



# Software Engineering I

Dr. Elham Mahmoudzadeh  
Isfahan University of Technology

[mahmoudzadeh@iut.ac.ir](mailto:mahmoudzadeh@iut.ac.ir)

2021

The background features a light gray gradient with several realistic water droplets of varying sizes scattered across the surface. In the center, there is a faint, circular logo. The logo consists of a gear-like outer ring with Persian text 'دانشگاه صنعتی اصفهان' (University of Technology of Isfahan) written along its top arc. Inside the gear is a stylized sunburst or star-like emblem.

# **Chapter 6**

## **Behavioral modeling(II)**

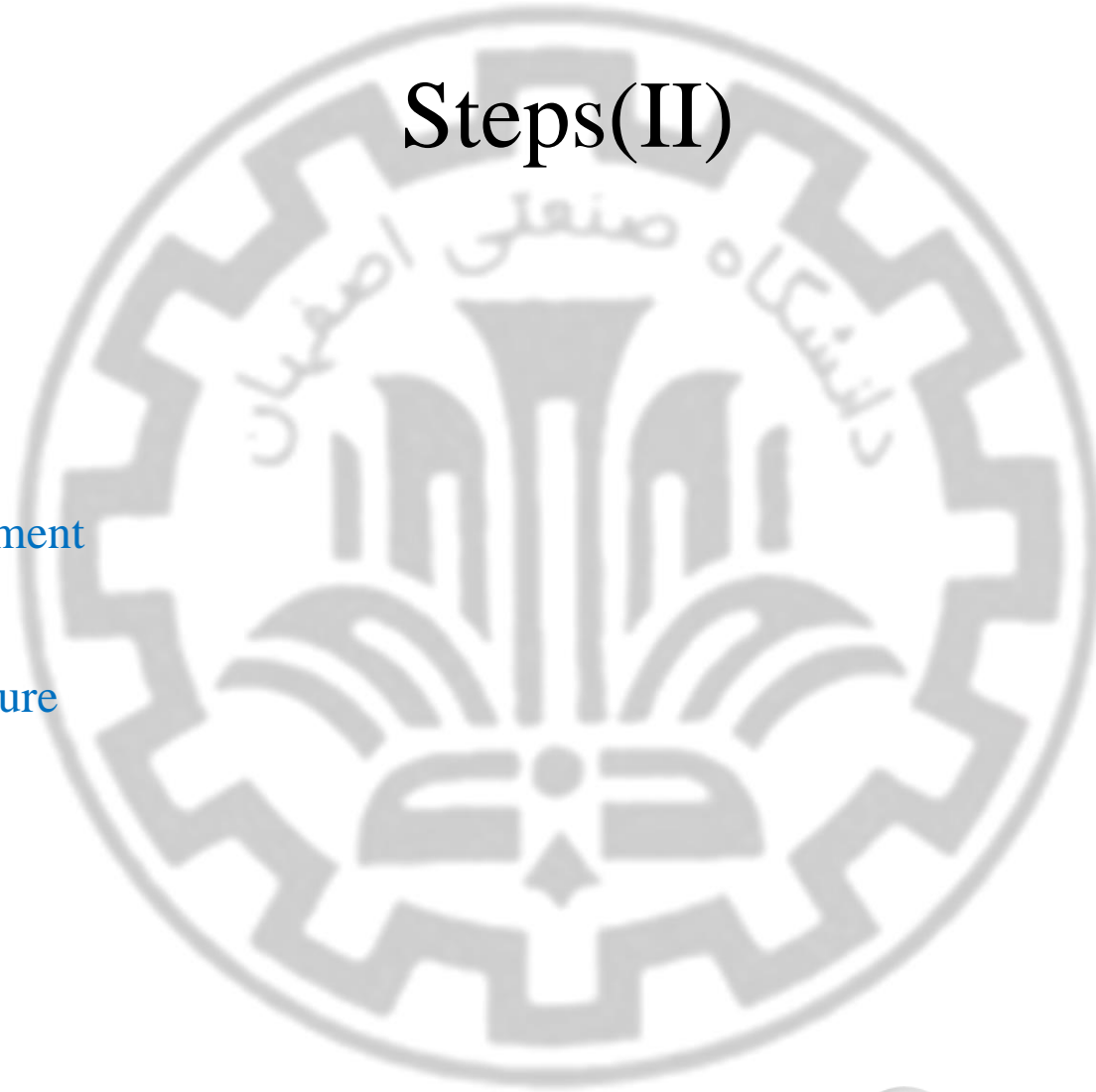
# Steps(I)

