

# Software Engineering

## Lecture 5

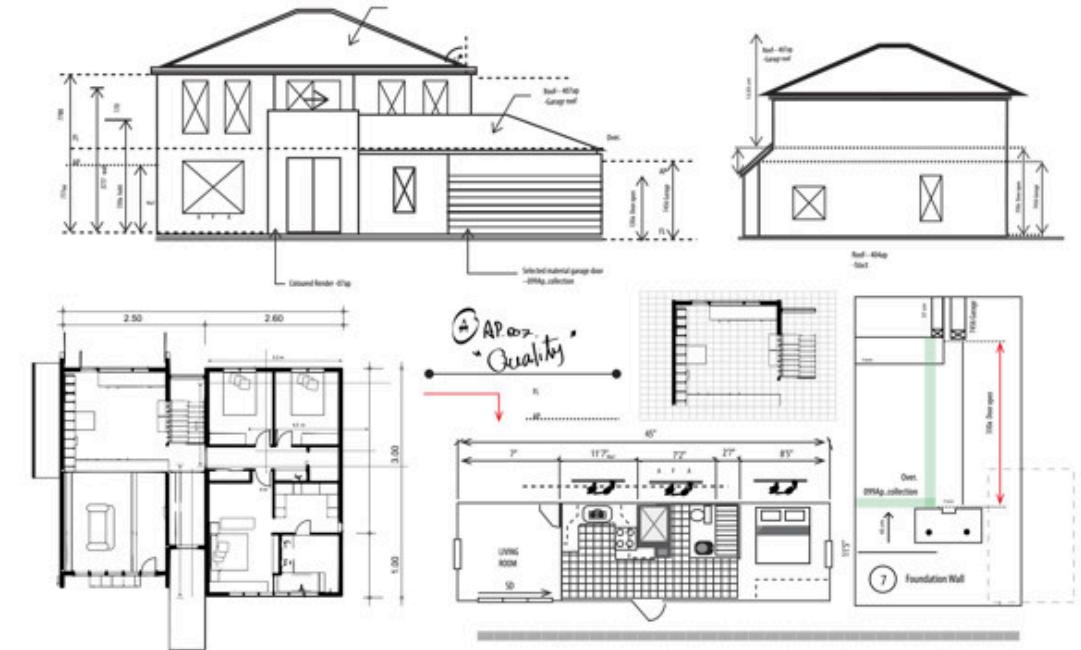
Lectures	Topics
1	Introduction to Software Engineering
2	Software Development Process (SDLC Activities) <ul style="list-style-type: none"> <li>- SDLC Activity: Specification or Requirement Engineering</li> <li>- SDLC Activity: System Modeling/Design</li> <li>- SDLC Activity: Implementation</li> <li>- SDLC Activity: Testing</li> <li>- SDLC Activity: Evolution</li> <li>- SDLC Activity: Deployment/Installation</li> <li>- SDLC Activity: Maintenance</li> </ul>
3	SDLC Activity: Requirement Engineering <ul style="list-style-type: none"> <li>- Requirement Elicitation</li> <li>- Requirement Analysis and Management</li> <li>- Requirement Validation</li> </ul>
4, 5, 6	SDLC Activity: System Modeling/Design <ul style="list-style-type: none"> <li>- Context Modeling</li> <li>- Data Modeling</li> <li>- Structural/Architectural Modeling</li> <li>- Process Modeling</li> <li>- UI/UX Modeling</li> </ul>
7,8,9	SDLC Activity: Implementation (Coding, tools, GIT – Version management, IDE, RESTFUL architecture)
10	SDLC Activity: Testing
11	SDLC Activity: Deployment (tools to deploy, cloud computing)
12	SDLC Activity: Maintenance

# Agenda

- Structural / Architectural Modeling
  - Class Diagram
    - Aggregation
    - Composition
    - Inheritance/Generalization
    - Association
  - Process Modeling
    - Generic Process Modeling
      - Business Process Modeling (BPM)
      - Flowchart
    - Interaction Model
      - Sequence Diagram
      - State Transition Diagram

## 4.3 Structural / Architectural Modeling

- Structural models of software displays 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
- You create structural models of a system when you are discussing and designing the system architecture



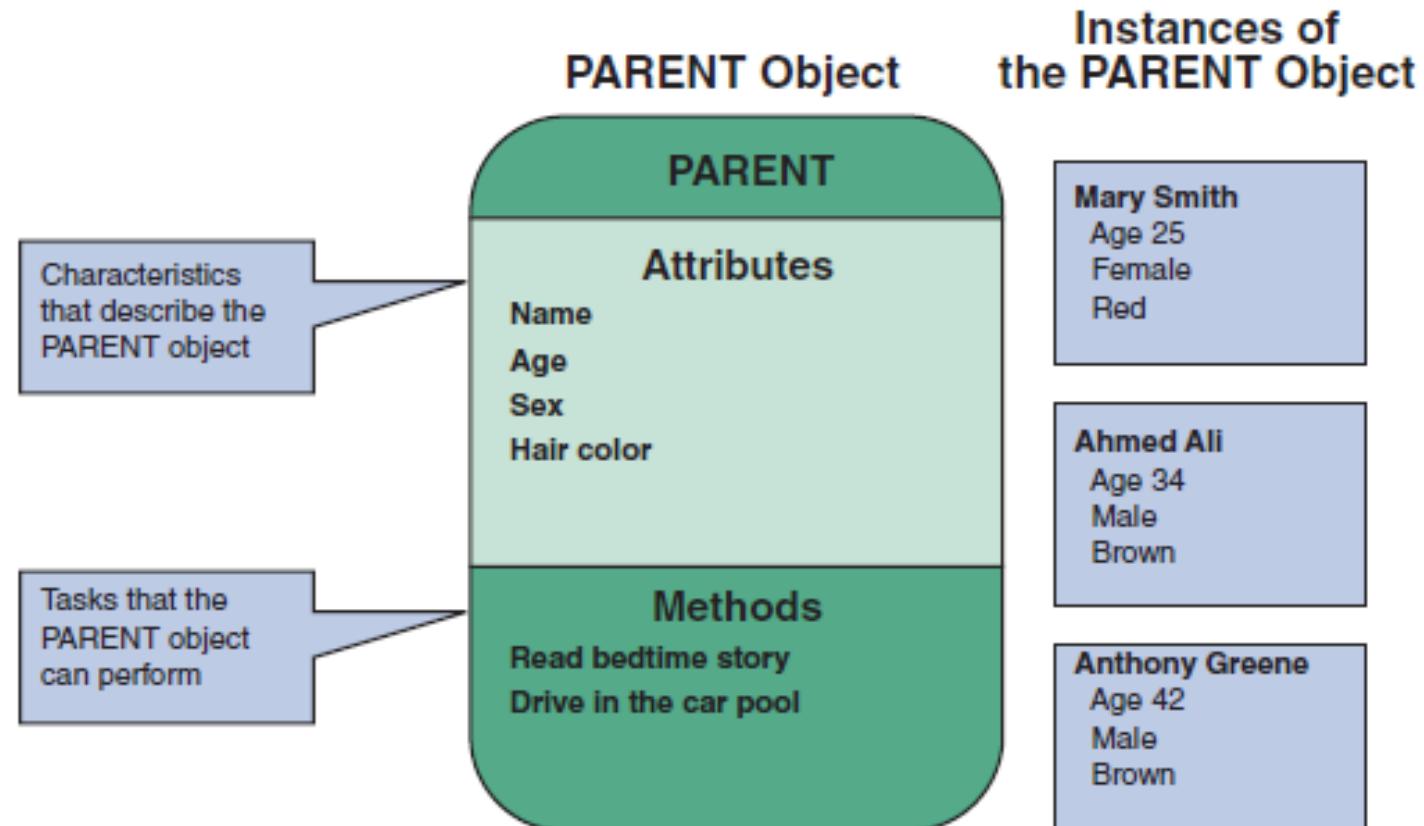
## 4.3 Structural / Architectural Modeling

