SOFTWARE DESIGN & ANALYSIS (Week-4)

USAMA MUSHARAF

MS-CS (Software Engineering)

LECTURER (Department of Computer Science)

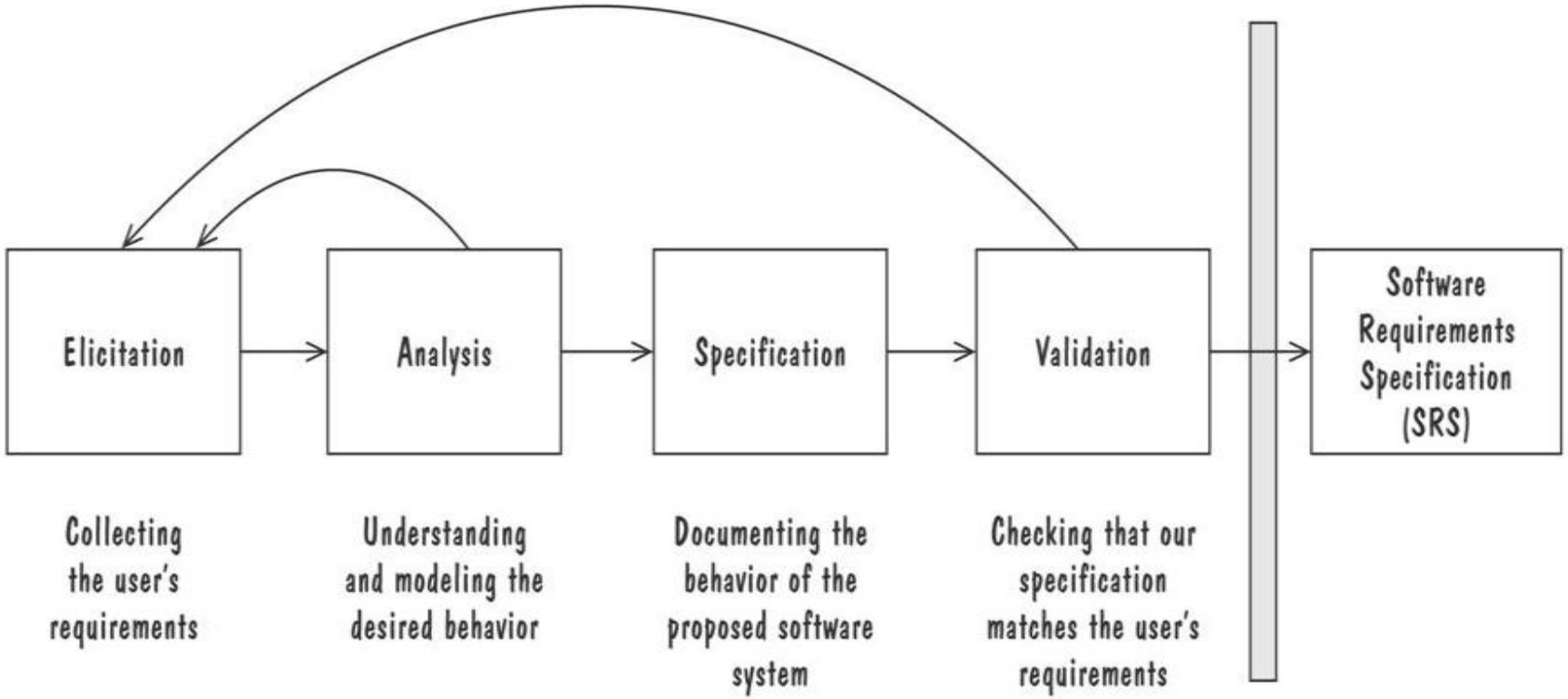
FAST-NUCES PESHAWAR

CONTENTS OF WEEK # 4

- Requirement Engineering (Cont…)
- Analysis Modeling with UML
- Object Oriented Analysis
- Modeling with UML (Static and Dynamic Models)
- Assignment # 1

