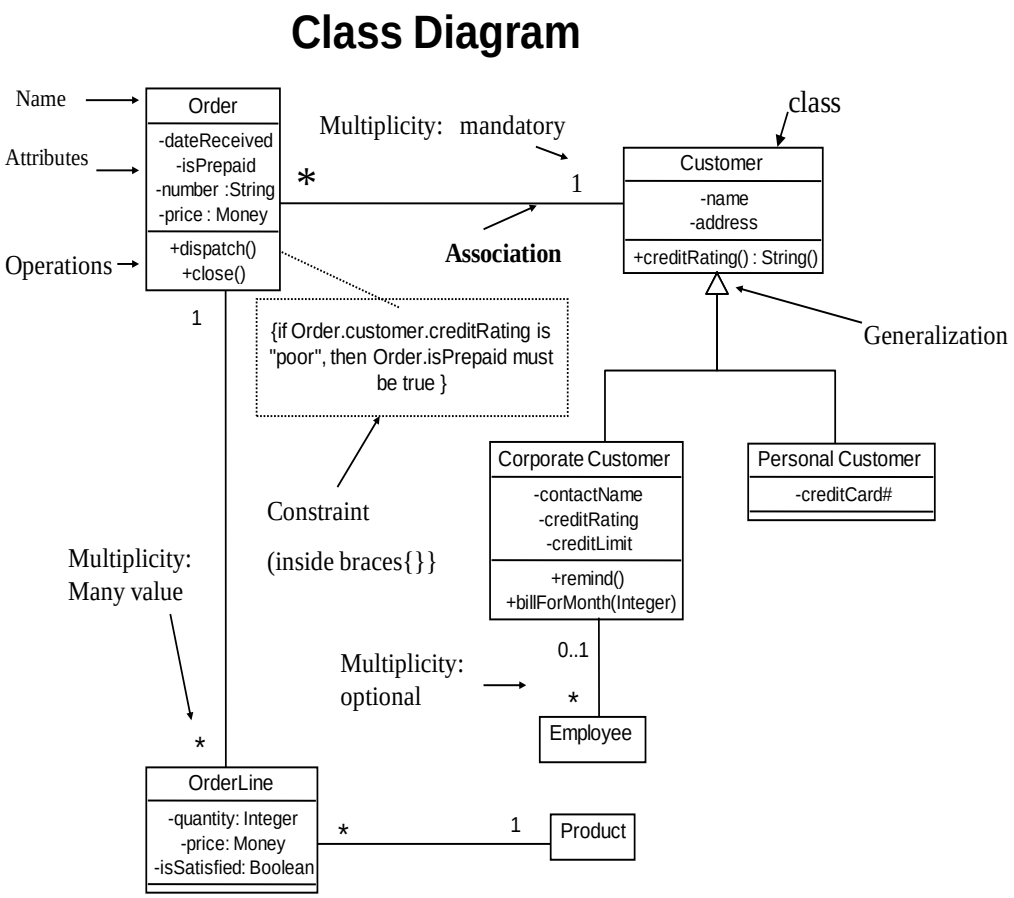
# Use Case Diagrams(cont.)




- **Pay Bill** is a parent use case and **Bill Insurance** is the child use case. (generalization)
- Both **Make Appointment** and **Request Medication** include **Check Patient Record** as a subtask.(include)
- The **extension point** is written inside the base case **Pay bill**; the extending class **Defer payment** adds the behavior of this extension point. (extend)

## 7.4. Object Modeling: Class Diagrams

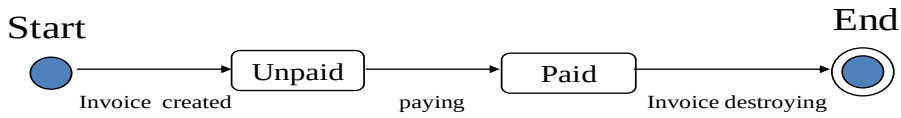
- Used for describing structure and behavior in the use cases
- Provide a conceptual model of the system in terms of entities and their relationships
- Used for requirement capture, end-user interaction
- Detailed class diagrams are used for developers