Paradigm	Architectural Modeling Technique
Structured Paradigm	<ul style="list-style-type: none"><li>- Structure Chart</li></ul>
Object Oriented Paradigm	<ul style="list-style-type: none"><li>- Class Diagram (Our Focus)</li></ul>

## 4.3.1 Class

- Class is a concept.
- A class is a description of a collection of objects with common attributes and behaviors.
- In practice, the definition or specification of a class includes the definition of attributes comprising the state, the method implementing the behavior and how to handle creation and destruction of an object.
- A class is a group of similar objects. For example, Ford Fiestas belong to a class called CAR.

## 4.3.1 Class Diagram



## 4.3.1 Object

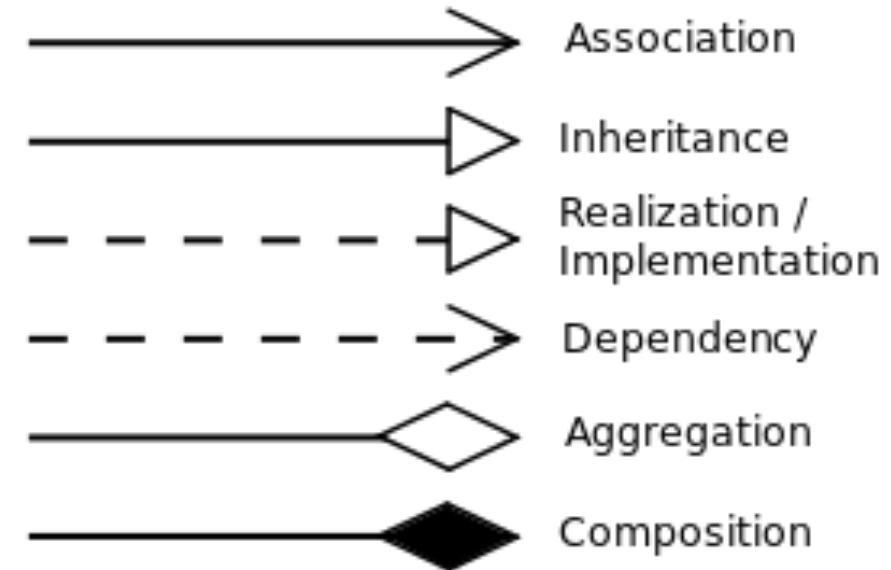
- An object is **actual** thing or a result of a concept. It can be real world thing or concept, or an abstraction of thing or concept expressed as software representation.
- An **object has state and behavior.**
- Individual objects are called **instances**, have identity and are distinct, distinguishable from other objects.
- An instance is a specific member of a class. Your Ford Fiesta, therefore, is an instance of the CAR class.

## 4.3.1 Class Diagram

Modeling Technique: Class Diagrams (OOD Paradigm)

Required concept to know before drawing Class Diagram (Class Diagram relationships)

1. Aggregation
2. Composition
3. Inheritance / Generalization
4. Association

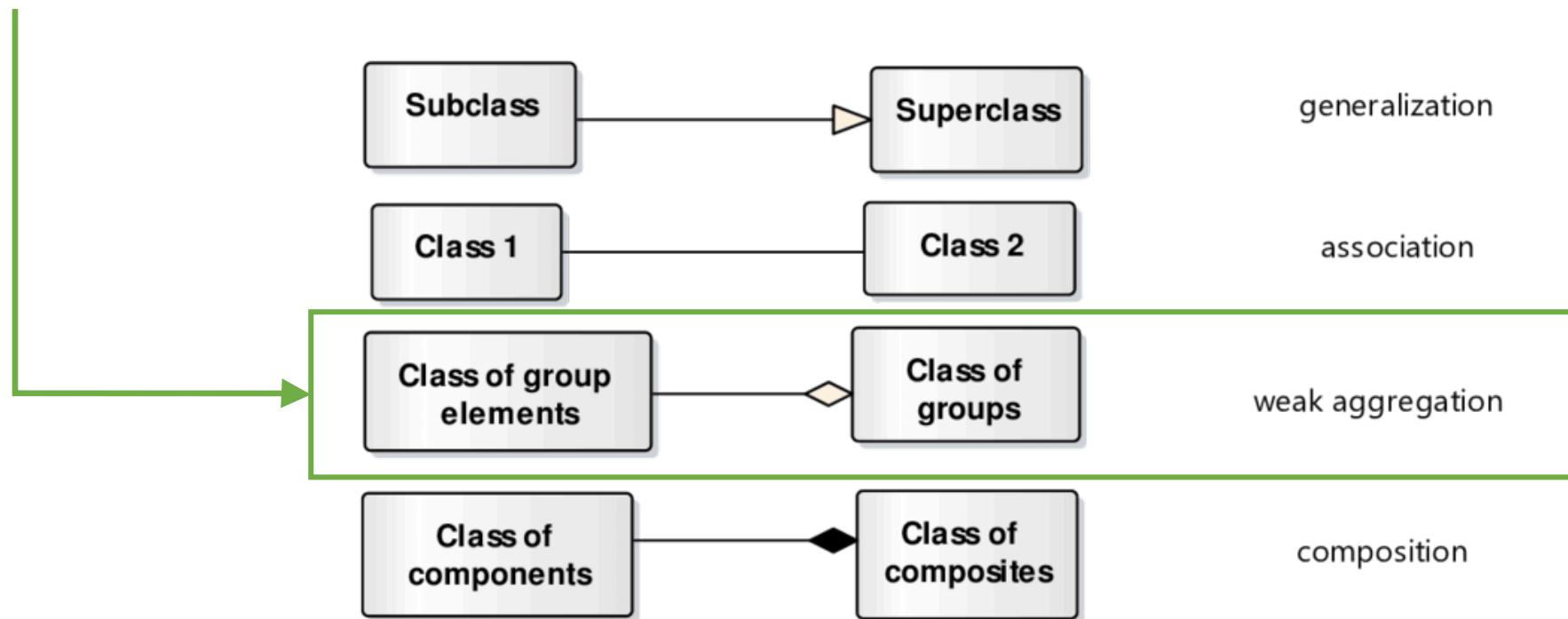


## 4.3.1 Class Diagram Basics

- Unified Modeling Language (UML) is a widely used method of visualizing and documenting an information system.
- First, let us understand the basic concepts involved in objects oriented
  - 1. Classes
  - 2. Objects / Instances
  - 3. Attributes
  - 4. Methods