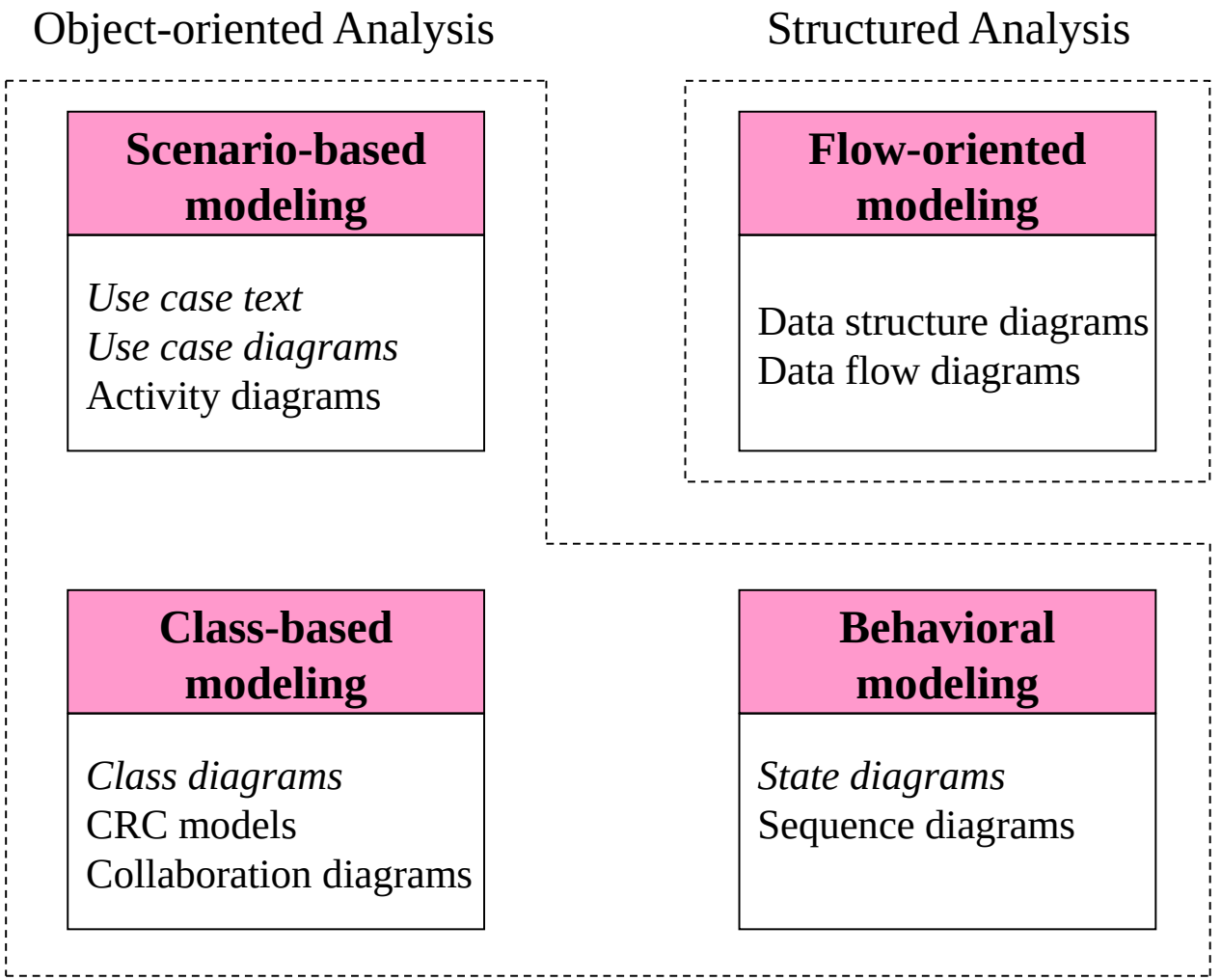
THE REQUIREMENTS PROCESS (PROCESS FOR CAPTURING REQUIREMENTS)

- Performed by the req. analyst or system analyst
- The final outcome is a Software Requirements Specification (SRS) document



ANALYSIS MODELING

ELEMENTS OF THE ANALYSIS MODEL



UNIFIED MODELING LANGUAGE (UML)

WHAT IS UML

- UML is the “Unified” Modelling Language”.
- Standard modelling language for software development.
- Maintained by Object Management Group.

