SAD

Lecture 3

Eng. Joud Khattab

Overview

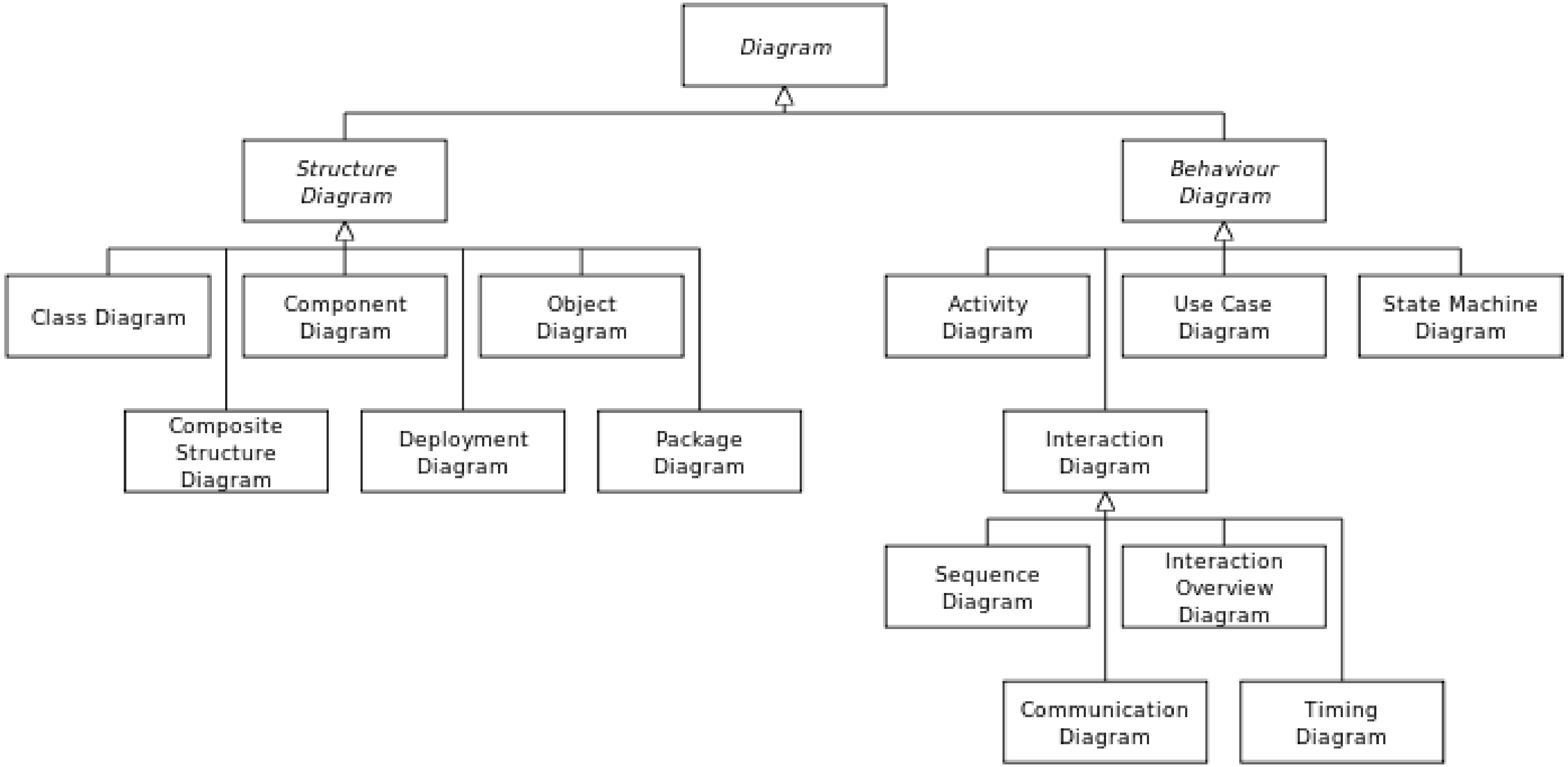
- Unified Modeling Language (UML)
  - Definition
  - Kinds
- Use Case Diagram
  - Purpose
  - Notation
  - How to draw
- Use Case Specification

UML

UML (Unified Modeling Language)

- A standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.
- **Pictorial** language used to make software blueprints.
- It is not a programming language but tools can be used to generate code in various languages using UML diagrams.
- It has a direct relation with object oriented analysis and design.
- UML diagrams are not only made for developers but also for business users, common people, and anybody interested to understand the system.

