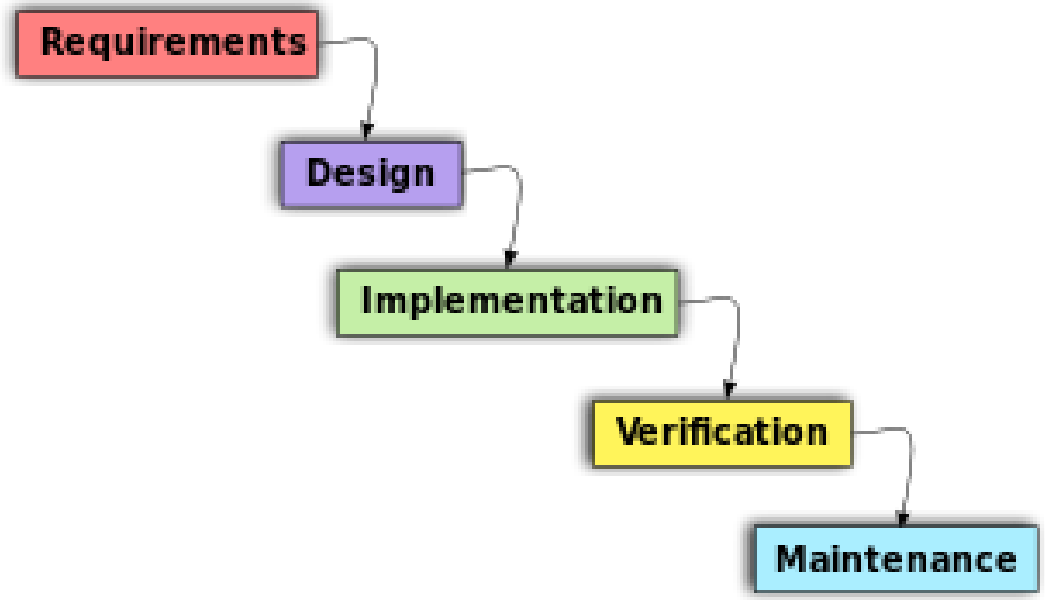
Object Oriented Analysis and Design

OOAD

- Object-Oriented Analysis And Design (OOAD) It's a structured method for analyzing, designing a system by applying the object-orientated concepts, and developing a set of graphical system models during the development life cycle of the software.