1. Preparing proposal
2. Requirements determination
  - User story
3. Abstract Business Process Modelling
4. Analysis
  - Functional Modelling
  - Structural Modelling
  - Behavioral Modelling

## Steps(II)

### 5. Design

- Optimization
- Database Management
- User Interface
- Physical Architecture



# Behavioral model

- Describe the internal dynamic aspects of an information system that supports the business processes in an organization.
- During analysis, behavioral models describe what the internal logic of the processes is without specifying how the processes are to be implemented.
- Later, in the design and implementation phases, the detailed design of the operations contained in the object is fully specified.

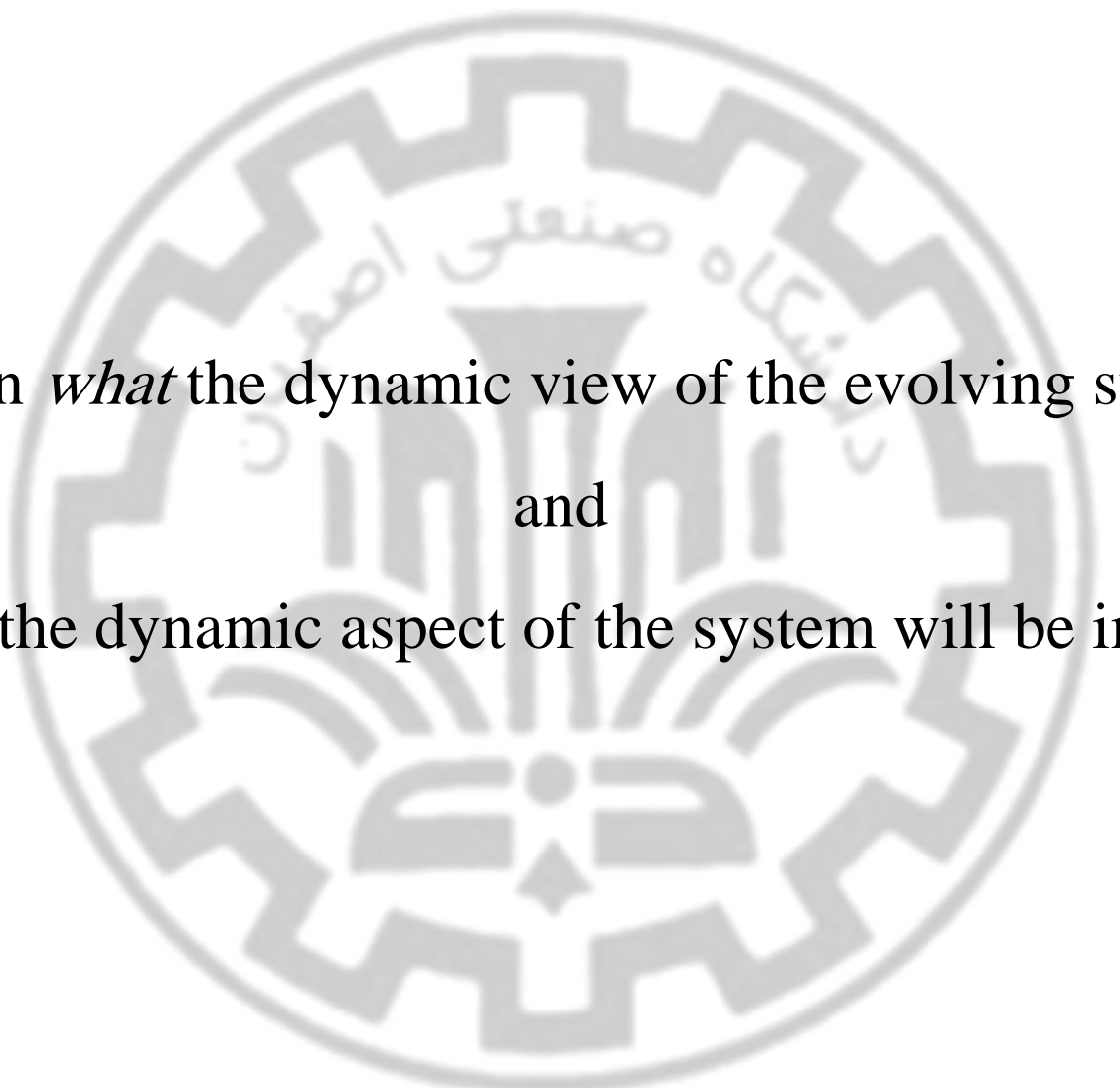
# Inputs

- Use **business process** and **functional models** to describe the **functional or external behavioral** view of an information system.
- Use **structural models** to depict the **internal structural or static view** of an information system.