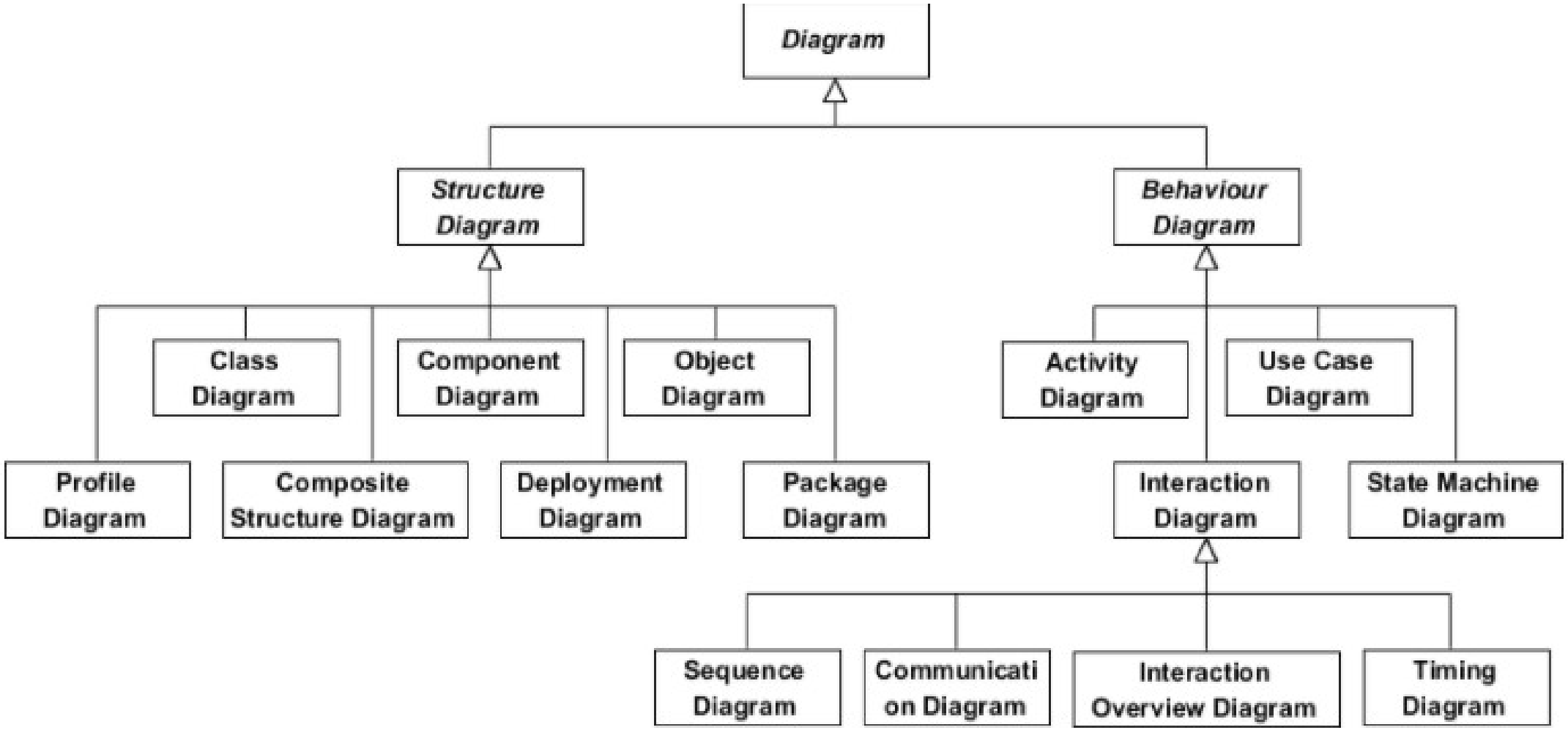
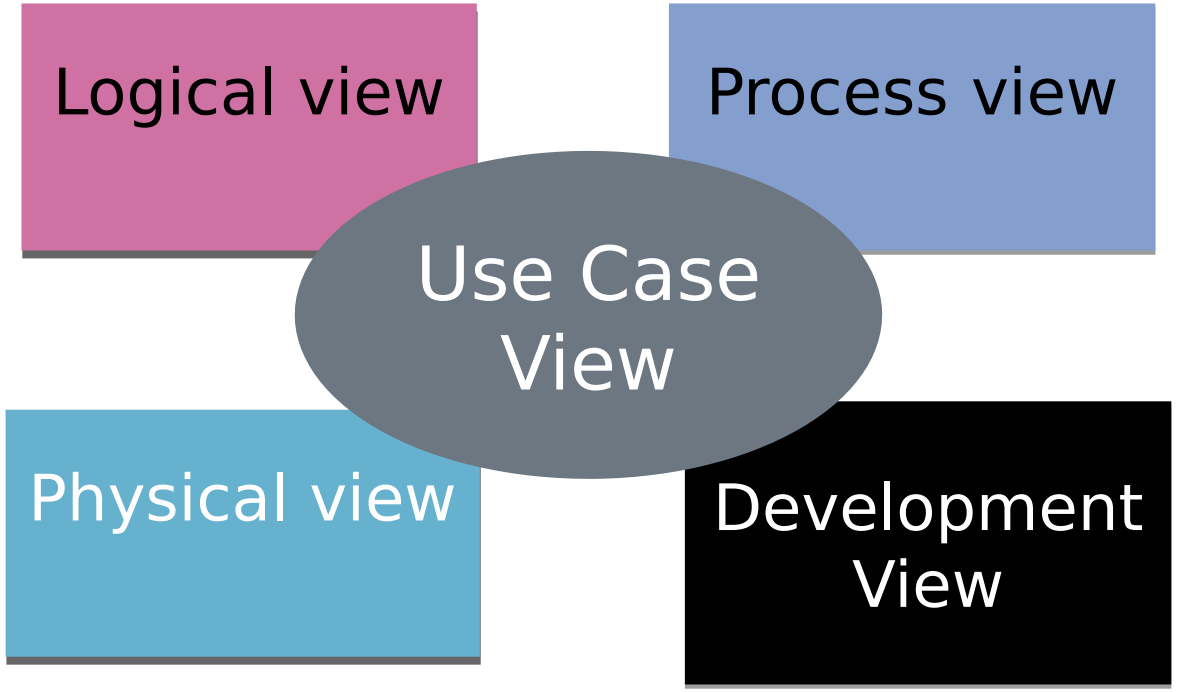


Diagram
Structure Diagram
Behaviour Diagram
Class Diagram
Component Diagram
Object Diagram
Activity Diagram
Use Case Diagram
Profile Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Interaction Diagram
State Machine Diagram
Sequence Diagram
Communicati on Diagram
Interaction Overview Diagram
Timing Diagram

VIEW OF THE MODEL

- UML model diagrams can be broken into different perspective or view.
- Can be explained through kruchen’s 4+1 Model.