## 7.5. Dynamic Modeling: State Diagrams

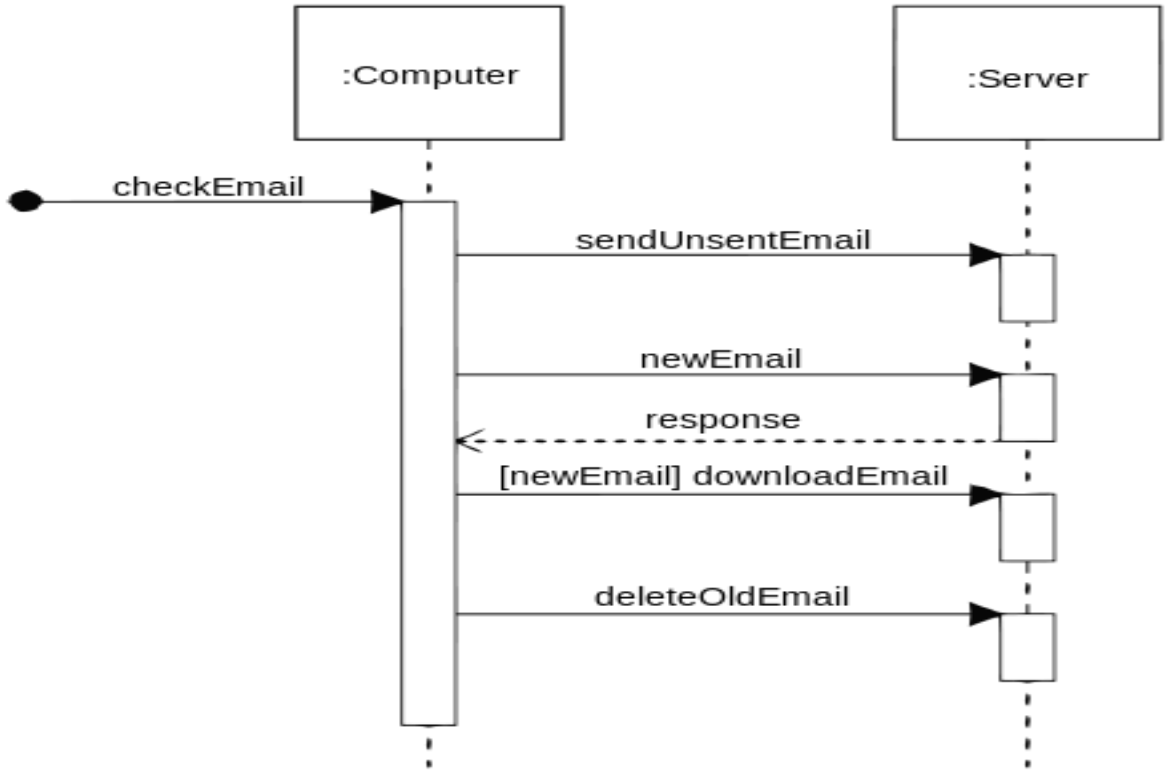
- State diagrams are used to detail the changes of state an object can go through in the system.
- They show how an object moves from one state to another and the rules that govern that change.
- State diagrams typically have a start and end condition.