# Types of behavioral models

- Behavioral models used to represent the underlying details of a business process portrayed by a use-case model, for example in UML, **interaction diagrams (sequence and communication)**.
  - Interaction diagrams allow the analyst to model the distribution of the behavior of the system over the actors and objects in the system.
- Behavioral model is used to represent the **changes that occur in the underlying data**, for example in UML, behavioral state machines.



Focus on *what* the dynamic view of the evolving system is  
and  
not on *how* the dynamic aspect of the system will be implemented



# Communication Diagrams

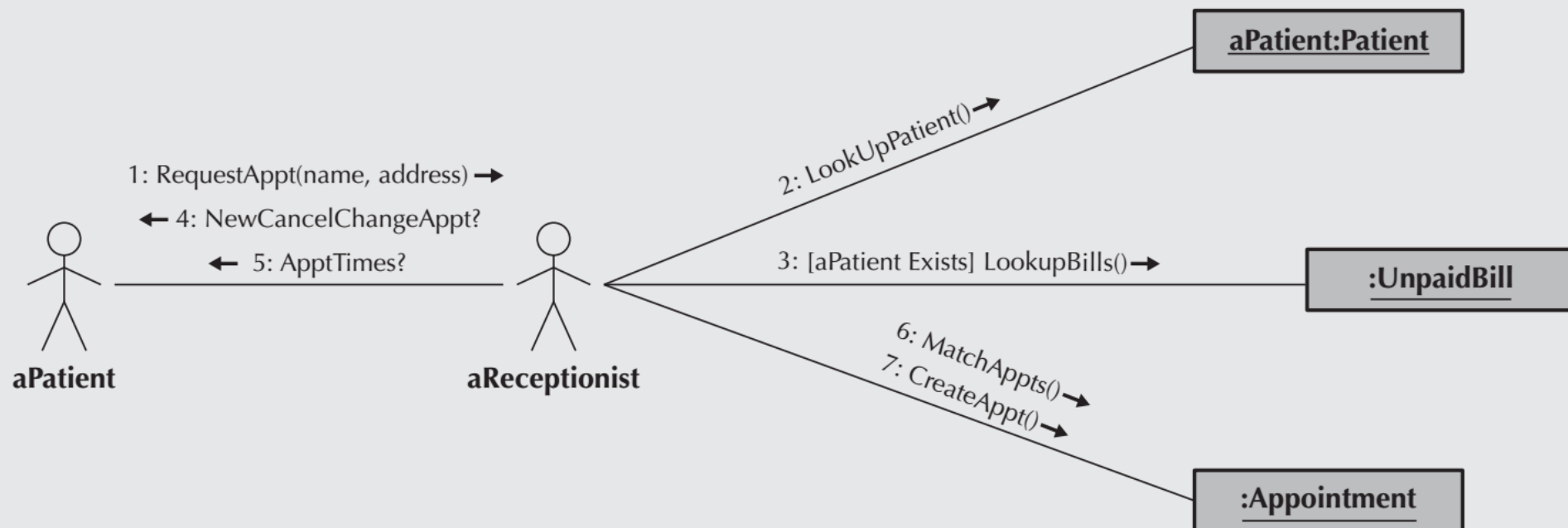
- Communication diagrams, like sequence diagrams, essentially provide a view of the dynamic aspects of an object-oriented system.
- Show how the members of a set of objects collaborate to implement a use case or a use-case scenario.
- Used to model all the interactions among a set of collaborating objects.
- A communication diagram can portray how dependent the different objects are on one another.
- A communication diagram is essentially an object diagram that shows message-passing relationships instead of aggregation or generalization associations.

# Communication Dia. vs. Sequence Dia.

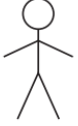
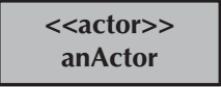


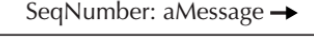

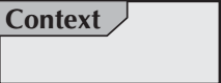
- Communication diagrams emphasize the flow of messages through a set of objects, whereas the sequence diagrams focus on the time ordering of the messages being passed.
- To understand the flow of control over a set of collaborating objects or to understand which objects collaborate to support business processes, a communication diagram can be used.
- For time ordering of the messages, a sequence diagram should be used.
- In some cases, both can be used to more fully understand the dynamic activity of the system.