LOGICAL VIEW

- The logical view shows the parts that make up the system and how they interact with each other.
- It represents the abstractions that are used in the problem domain
  - These abstractions are classes and objects
- Different UML diagrams show the logical way such as class diagram state diagram sequence, diagram communication diagram and object diagram.

Logical view

- class diagram
- state diagram
- Sequence diagram
- communication diagram
- object diagram

PROCESS VIEW

- Then we have the process view
- Through this view, we can describe the processes of the system and how they communicate with each other using process
- Using process view, we can find out what needs to happen to the system
- So using process view we can understand the overall functioning of the system
- Activity diagram in UML represents the process view

Process view

- Activity Diagram

PHYSICAL VIEW

- Next is physical view
- The physical view is the view that models the execution environment of the system
- Using this view, we can model the software entities onto the hardware that will host and run the entities
- The physical view in UML is represented through deployment diagrams

Physical view

- Deployment Diagrams

DEVELOPMENT VIEW

- The development view describes the modules are the components of the system.
- This might include packages or libraries.
- It gives a high-level view of the architecture of the system and helps in managing the layers of the system.
- UML provides two diagrams for development view.
  - component Diagram
  - package Diagrams

- All these four views are dependent on Use Case view

Development View
- Component Diagram
- Packages Diagram

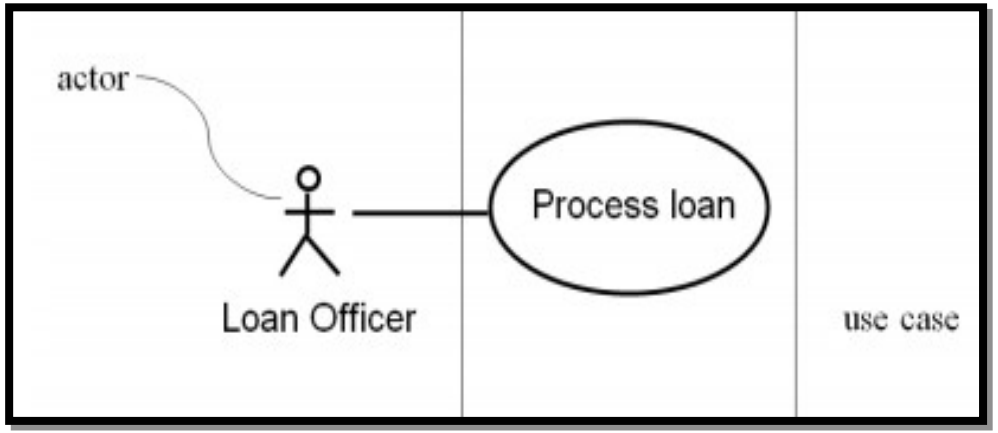
USE CASE

- They use case view illustrates the functionality of the system.
- Using use case we can capture the goals of the user or what the user expects from the system.
- In UML, Use Cases can be created through use case diagrams or use case descriptions (we will discuss it later).
- Use cases can be created by analysts' architects or even by the users.

SCENARIO-BASED MODELING

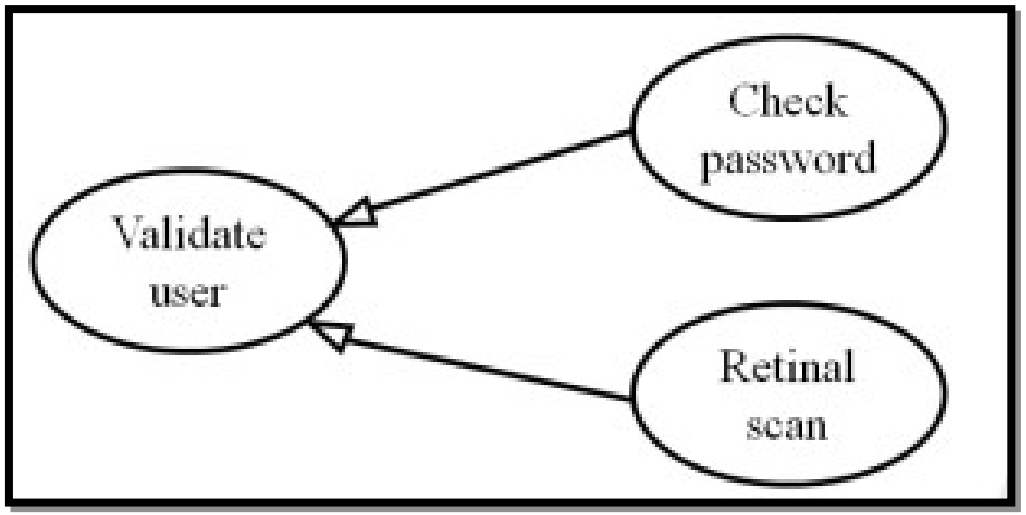
DEVELOPING USE CASES

- – Define the set of actors that will be involved in the story
  - Actors are people, devices, or other systems that use the system or product within the context of the function and behavior that is to be described
  - Actors are anything that communicate with the system or product and that are external to the system itself