UML

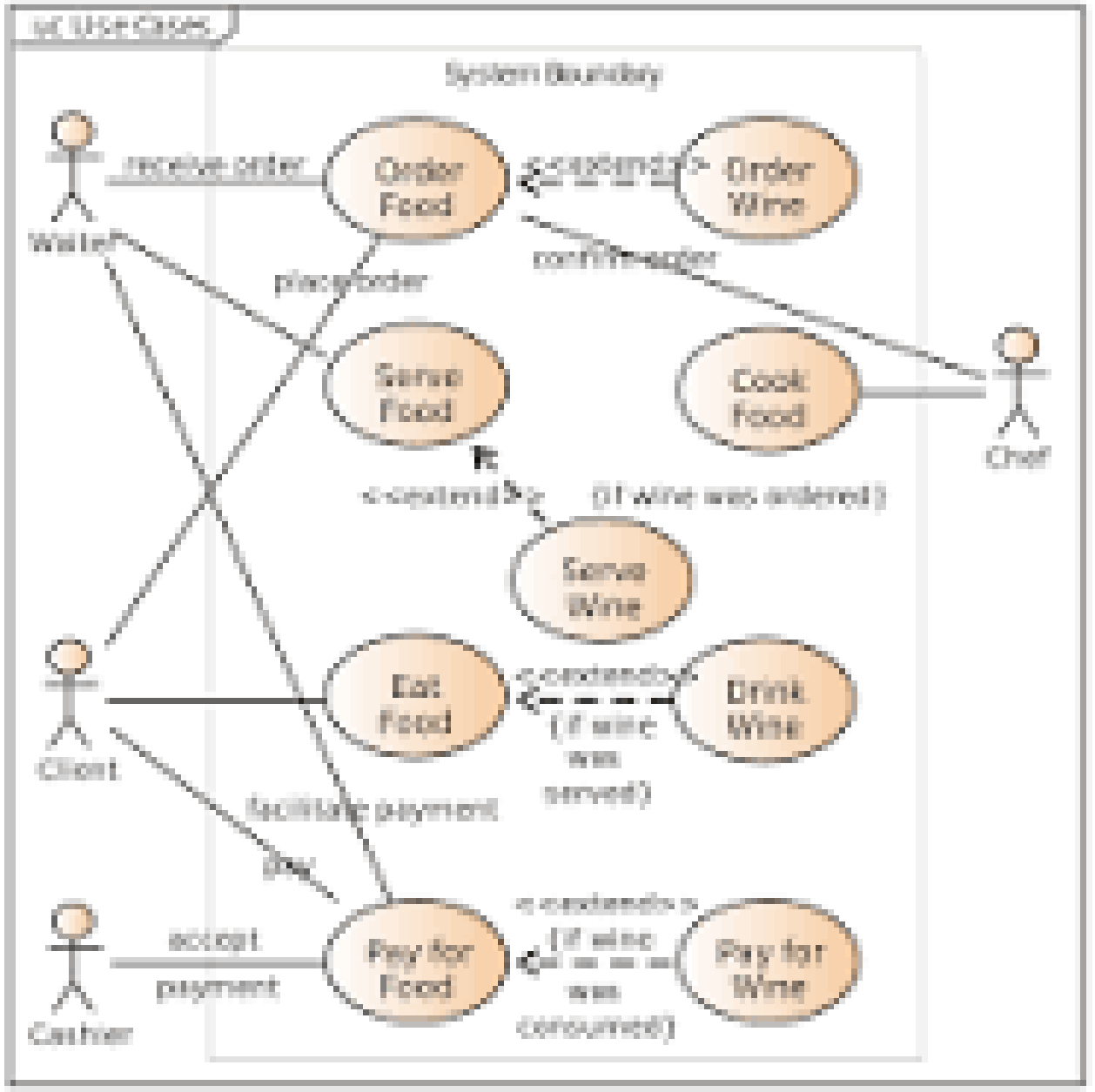
- The **Unified Modeling Language** (UML) is a general-purpose, developmental modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system.

UML Diagrams

1. **Structure Diagrams –** Used to model the static structure of a system.
   - **class diagram, package diagram, object diagram, deployment diagram etc.**