# An example

## sd Make Appt Use Case



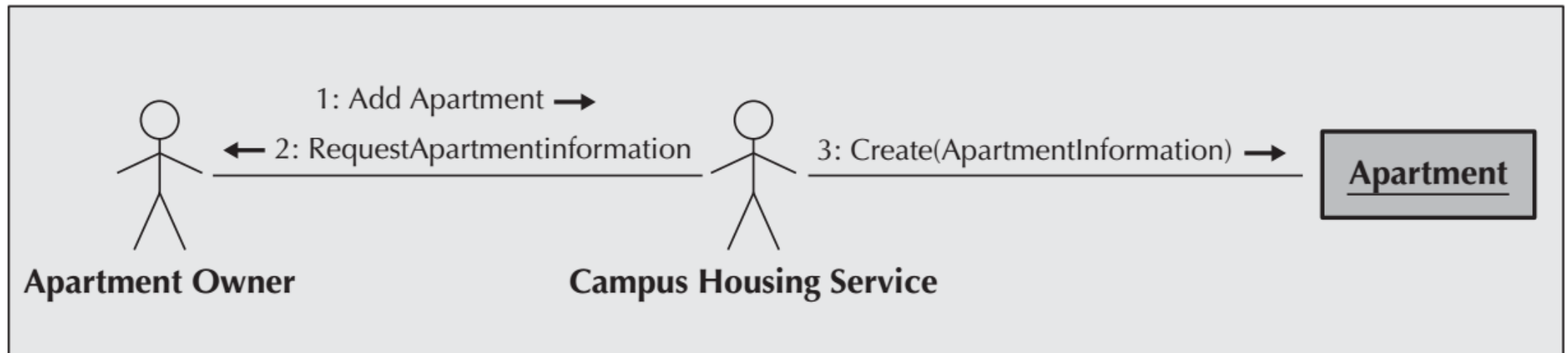
# Elements of a Communication Diagram

Term and Definition	Symbol
<b>An actor:</b> <ul style="list-style-type: none"><li>■ Is a person or system that derives benefit from and is external to the system.</li><li>■ Participates in a collaboration by sending and/or receiving messages.</li><li>■ Is depicted either as a stick figure (default) or, if a nonhuman actor is involved, as a rectangle with &lt;&lt;actor&gt;&gt; in it (alternative).</li></ul>	 <b>anActor</b> 
<b>An object:</b> <ul style="list-style-type: none"><li>■ Participates in a collaboration by sending and/or receiving messages.</li></ul>	
<b>An association:</b> <ul style="list-style-type: none"><li>■ Shows an association between actors and/or objects.</li><li>■ Is used to send messages.</li></ul>	
<b>A message:</b> <ul style="list-style-type: none"><li>■ Conveys information from one object to another one.</li><li>■ Has direction shown using an arrowhead.</li><li>■ Has sequence shown by a sequence number.</li></ul>	
<b>A guard condition:</b> <ul style="list-style-type: none"><li>■ Represents a test that must be met for the message to be sent.</li></ul>	
<b>A frame:</b> <ul style="list-style-type: none"><li>■ Indicates the context of the communication diagram.</li></ul>	

# Difference between Communication Dia. And Sequence Dia.

- Unlike the sequence diagram, the communication diagram **does not have a means to explicitly show an object being deleted or created.** It is assumed that when a delete, destroy, or remove message is sent to an object, it will go out of existence, and a create or new message will cause a new object to come into existence.
- Another difference between the two interaction diagrams is that the communication diagram **never shows returns from message sends,** whereas the sequence diagram can optionally show them.

# Communication Diagram for the Add Apartment Use Case





# Behavioral State Machines

- Some of the classes in the *class diagrams* represent a set of objects that are quite dynamic in that they pass through a variety of states over the course of their existence.
- A behavioral state machine is a dynamic model that shows the different states through which a single object passes during its life in response to events, along with its responses and actions.
- Typically, behavioral state machines are not used for all objects; rather, behavioral state machines are used with complex objects to further define them and to help simplify the design of algorithms for their methods.
- The behavioral state machine shows the different states of the object and what events cause the object to change from one state to another.
- Behavioral state machines should be used to help understand the dynamic aspects of a single class and how its instances evolve over time, Unlike interaction diagrams that show how a particular use case or use-case scenario is executed over a set of classes.