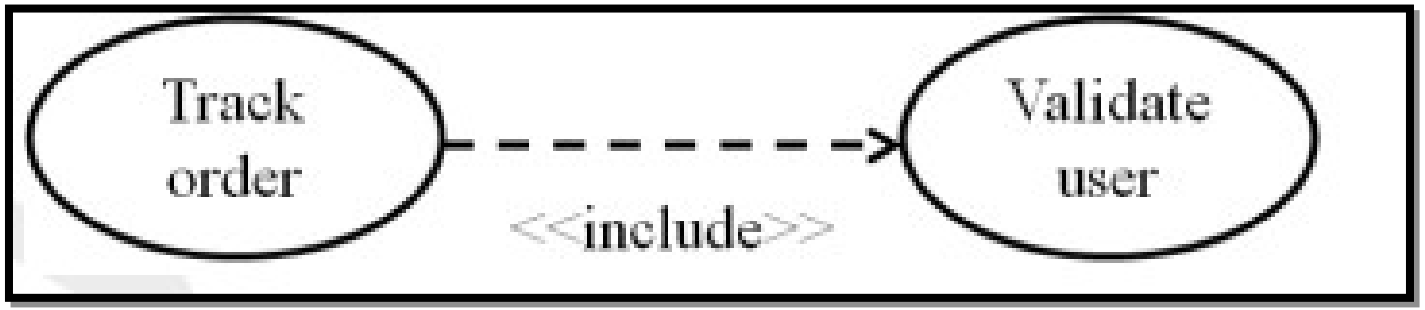
SPECIALIZED USE CASES

You may have two specialized children of this use case (Check password and Retinal scan).



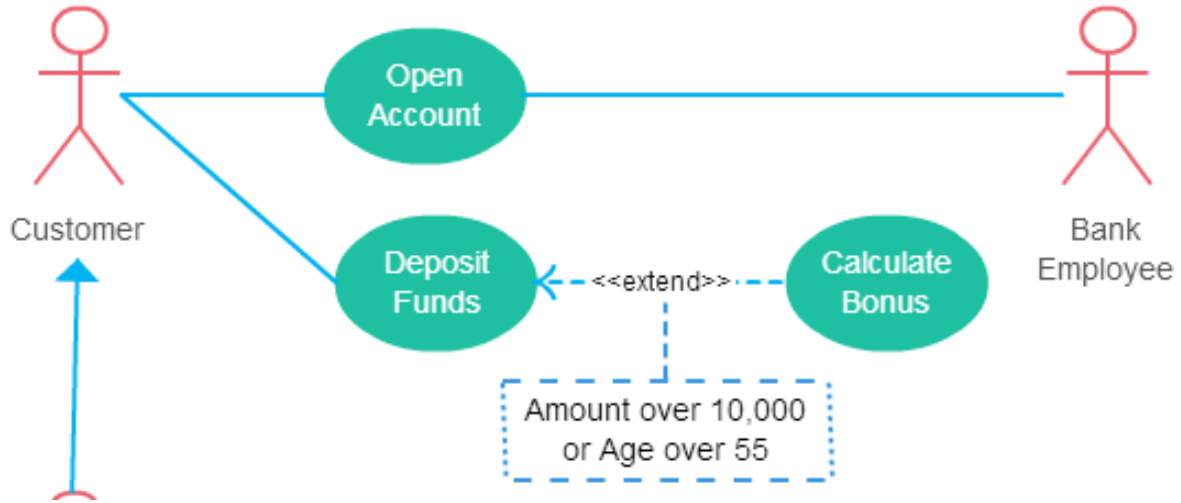
INCLUDE

- Include relationship is used to avoid describing the same flow of events several times, by putting the common behavior in a use case of its own
- This is an example of dependency