2. **Behavior diagram –** Used to model the dynamic change in the system over time. They are used to model and construct the functionality of a system.
   - **Use case diagrams, Interaction diagrams, Activity diagrams and State diagrams.**

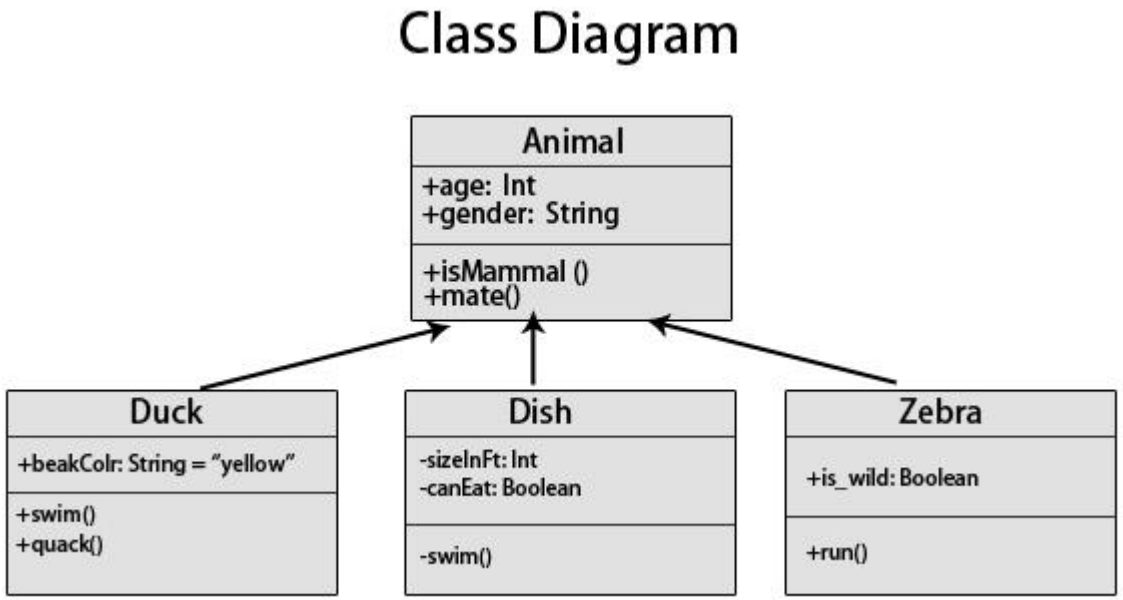
Use Case Diagram

- In Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system.