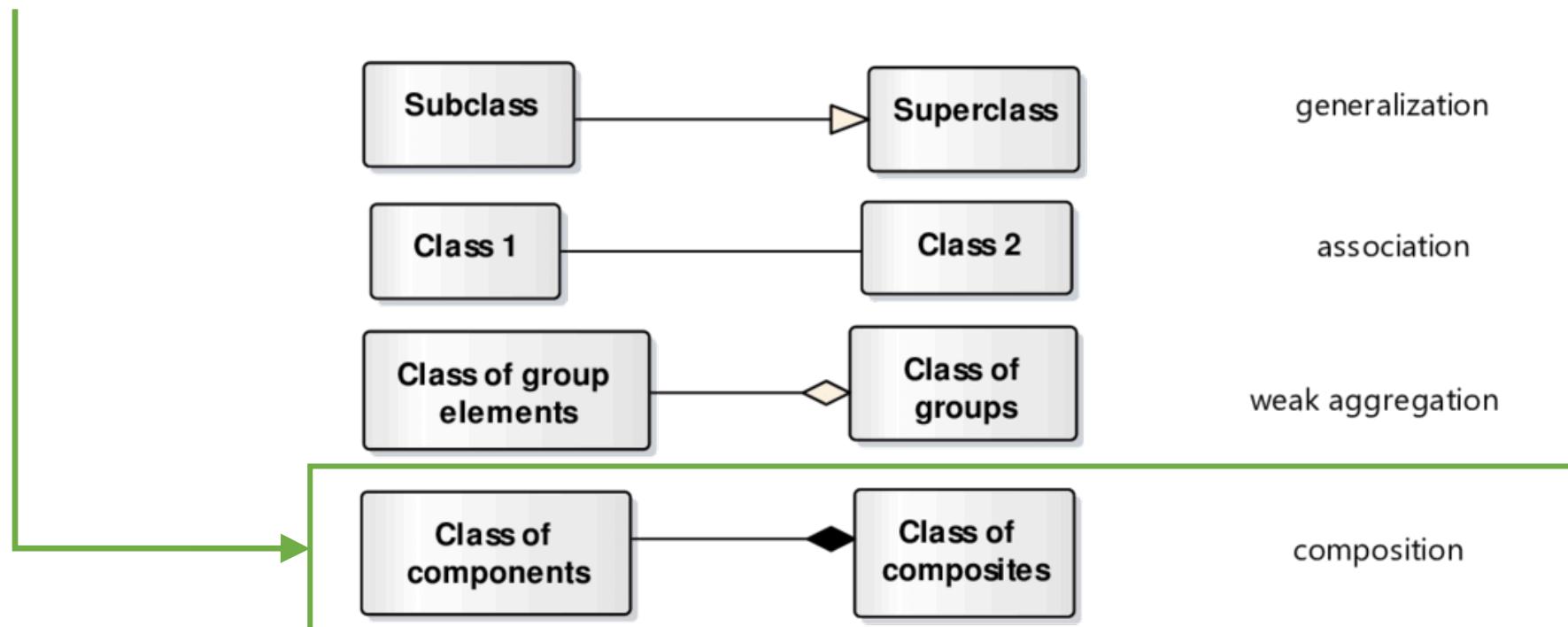
## 4.3.1.1 Class Diagram: Aggregation

- An aggregation model shows how classes that are collections are composed of other classes
- loose relationship



## 4.3.1.2 Class Diagram: Composition

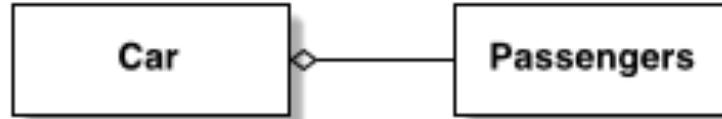
- An aggregation model shows how classes that are collections are composed of other classes
- Strong relationship



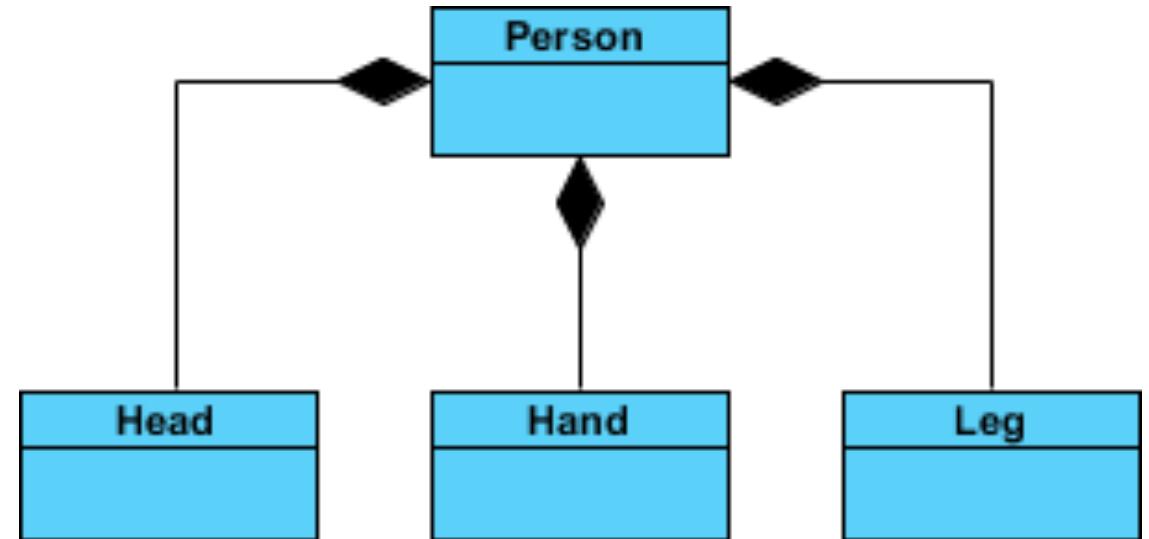
# Difference between Aggregation and Composition relationship



Composition: every car has an engine.