# State

- The *state* of an object is defined by the value of its attributes and its relationships with other objects at a particular point in time.
- The attributes or properties of an object affect the state that it is in;
- Not all attributes or attribute changes will make a difference.
- Is a set of values that describes an object at a specific point in time and represents a point in an object's life in which it satisfies some condition, performs some action, or waits for something to happen.

# Event

- An *event* is something that takes place at a certain point in time and changes a value or values that describe an object, which, in turn, changes the object's state.
- It can be a designated condition becoming true, the receipt of the call for a method by an object, or the passage of a designated period of time.

# Transition

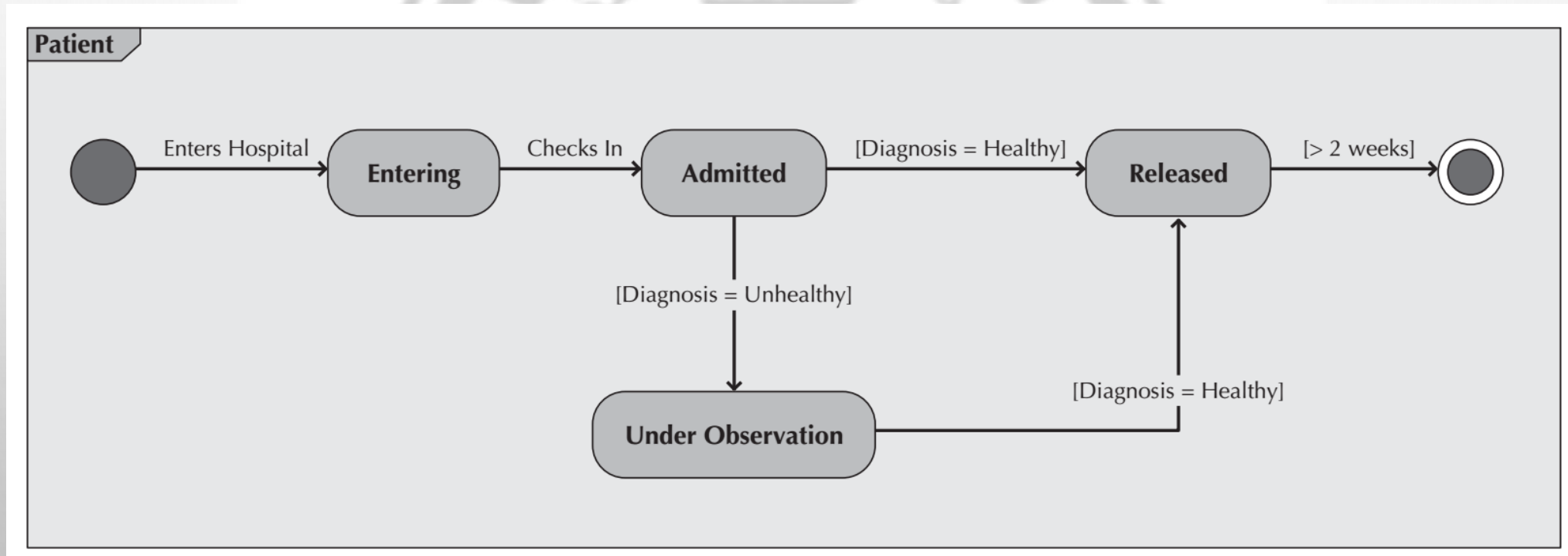
- A *transition* is a relationship that represents the movement of an object from one state to another state.
- Some transitions have a guard condition. A *guard condition* is a Boolean expression that includes attribute values, which allows a transition to occur only if the condition is true.
- An object typically moves from one state to another based on the outcome of an action triggered by an event.

# Action and Activity

- An *action* is an atomic, non-decomposable process that cannot be interrupted. From a practical perspective, actions take zero time, and they are associated with a transition.
- In contrast, an *activity* is a non-atomic, decomposable process that can be interrupted. Activities take a long period of time to complete, and they can be started and stopped by an action.