Class Diagram

- The *class diagram* depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them.



Sequence Diagram

- The sequence diagram *represents the flow of messages in the system* and is also termed as an event diagram.

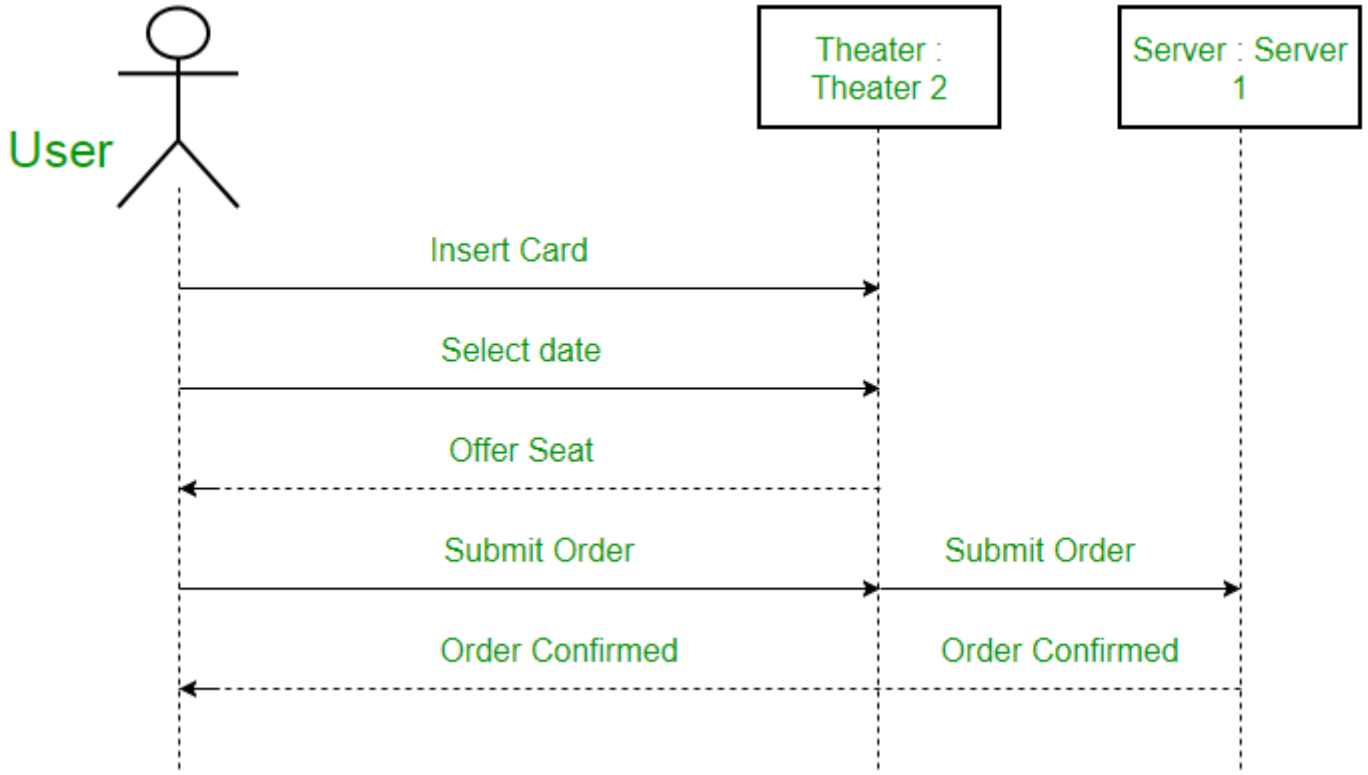
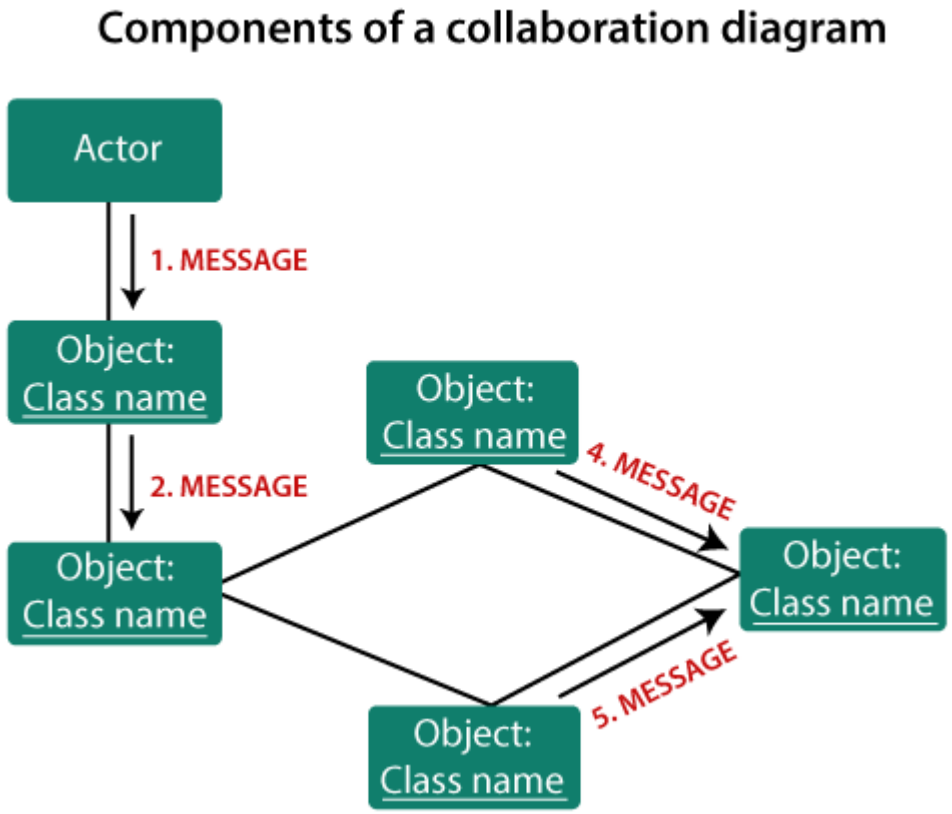