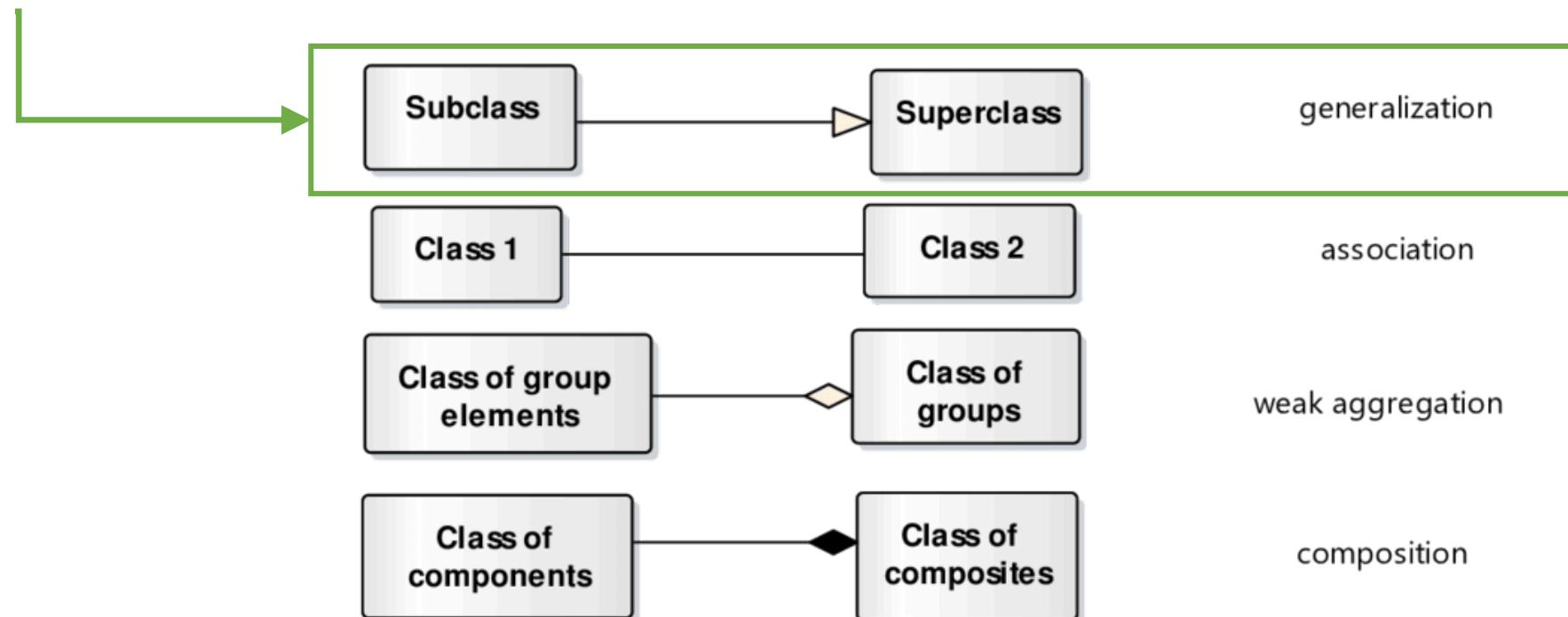


Aggregation: cars may have passengers, they come and go

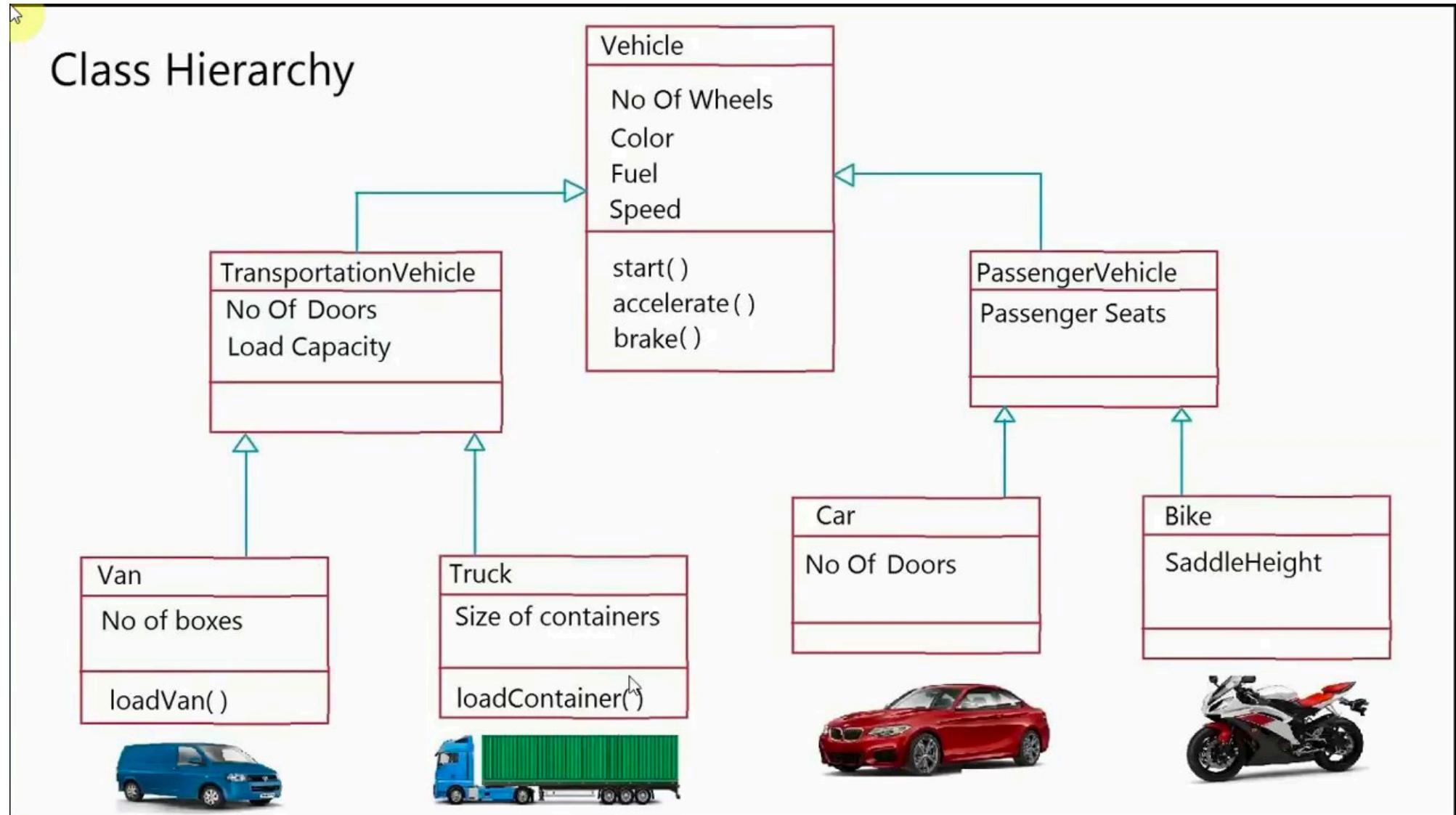


## 4.3.1.3 Class Diagram: Inheritance/Generalization

- Rather than learn the detailed characteristics of every entity that we experience, we place these entities in more general classes (animals, cars, houses, etc.) and learn the characteristics of these classes.