UML (Unified Modeling Language)



UML Kinds

- UML specification defines two major kinds of UML diagram:
  1. **structure diagrams**
  2. **behavior diagrams**

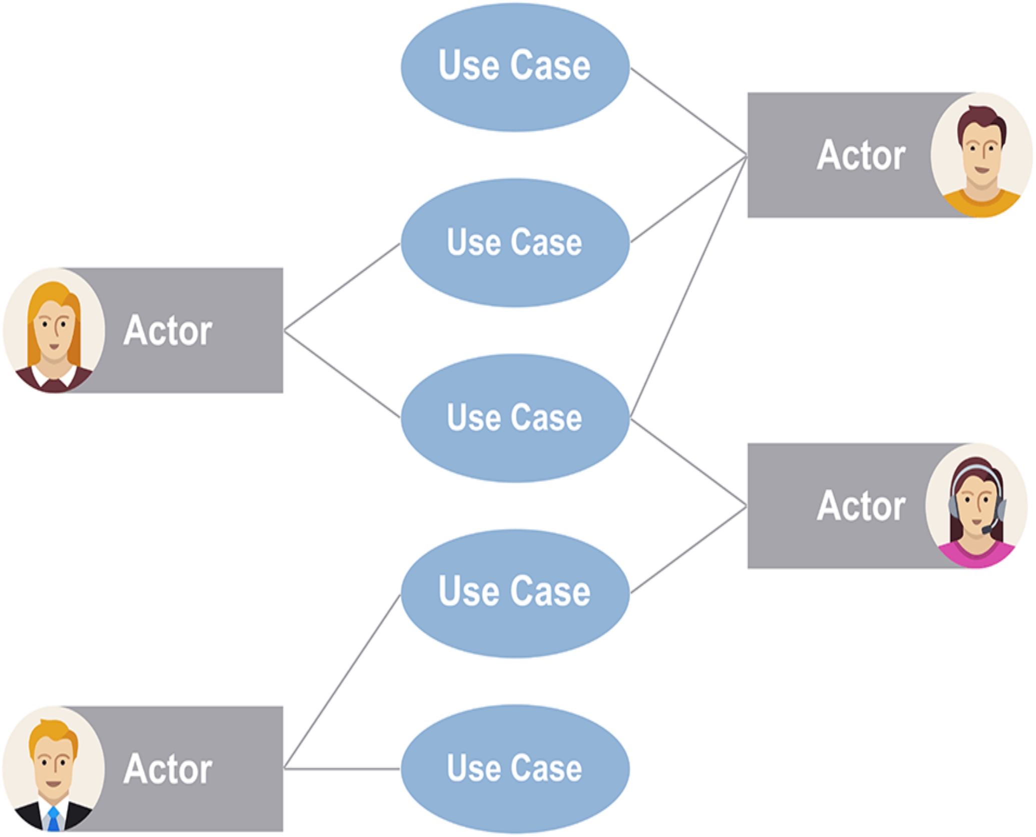
- **Structure diagrams**
  - show the **static structure** of the system and its parts on different abstraction and implementation **levels** and how they are related to each other.
  - The elements in a structure diagram represent the meaningful concepts of a system, and may include abstract, real world and implementation concepts.

- **Behavior diagrams**
  - show the **dynamic behavior** of the objects in a system, which can be described as a series of changes to the system over **time**.