Figure – a Sequence diagram for Movie Booking

Collaboration Diagram

- The *collaboration diagram* is used to show the relationship between the objects in a system.



State Diagram

- The basic purpose of a *state diagram* is to portray various changes in *state* of the class and not the processes or commands causing the changes.

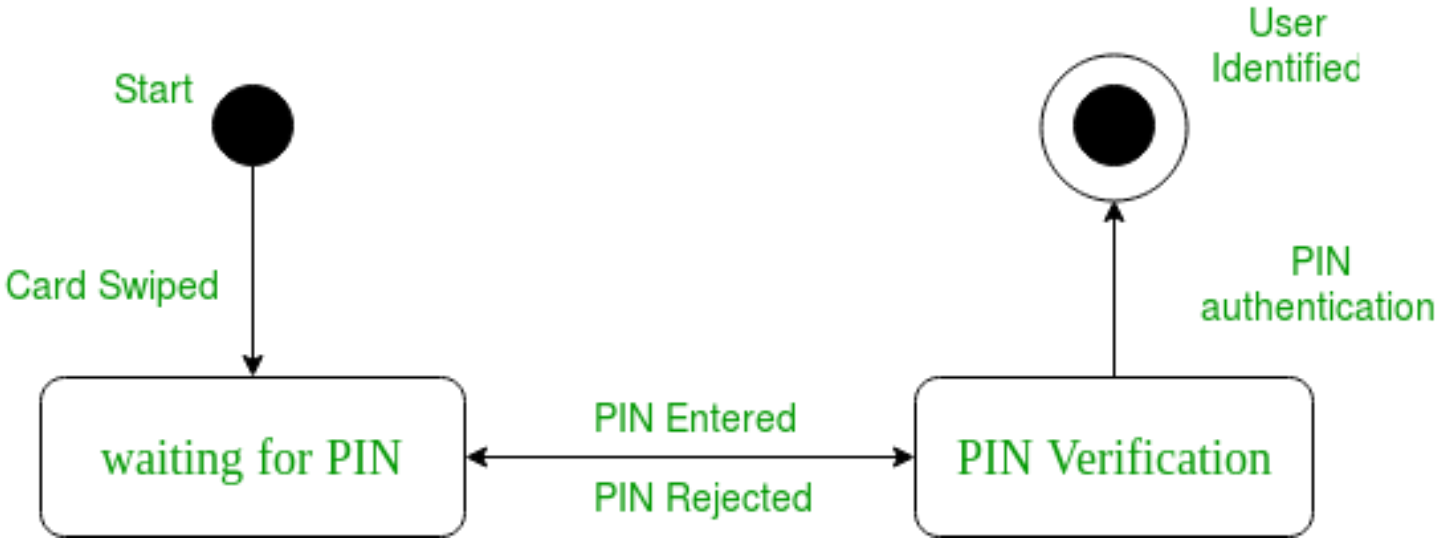
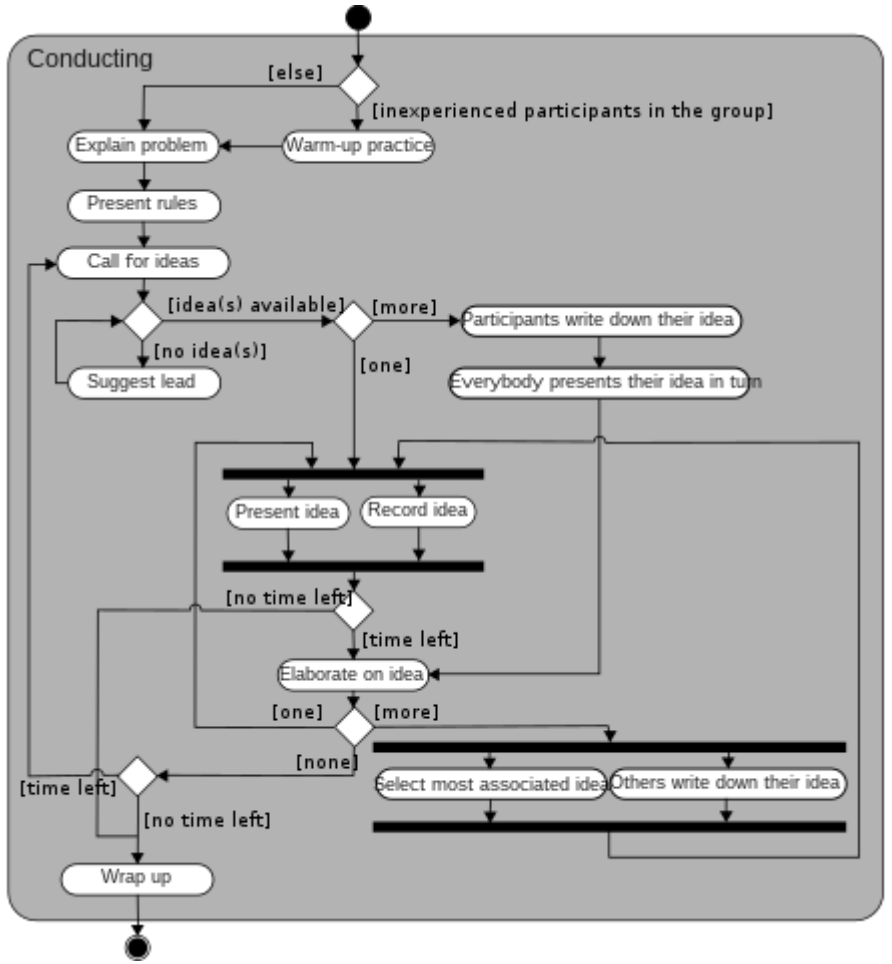