## State Diagrams
**(Billing Example)**



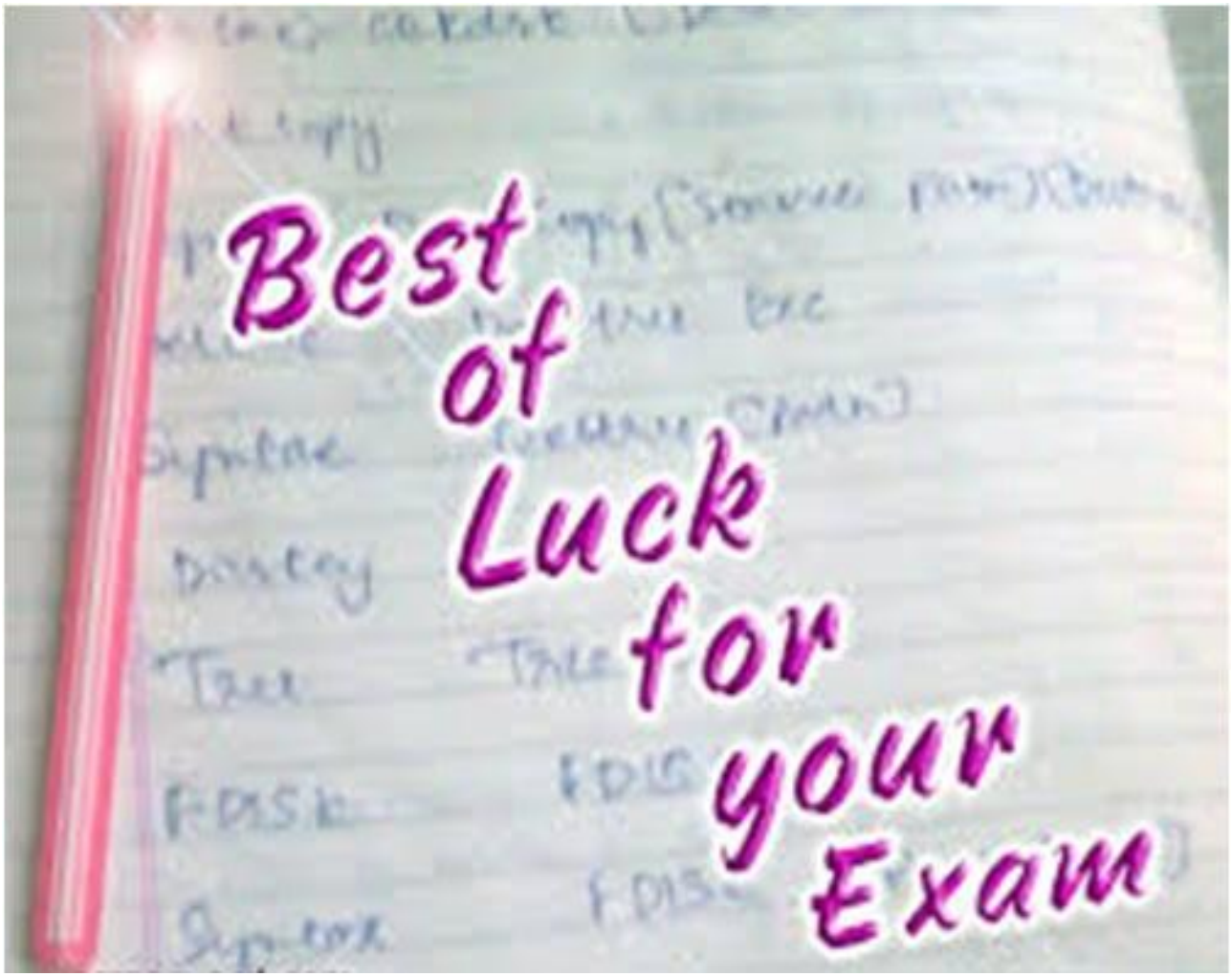

## 7.6 Dynamic Modeling: Sequence Diagramming

- A **Sequence diagram** is an interaction diagram that shows how processes operate with one another and in what order.
- It is a construct of a Message Sequence Chart.
- A sequence diagram shows object interactions arranged in time sequence.
- Sequence diagrams are sometimes called **event diagrams** or **event scenarios**.
- This allows the specification of simple runtime scenarios in a graphical manner.


:Computer
:Server
checkEmail
sendUnsentEmail
newEmail
response
[newEmail] downloadEmail
deleteOldEmail

## Assignment VII

1. Explain object oriented development cycle. Write down the advantage of using OOAD.
2. Explain the Unified Modeling Language with example.
3. What are UML diagrams and were they are used? Explain use-case diagram with suitable example.
4. What is UML? Explain the types of UML diagrams.
5. Differentiate between state diagrams and sequence diagrams in object oriented analysis and design.
6. Differentiate between object modeling and dynamic modeling
7. Explain about class Diagram with suitable example.
8. Draw use- case diagram for university registration system.
9. Draw a use case diagram for Library management system.
10. What are the differences between Structured System and Object Oriented System?


Best
of
Luck
for
your
Exam