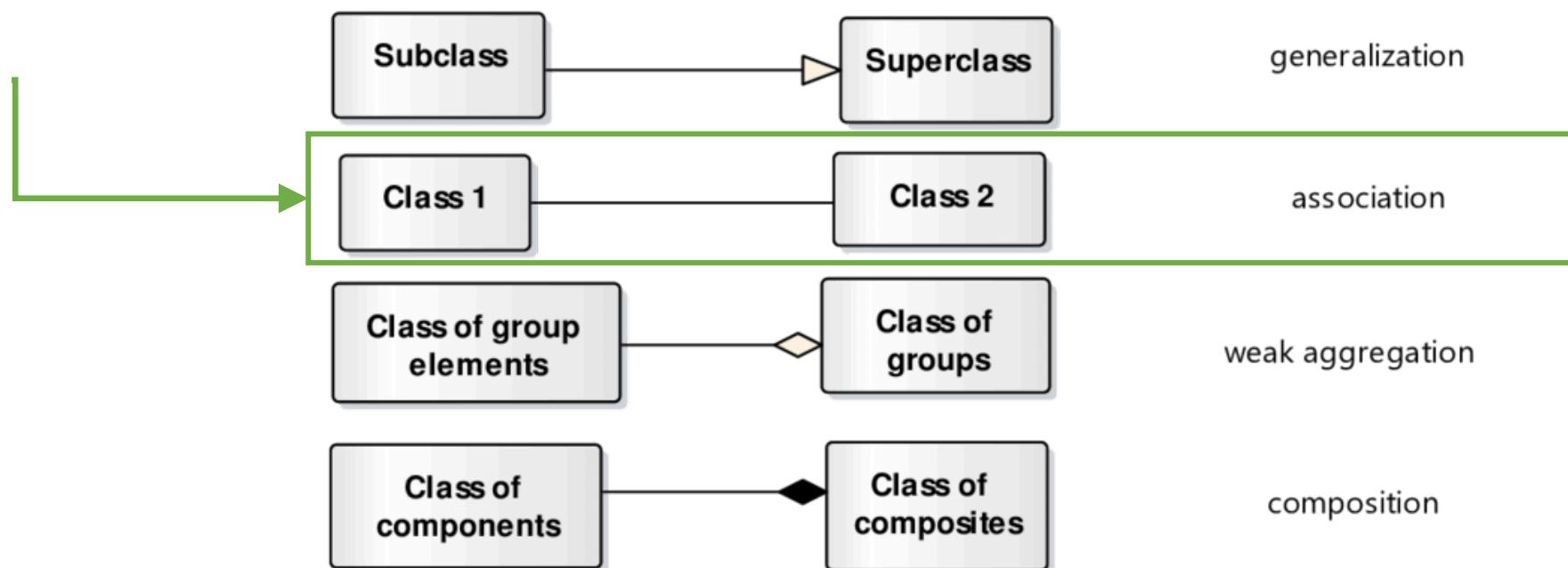


## 4.3.1.3 Inheritance Example

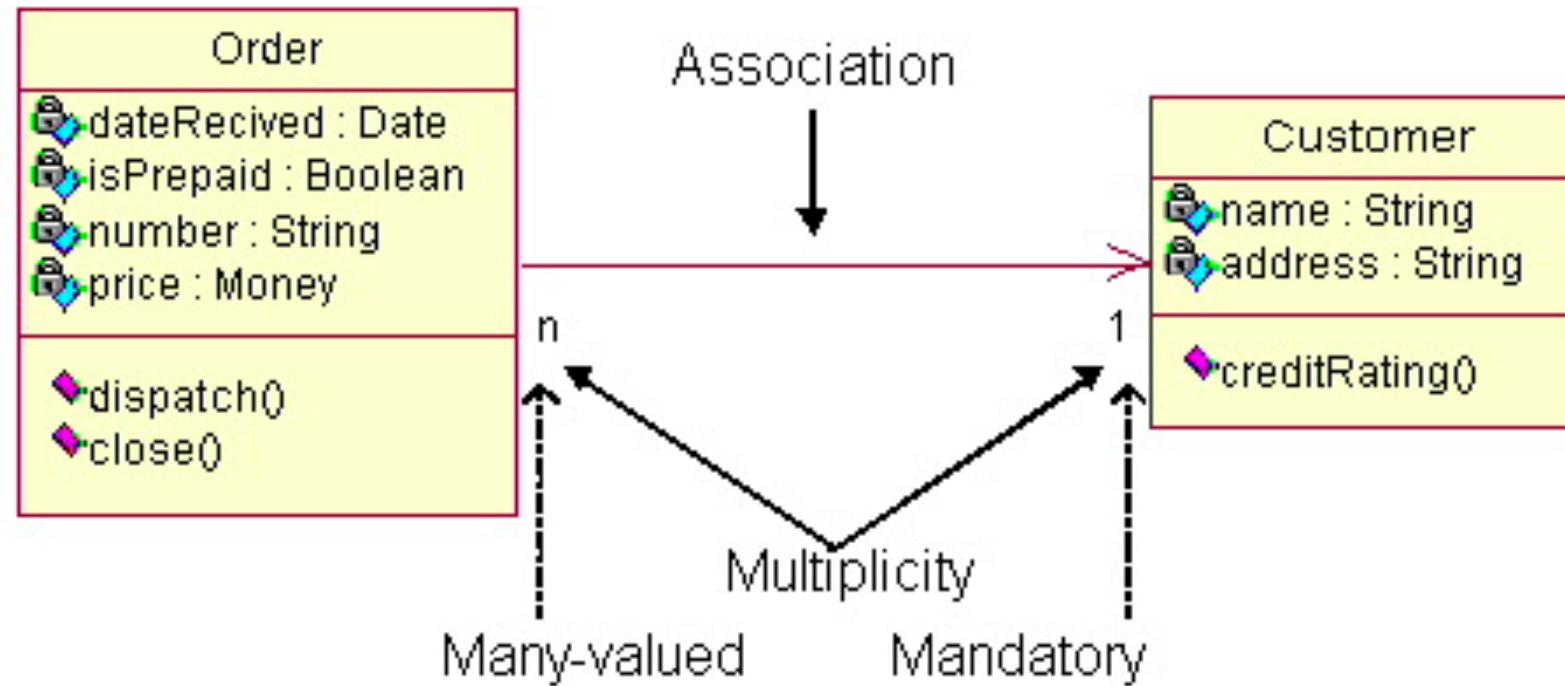


## 4.3.1.4 Class Diagram: Association

- Simple Relationship