USE CASE DIAGRAM

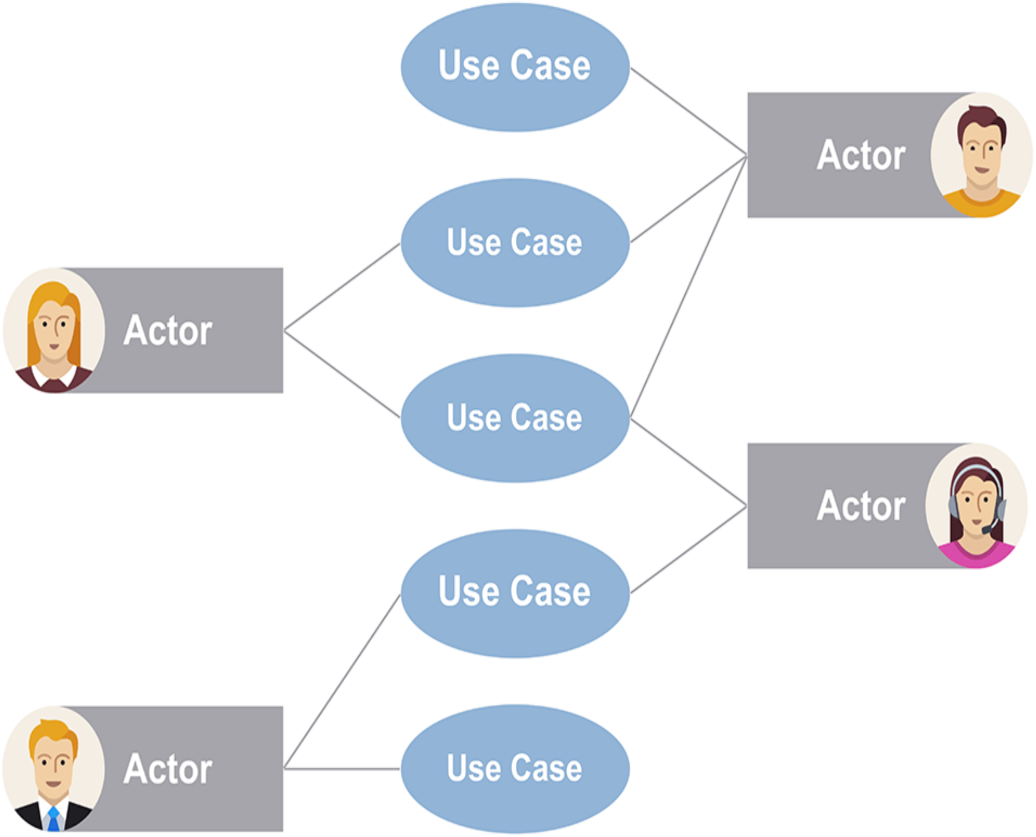
Use Case Diagram

- Use case diagrams are used to gather the requirements of a system including internal and external influences.
- These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared.