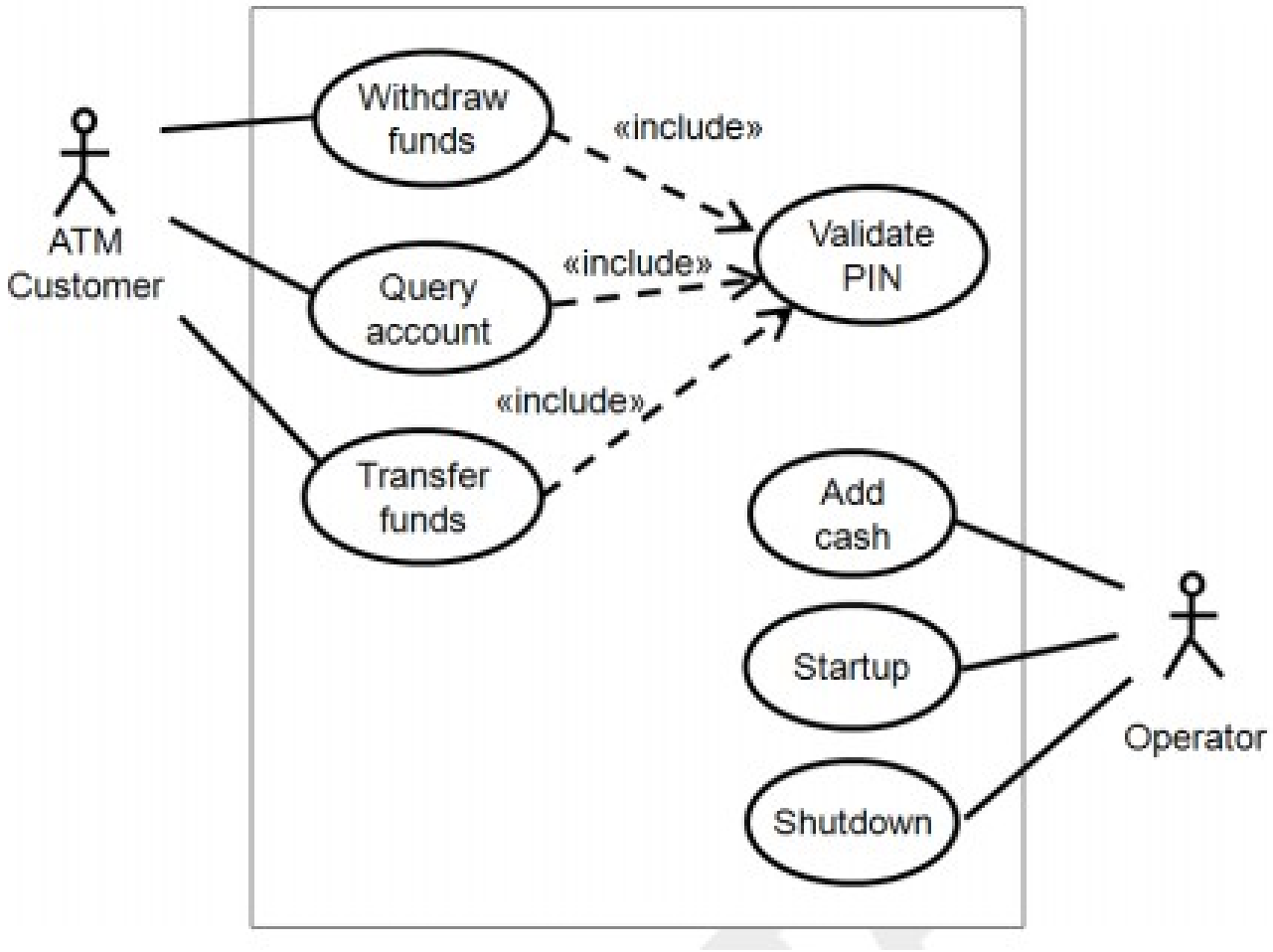


EXTEND

- **The extending use case is usually optional** and can be triggered conditionally. In the diagram, you can see that the extending use case is triggered only for deposits over 10,000 or when the age is over 55.



USE CASE DIAGRAM FOR ATM



Use Case Example

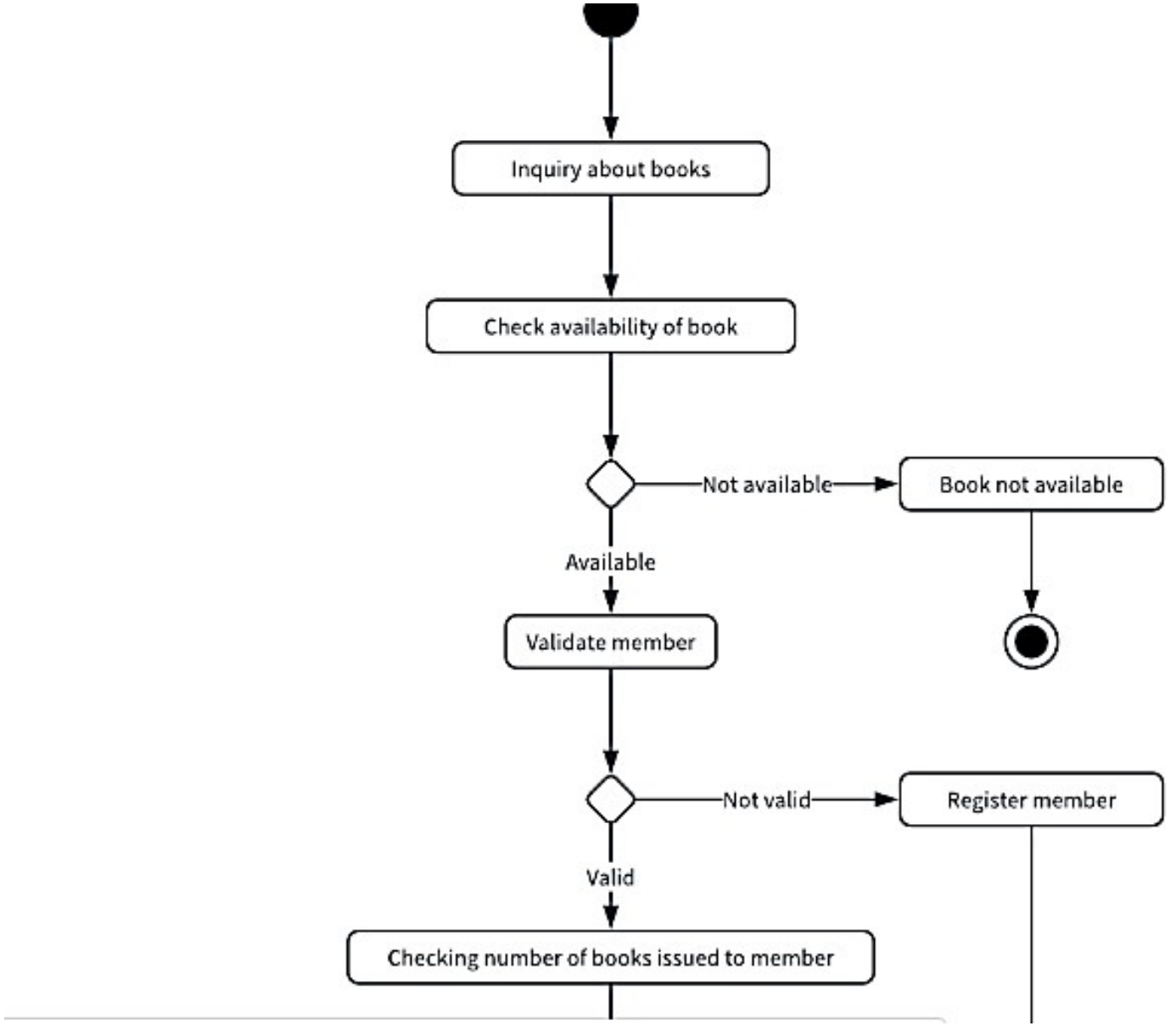
Name	Validate PIN
Summary	System validates customer PIN
Dependency	none
Actors	ATM, Customer
Preconditions	ATM is idle, displaying a Welcome message.
Flow of Events	Activity Diagram
Alternatives	• If the system does not recognize the card, the card is ejected. • If the system determines that the card date has expired, the card is confiscated. • If the system determines that the card has been reported lost or stolen, the card is confiscated. • If the customer-entered PIN does not match the PIN number for this card, the system re-prompts for PIN. • If the customer enter the incorrect PIN three times, the system confiscates the card. • If the customer enters Cancel, the system cancels the transaction and ejects the card
Post condition	Customer PIN has been validated.