## 4.3.1.4 Association Relationship Example



## 4.4 Process Modeling

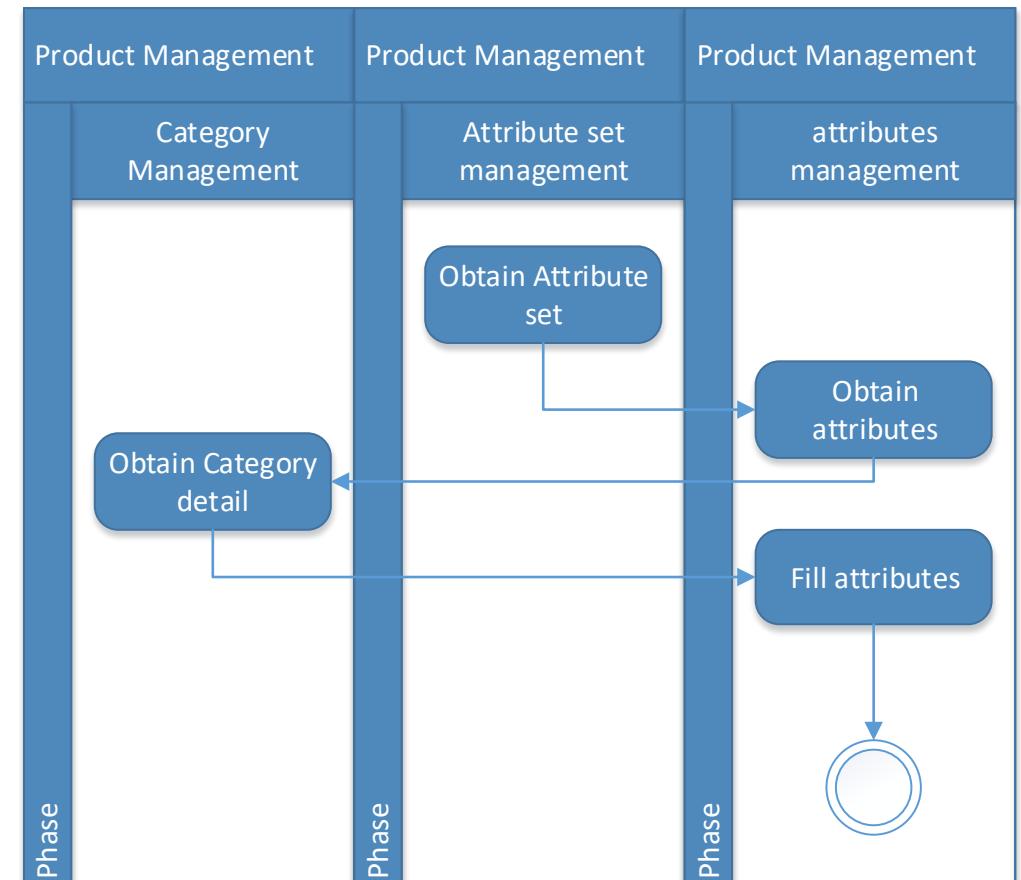
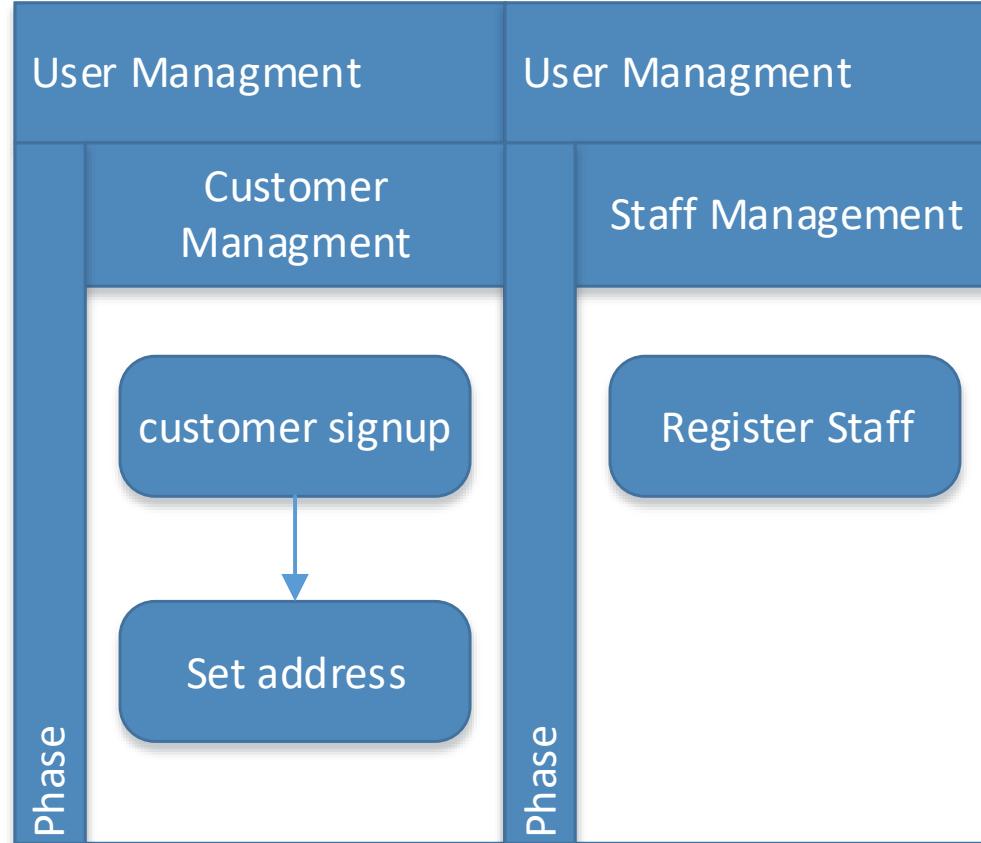
1. Generic Business Process Model
2. Interaction Model