Figure – a state diagram for user verification

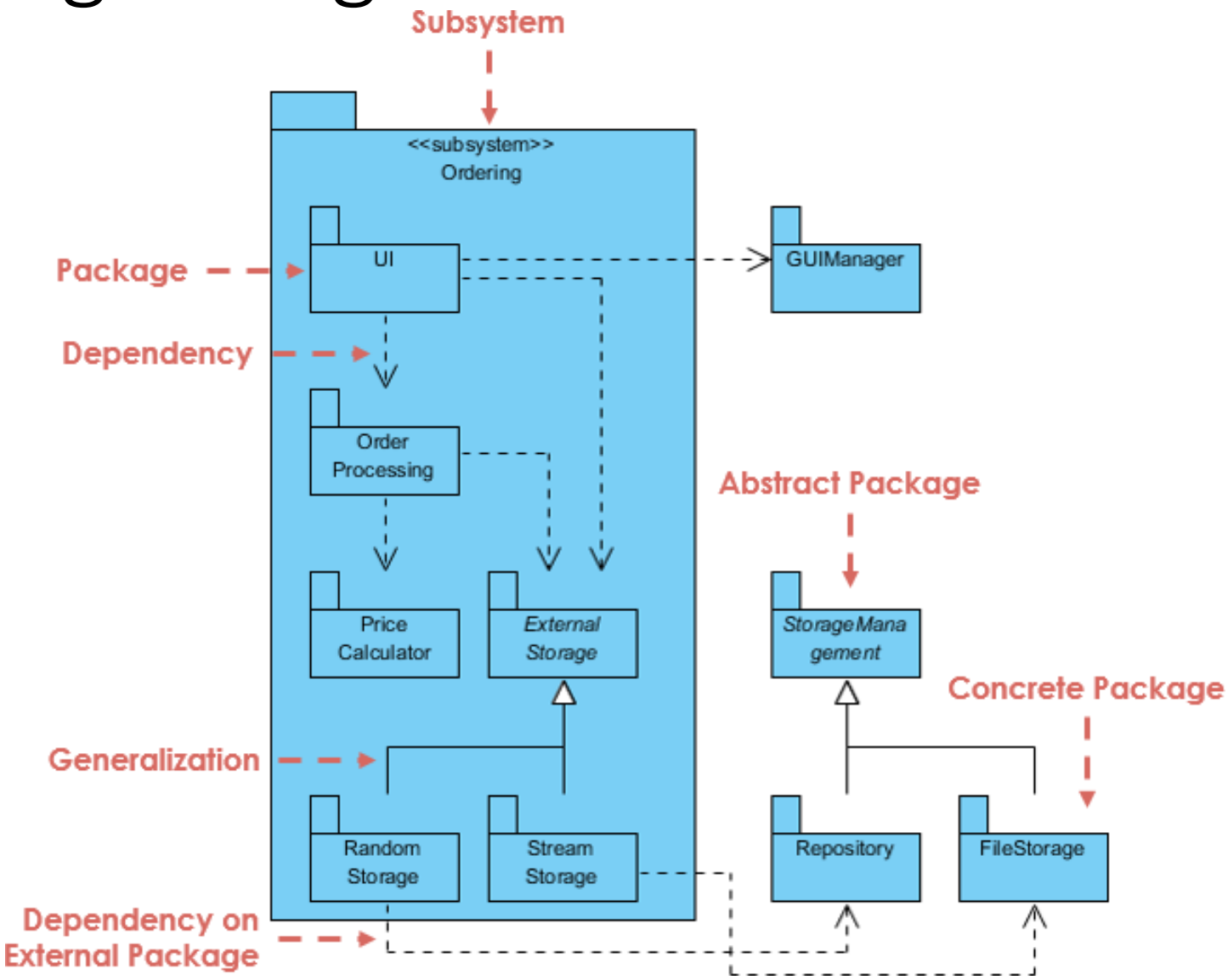
Activity Diagram

- Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system.
- Activity diagram is essentially an advanced version of flow chart that modeling the flow from one activity to another activity