BEHAVIORAL MODELING

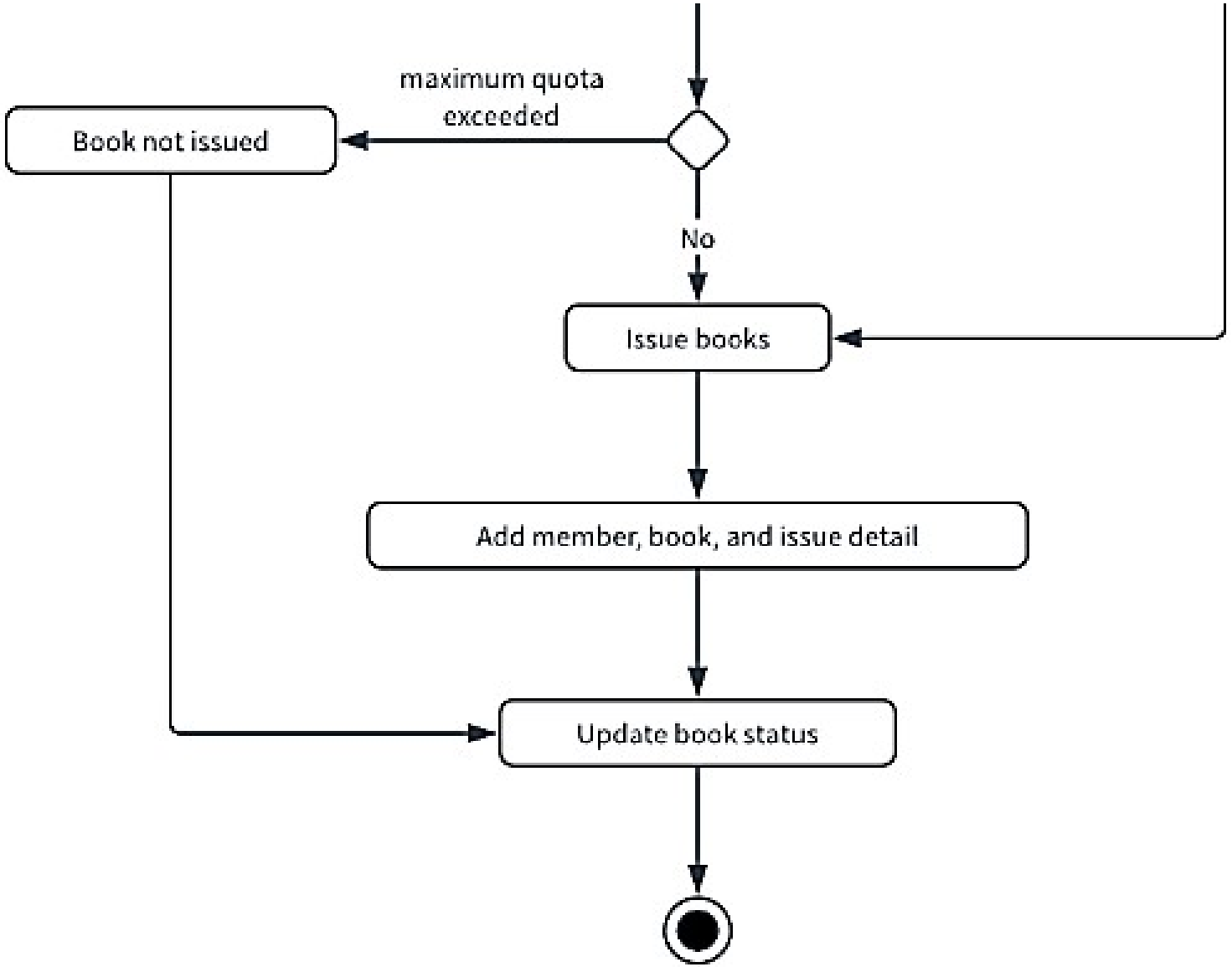
ACTIVITY DIAGRAM

- An activity diagram in the use-case model can be used to capture the activities and actions performed in a use case.
- It expresses the dynamic aspect of the system.

Activity diagram



Activity diagram

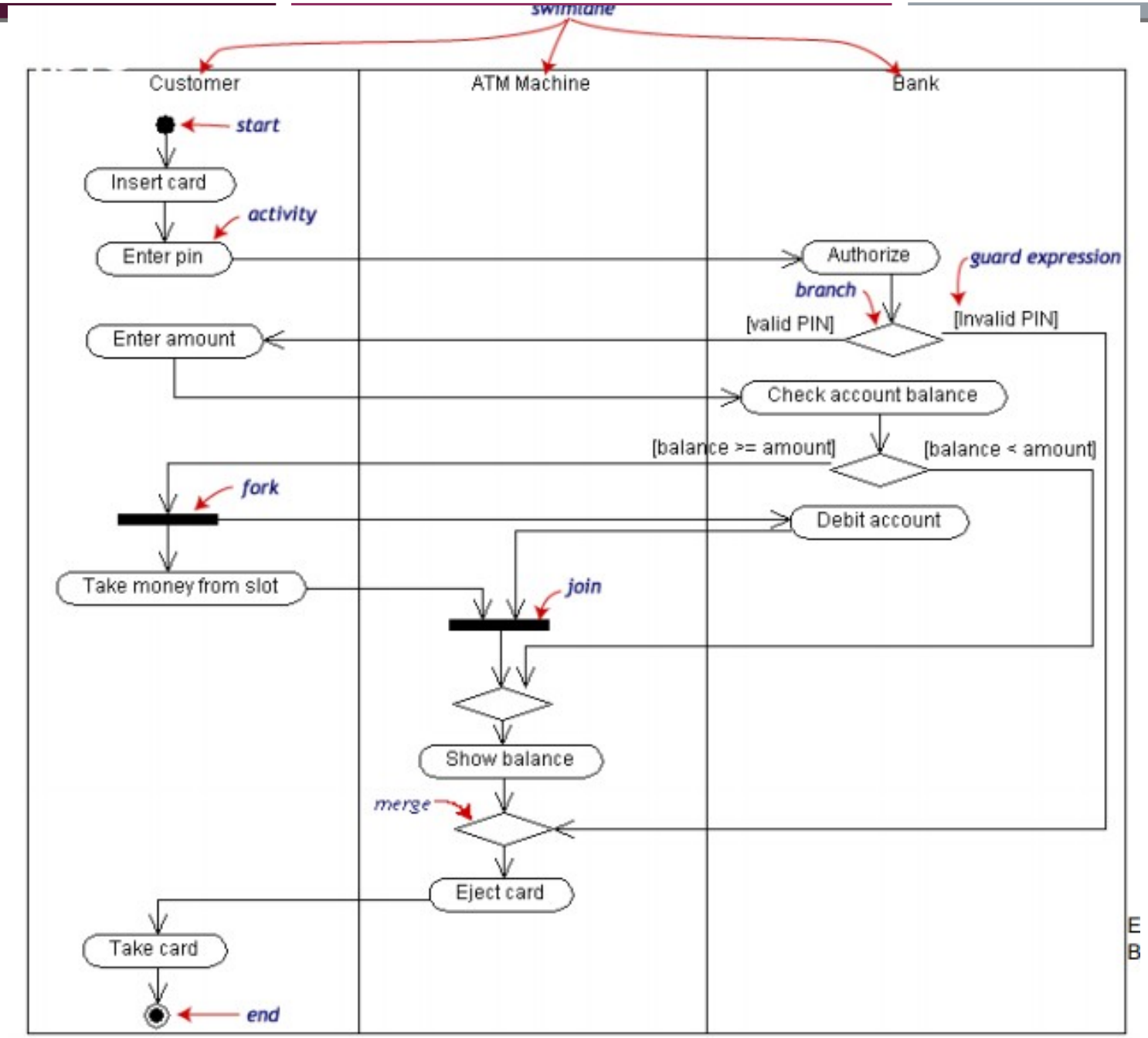


SWIMLANE DIAGRAM

Mapping Who Does What to Whom

- You are assigning a responsibility to an actor.
- Note, we did not say to an object - to an actor.

Swimlane Diagram



HAVE A GOOD DAY!