## 4.4.1 Generic Process Modeling

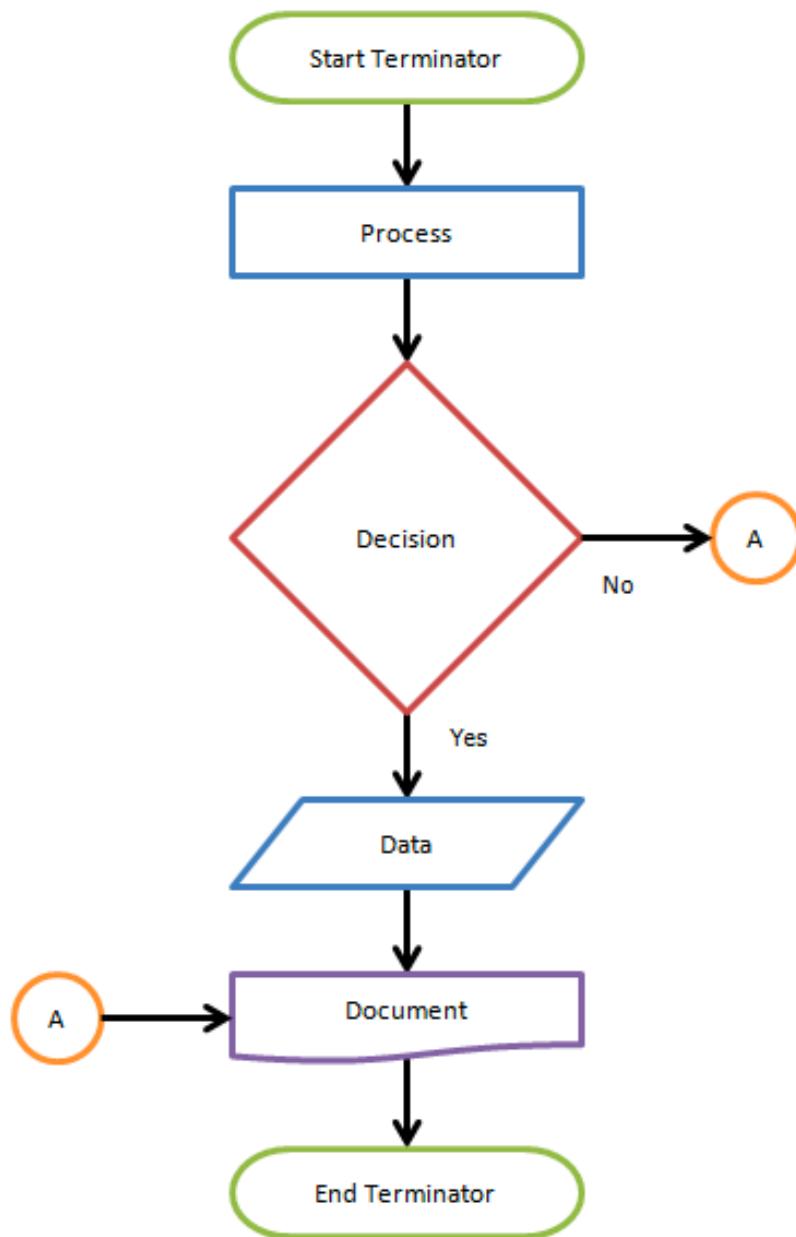
- Techniques
  - Business Process Modeling (BPM)
  - Flowchart
- Flow of business process to better understand the domain

## 4.4.1.1 Business Process Modeling (BPM)

- Use business process modeling notation (BPMN) to clarify business process
- Includes various shapes and symbols to represent events, processes and workflows.



## 4.4.1.2 Flowchart



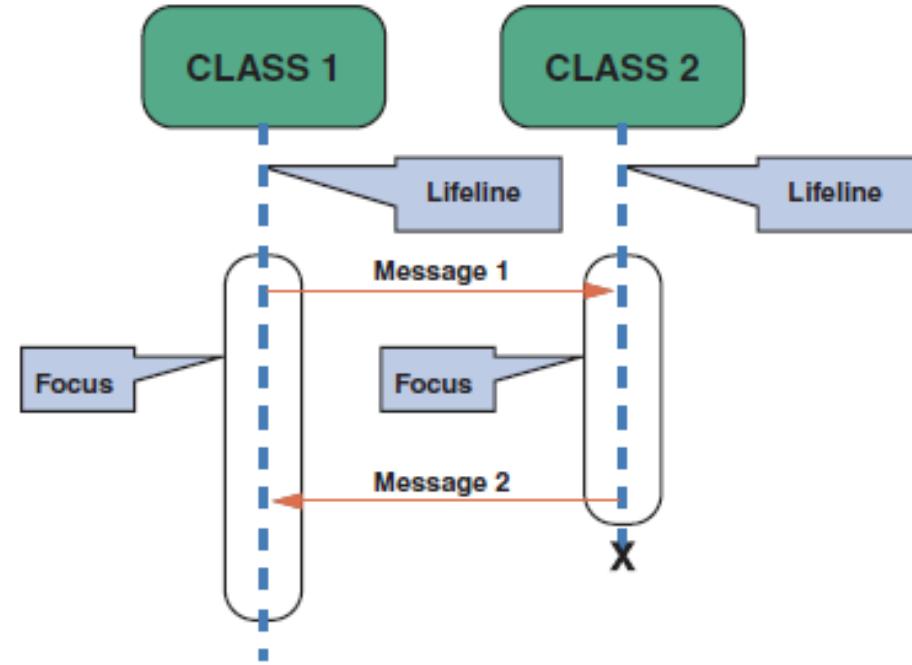
## 4.4.2 Interaction Models

- Behavioral models are models of the **dynamic behavior** of a system as it is executing. They show what happens or what is supposed to happen when a system responds to a stimulus from its environment.
- Two Sub model:
  - Data Driven Modeling
    - Sequence Diagram
  - Event Driven Modeling
    - State Transition Diagram (STD) (Figure – 5,42,43,44)

## 4.4.2.1 Sequence Diagram

- A sequence diagram is a dynamic model of a use case, showing the **interaction among classes during a specified time period**.
- A sequence diagram graphically documents the use case by showing the classes, the messages, and the timing of the messages.
- Sequence diagrams include symbols that represent classes, lifelines, messages, and focuses.