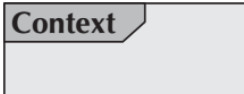


# An example

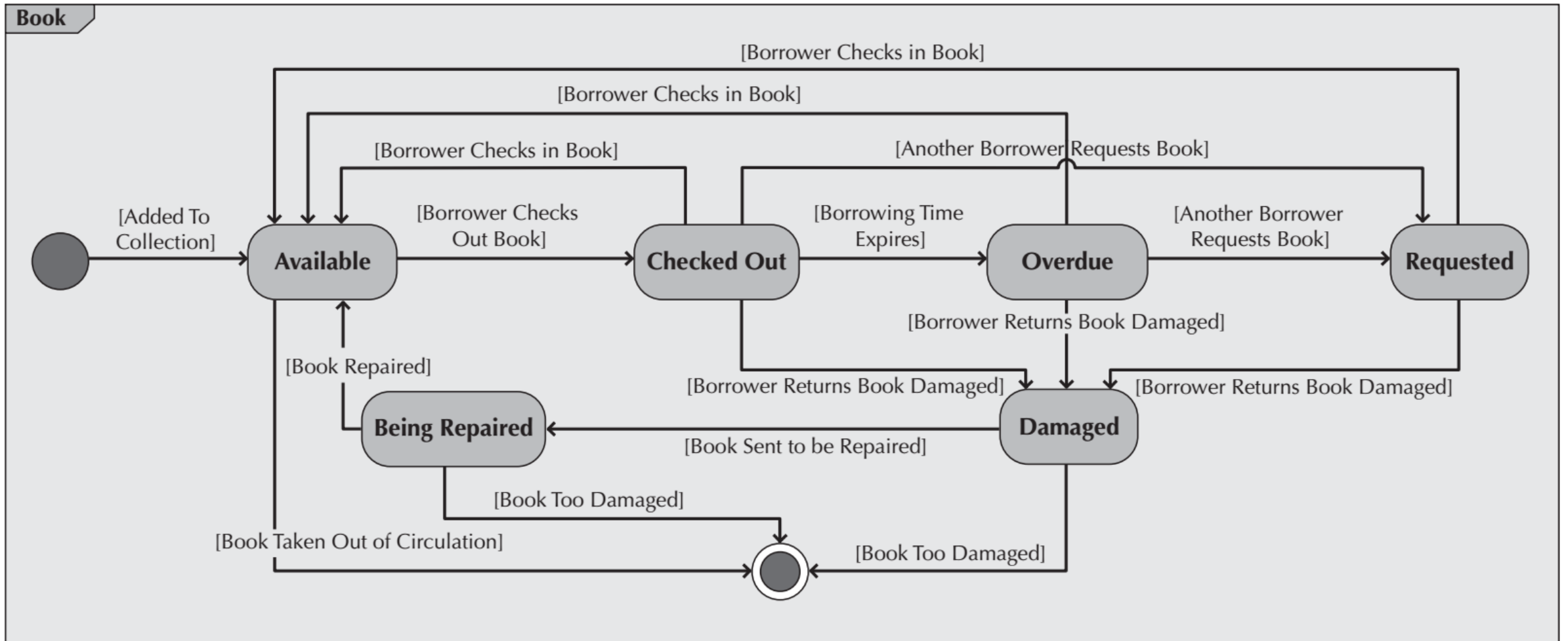




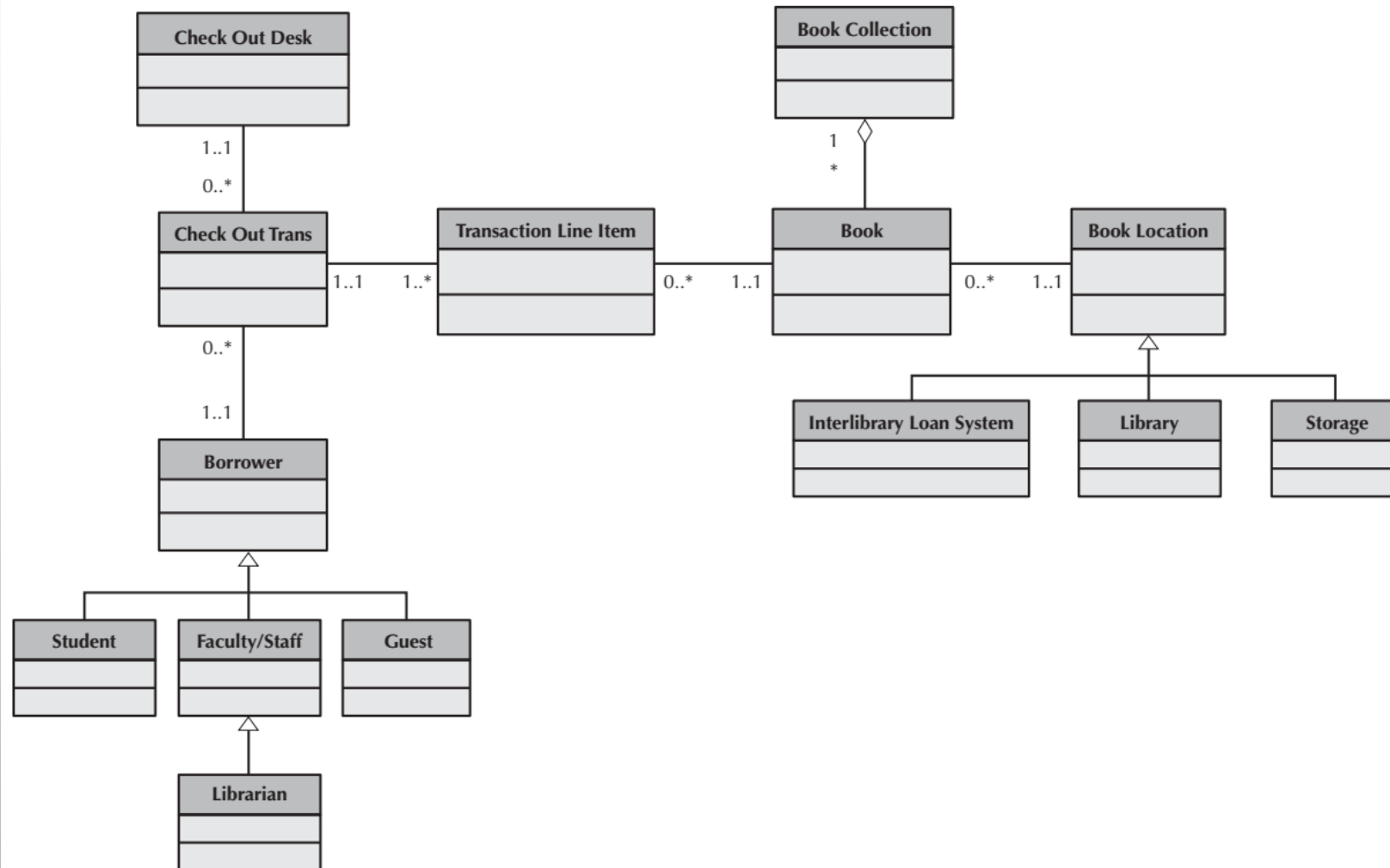
# Elements of a Behavioral State Machine

Term and Definition	Symbol
<b>A state:</b> <ul style="list-style-type: none"><li>■ Is shown as a rectangle with rounded corners.</li><li>■ Has a name that represents the state of an object.</li></ul>	
<b>An initial state:</b> <ul style="list-style-type: none"><li>■ Is shown as a small, filled-in circle.</li><li>■ Represents the point at which an object begins to exist.</li></ul>	
<b>A final state:</b> <ul style="list-style-type: none"><li>■ Is shown as a circle surrounding a small, filled-in circle (bull's-eye).</li><li>■ Represents the completion of activity.</li></ul>	
<b>An event:</b> <ul style="list-style-type: none"><li>■ Is a noteworthy occurrence that triggers a change in state.</li><li>■ Can be a designated condition becoming true, the receipt of an explicit signal from one object to another, or the passage of a designated period of time.</li><li>■ Is used to label a transition.</li></ul>	
<b>A transition:</b> <ul style="list-style-type: none"><li>■ Indicates that an object in the first state will enter the second state.</li><li>■ Is triggered by the occurrence of the event labeling the transition.</li><li>■ Is shown as a solid arrow from one state to another, labeled by the event name.</li></ul>	
<b>A frame:</b> <ul style="list-style-type: none"><li>■ Indicates the context of the behavioral state machine.</li></ul>	