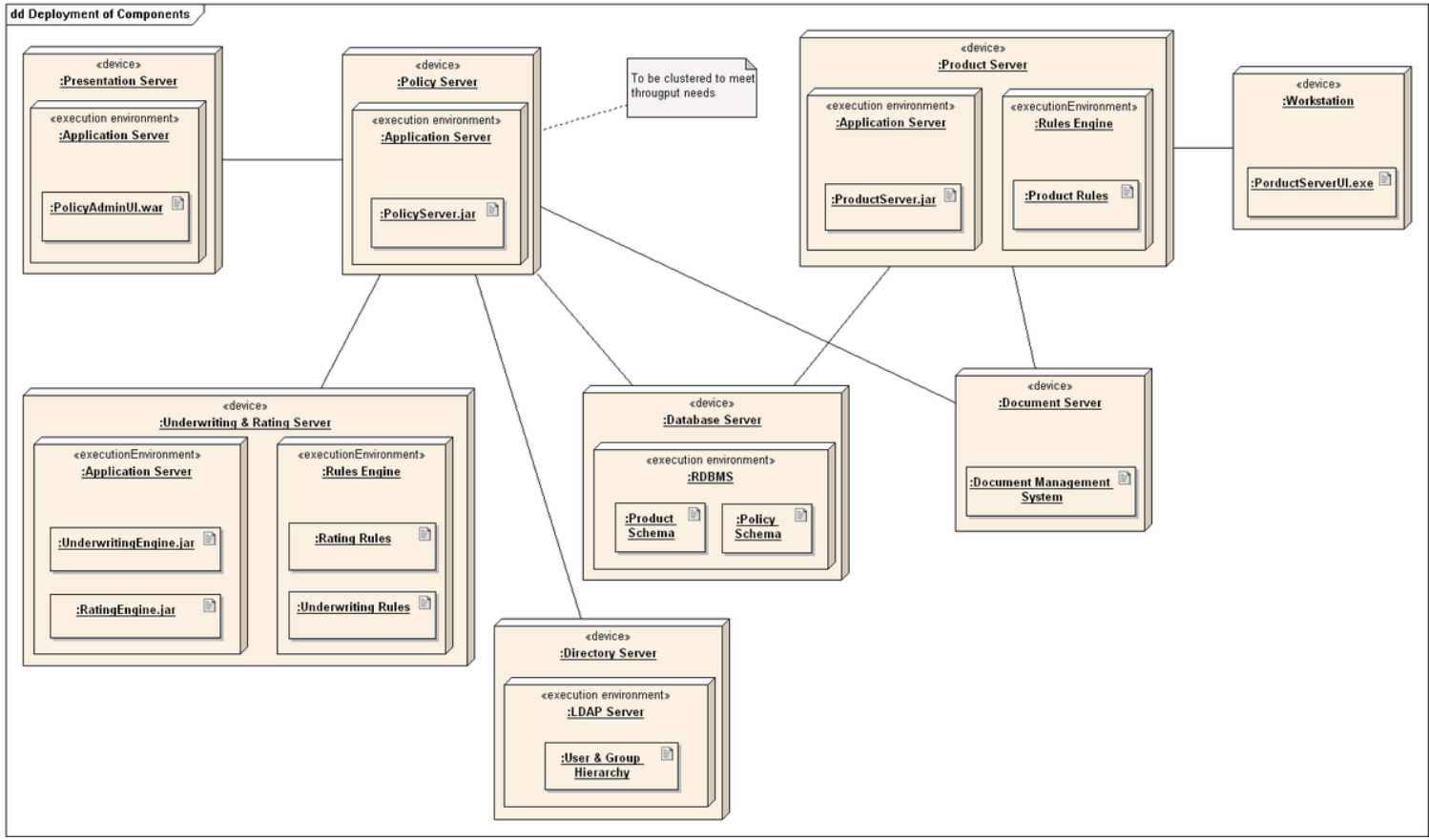
Package Diagram

- *Package diagram*, a kind of structural diagram, shows the arrangement and organization of model elements in middle to large scale project.



Deployment Diagram

- The deployment diagram *visualizes the physical hardware on which the software will be deployed*



Implementation

- Implement the application using any programming language.

Testing

- Evaluate the implemented model by writing test cases.

Design Patterns

- A design pattern provides a general reusable solution for the common problems that occur in software design.
- The pattern typically shows relationships and interactions between classes or objects.
- The idea is to speed up the development process by providing well-tested, proven development/design paradigms.
- Design patterns are programming language independent strategies for solving a common problem.
- A design pattern represents an idea, not a particular implementation.
- By using design patterns, you can make your code more flexible, reusable, and maintainable.

Refine the System and Test Again

- Refine the developed application and test the implemented application.