## 4.4.2.1 Sequence Diagram



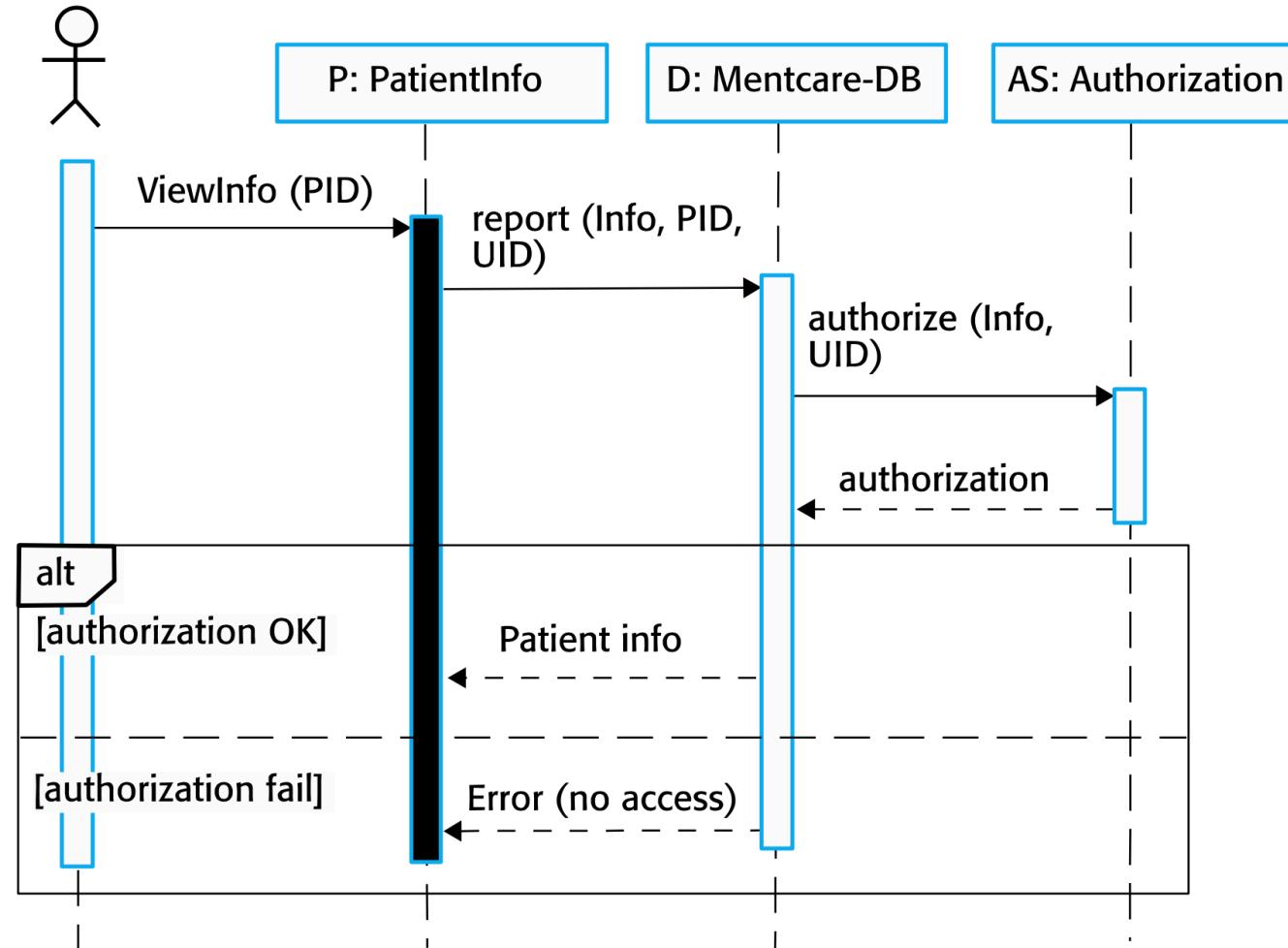
- Notice the X that indicates the end of the CLASS 2 lifeline.
- Also notice that each message is represented by a line with a label that describes the message, and that each class has a focus that shows the period when messages are sent or received.

## 4.4.2.1 Sequence Diagram

- **CLASSES** A class is identified by a rectangle with the name inside. Classes that send or receive messages are shown at the top of the sequence diagram.
- **LIFELINES** A lifeline is identified by a dashed line. The lifeline represents the time during which the object above it is able to interact with the other objects in the use case. An X marks the end of the lifeline.
- **MESSAGES** A message is identified by a line showing direction that runs between two objects. The label shows the name of the message and can include additional information about the contents.
- **FOCUSES** A focus is identified by a narrow vertical shape that covers the lifeline. The focus indicates when an object sends or receives a message.

## 4.4.2.1 Sequence Diagram

Medical Receptionist

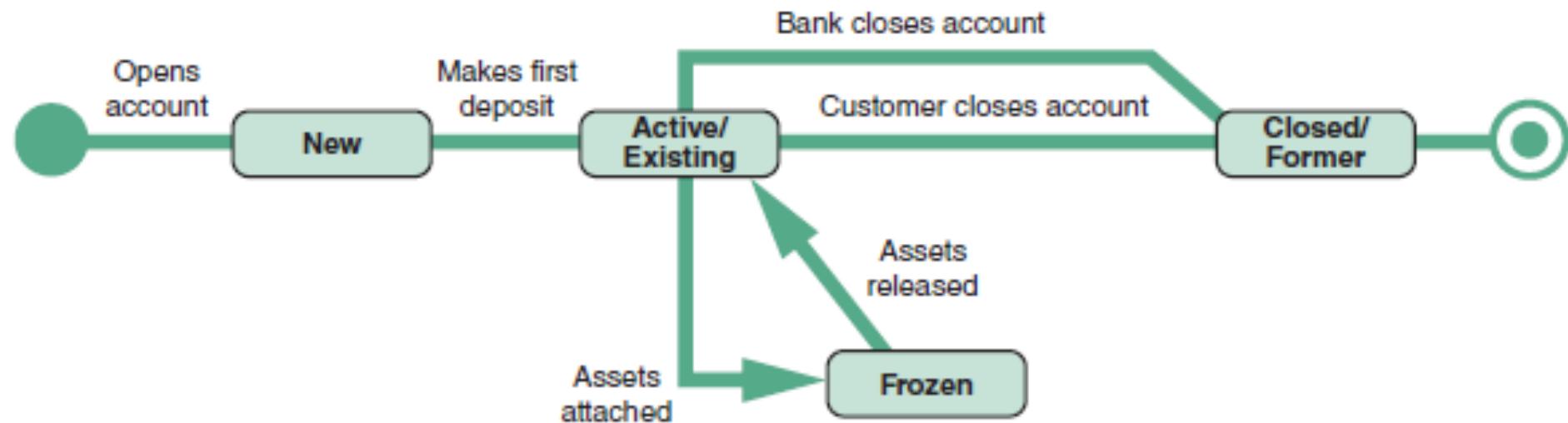


## 4.4.2.2 State Transition Diagram

- A state transition diagram shows how an **object changes from one state to another**, depending on events that affect the object.
- All possible states must be documented in the state transition diagram
- A bank account, for example, could be opened as a NEW account, change to an ACTIVE or EXISTING account, and eventually become a CLOSED or FORMER account. Another possible state for a bank account could be FROZEN, if the account's assets are legally attached.

## 4.4.2.2 State Transition Diagram

- In a state transition diagram, the states appear as rounded rectangles with the state names inside. The small circle to the left is the initial state, or the point where the object first interacts with the system.
- Reading from left to right, the lines show direction and describe the action or event that causes a transition from one state to another.
- The circle at the right with a hollow border is the final state.



# In summary,

Process Modeling	Modeling Technique (OOD)	Modeling Technique (Structural)
Generic Process Modeling	<ul style="list-style-type: none"> <li>- BPM</li> <li>- Flowchart</li> </ul>	<ul style="list-style-type: none"> <li>- BPM</li> <li>- Flowchart</li> </ul>
Interaction Process Modeling	<ul style="list-style-type: none"> <li>- Sequence Diagram</li> <li>- Activity Diagram</li> </ul>	<ul style="list-style-type: none"> <li>- DFD</li> </ul>