# Behavioral State Machine for an Instance of the Book Class in the Library Book Collection Management System



# Class Diagram for the Library Book Collection Management System

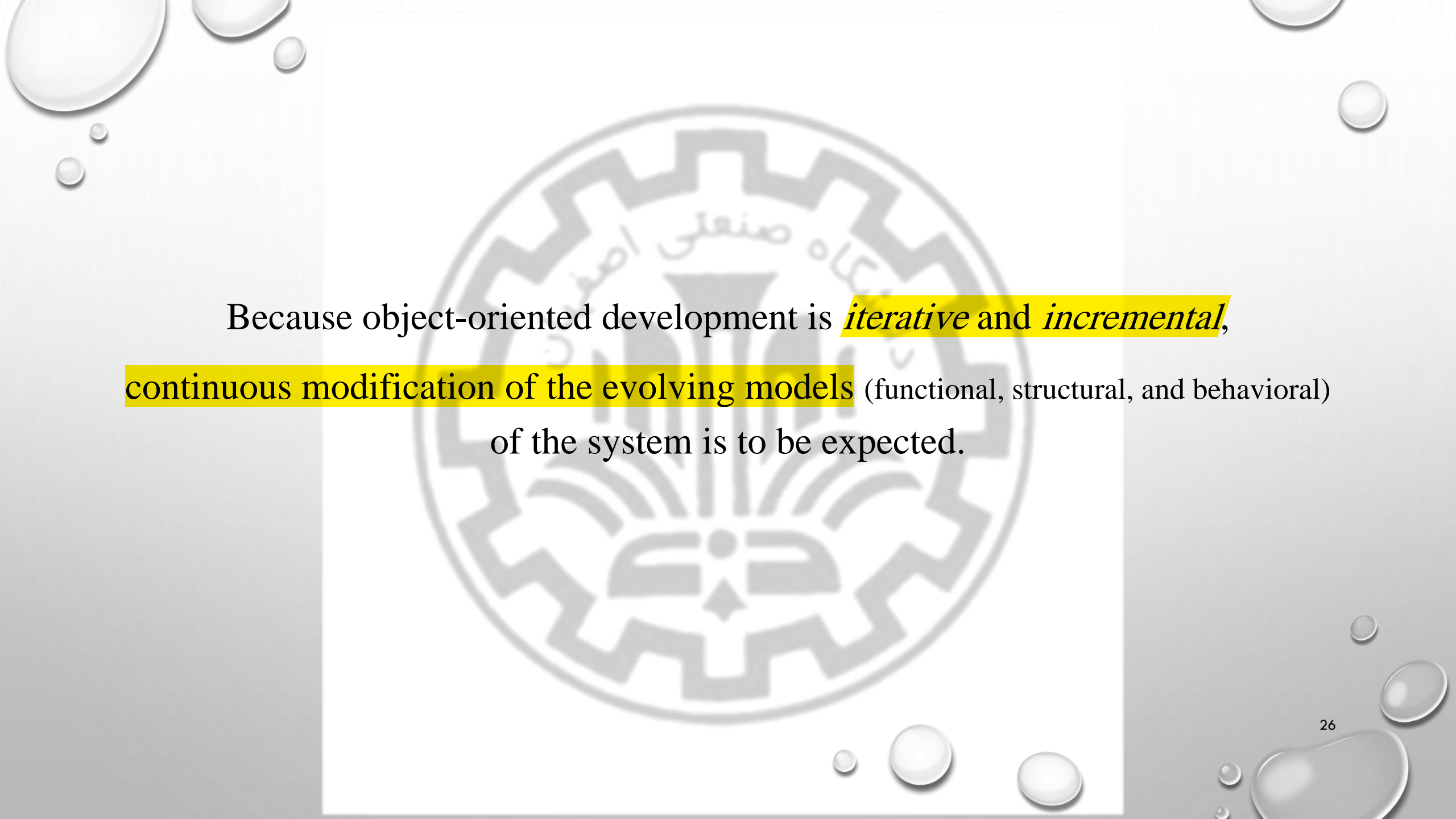


# Verifying and Validating The Behavioral Models

- First, every actor and object included on a sequence diagram must be included as an actor and an object on a communication diagram, and vice versa.
- Second, every message that is included on a sequence diagram must appear as a message on an association in the corresponding communication diagram, and vice versa.

# Verifying and Validating The Behavioral Model(Cnt'd)

- Third, if a **guard condition appears on a message** in the **sequence diagram**, there must be an **equivalent guard condition** on the corresponding **communication diagram**, and vice versa.
- Fourth, the **sequence number** included as part of a message label in a **communications diagram** implies the **sequential order** in which the message will be sent. Therefore, it must correspond to the **top-down ordering of the messages** being sent on the **sequence diagram**.
- Fifth, **all transitions** contained in a **behavior state machine** must be **associated with a message** being sent on a **sequence and communication diagram**.



Because object-oriented development is *iterative* and *incremental*,  
continuous modification of the evolving models (functional, structural, and behavioral)  
of the system is to be expected.



# What should you do for your project?

1. Create behavioral models.

*We will work in the lab.*

# Reference

- **Dennis, Wixon, Tegarden**, “System Analysis and Design, An Object Oriented Approach with UML”, 5th Edition, 2015.