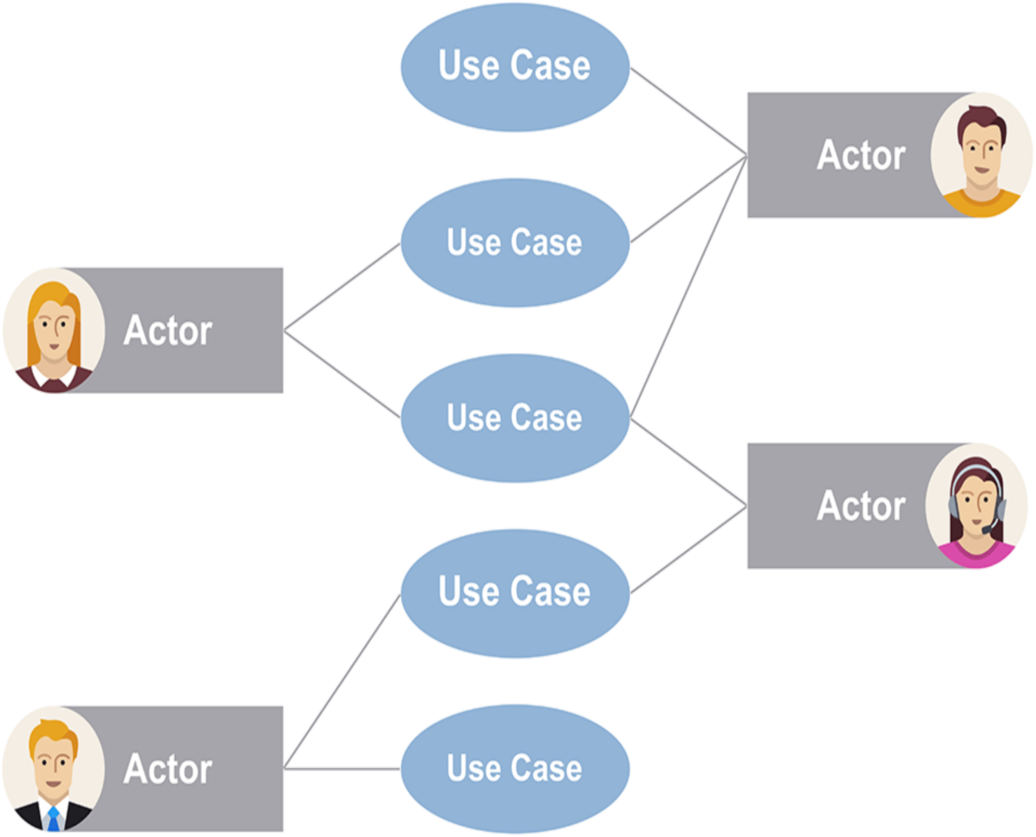
Use Case Diagram

- **Purposes**:
  - Used to gather the requirements of a system.
  - Used to get an outside view of a system.
  - Identify the external and internal factors influencing the system.
  - Show the interaction among the requirements are actors.



Use Case Diagram

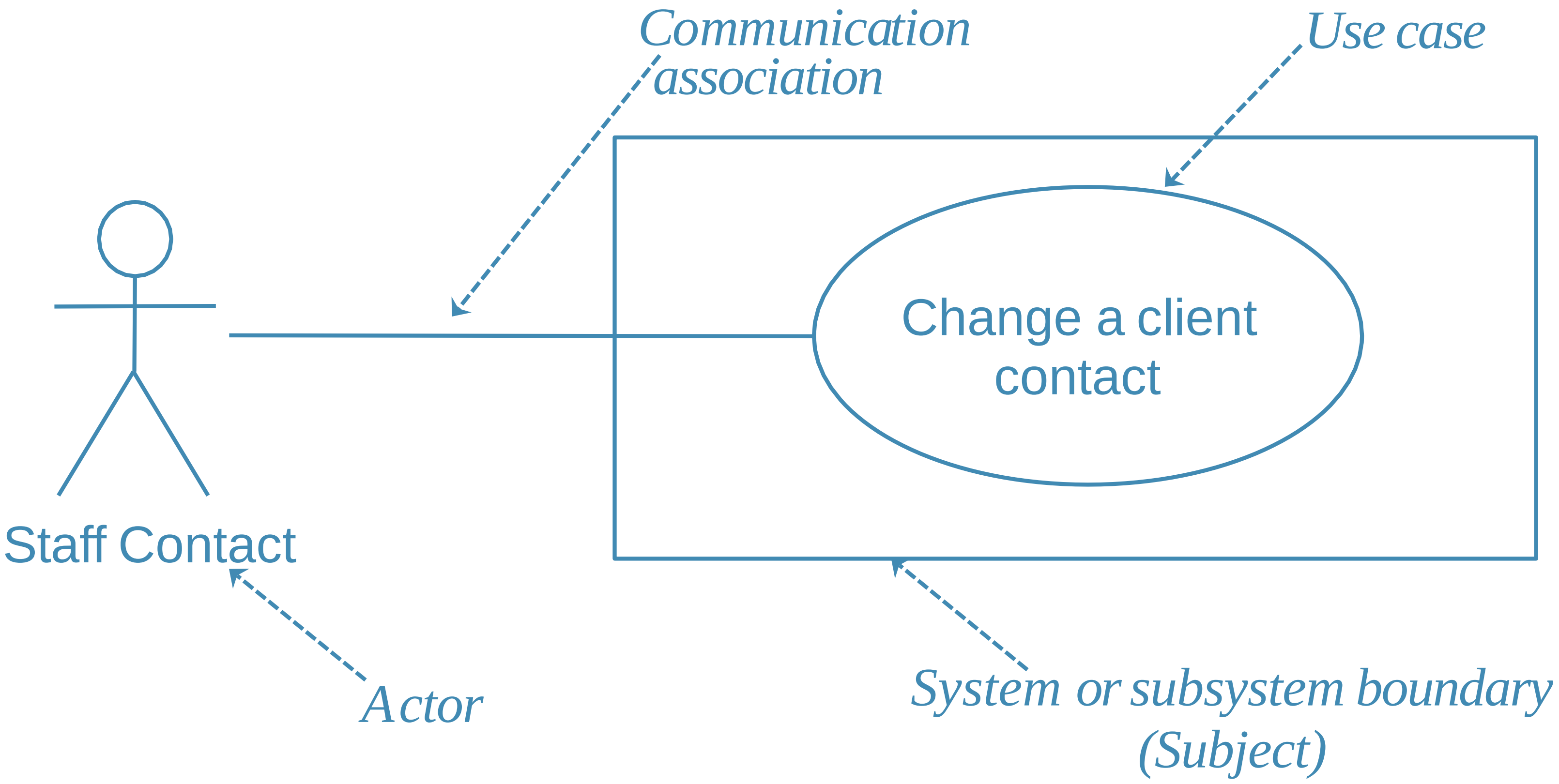
- **Use case diagrams** are **behavior diagrams** used to describe a set of actions (**use cases**) that some system or systems (**subject**) should or can perform in collaboration with one or more external users of the system (**actors**) to provide some observable and valuable results to the actors or other stakeholders of the system(s).



How to Draw a Use Case Diagram?

- When we are planning to draw a use case diagram, we should identify:
  - Functionalities (represented as use case).
  - Actors (human user, internal applications, external applications).
  - Relationships (among the use cases and actors).

How to Draw a Use Case Diagram?

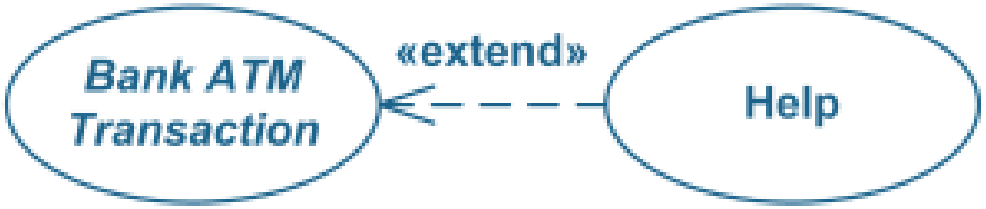

Communication association
Use case
Change a client contact
Staff Contact
Actor
System or subsystem boundary (Subject)

Use Case Diagram Guidelines

- The name should be chosen in such a way so that it can identify the functionalities performed.
- Give a suitable name for actors.
- Show relationships and dependencies clearly in the diagram.
- Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.
- Use notes whenever required to clarify some important points.

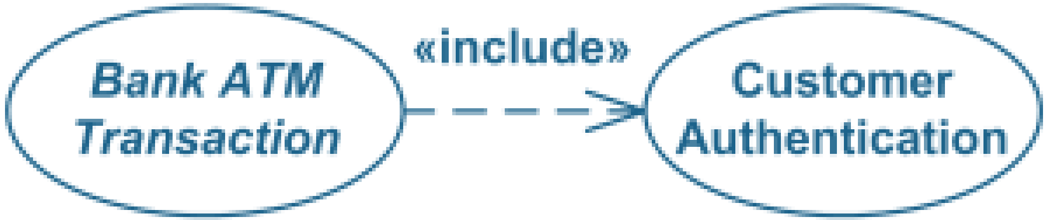
Use Case Diagram Guidelines

- Extend relationship:
  - Used when one use case provides additional functionality that may be required in another use case.
  - There may be multiple ways of extending a use case, which represent variations in the way that actors interact with the use case.



Use Case Diagram Guidelines

- Include relationship:
  - Used when one use case always includes the functionality of another use case.
  - A use case may include more than one other.
  - Can be used to separate out a sequence of behavior that is used in many use cases.
  - Should not be used to create a hierarchical functional decomposition of the system.

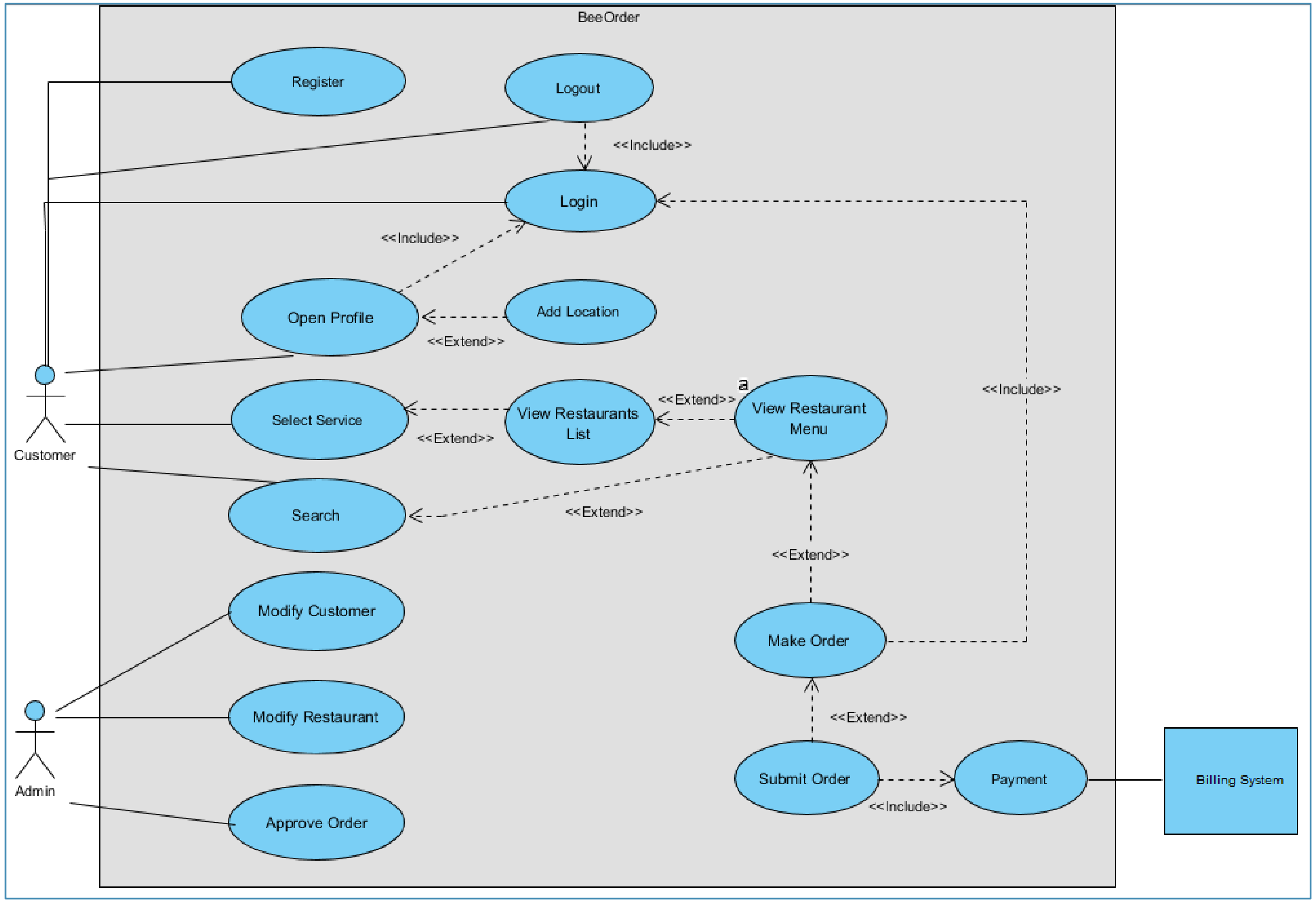


Use Case Diagram Guidelines

- Generalization relationship:
  - Use Case Generalization: shows that one use case provides all the functionality of the more general use case and some additional functionality.
  - Actor Generalization: shows that one actor can participate in all the associations with use cases that the more general actor can plus some additional use cases.

Practical Exercise

Bee Order Use Case



USE CASE SPECIFICATIONS

Use Case Specifications

Title	
Actor	
Brief Description	
Pre-condition	
Post-condition	
Flow of events	
Critical Scenario	
User Interface prototype	

What we have learned

- The use case modeling activity is part of the requirements workflow.
- Use case modeling is another form of requirements engineering that proceeds as follows:
  - Find the system boundary.
  - Find actors.
  - Find use cases.
- Actors are roles played by things external to the system that interact directly with the system.
- Use cases are functions that the system performs on behalf of, and to deliver benefit to, specific actors. You can find use cases by considering how each actor interacts with the system.

What we have learned

- The use case diagram shows:
  - The system boundary.
  - Actors.
  - Use cases.
  - Interactions.
- Use case modeling is most appropriate for systems that:
  - Are dominated by functional requirements.
  - Have many types of users.
  - Ave many interfaces to other systems.

HOME WORK

Write 3 use case specification for bee order use cases from your choice