



Software Engineering I

Dr. Elham Mahmoudzadeh
Isfahan University of Technology
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, Isfahan) written along its top arc. Inside the gear is a stylized emblem featuring a central star and crescent-like shape with radiating lines.

Chapter 4

Functional 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



Business Process Modelling

- Business process models describe the different activities that, when combined, support a business process.
- From an object-oriented perspective, these processes cut across multiple roles.

Activity diagrams are used to model the behavior in a business process.

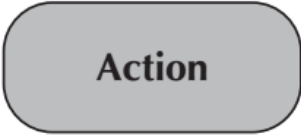

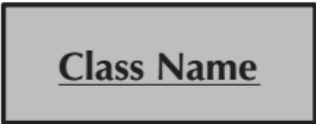


Activity diagram

- It can be used to model everything from a high-level business workflow.
- Can model a business process independent of any object implementation.
- It can be used at a high level as well as at a low level of abstraction.
- At a conceptual level, an activity is a task that needs to be done, whether by a human or a computer.
- At an implementation level, an activity is a method or a class.




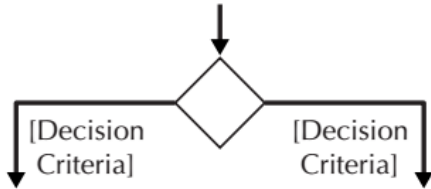
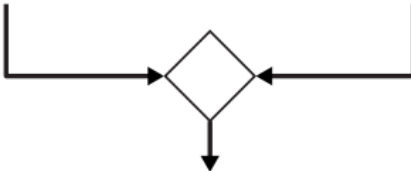
When to use an activity diagram

- It should be used only when it adds value to the project.
- Ask the following question: Does it add value or is it redundant?
- An activity diagram can be used to accomplish the following tasks.
 1. Depict the flow of control from activity to activity.
 2. Help in use case analysis to understand what actions need to take place.
 3. Help in identifying extensions in a use case.
 4. Model work flow and business processes.
 5. Model the sequential and concurrent steps in a computation process.

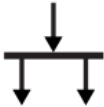
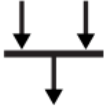

Elements of an Activity Diagram(I)

An action: <ul style="list-style-type: none">■ Is a simple, nondecomposable piece of behavior.■ Is labeled by its name.	
An activity: <ul style="list-style-type: none">■ Is used to represent a set of actions.■ Is labeled by its name.	
An object node: <ul style="list-style-type: none">■ Is used to represent an object that is connected to a set of object flows.■ Is labeled by its class name.	
A control flow: <ul style="list-style-type: none">■ Shows the sequence of execution.	
An object flow: <ul style="list-style-type: none">■ Shows the flow of an object from one activity (or action) to another activity (or action).	

Elements of an Activity Diagram(II)

An initial node: <ul style="list-style-type: none">■ Portrays the beginning of a set of actions or activities.	
A final-activity node: <ul style="list-style-type: none">■ Is used to stop all control flows and object flows in an activity (or action).	
A final-flow node: <ul style="list-style-type: none">■ Is used to stop a specific control flow or object flow.	
A decision node: <ul style="list-style-type: none">■ Is used to represent a test condition to ensure that the control flow or object flow only goes down one path.■ Is labeled with the decision criteria to continue down the specific path.	
A merge node: <ul style="list-style-type: none">■ Is used to bring back together different decision paths that were created using a decision node.	

Elements of an Activity Diagram(III)

A fork node: Is used to split behavior into a set of parallel or concurrent flows of activities (or actions)	
A join node: Is used to bring back together a set of parallel or concurrent flows of activities (or actions)	
A swimlane: Is used to break up an activity diagram into rows and columns to assign the individual activities (or actions) to the individuals or objects that are responsible for executing the activity (or action) Is labeled with the name of the individual or object responsible	

Actions and Activities

- Can represent manual or computerized behavior.
- Depicted in an activity diagram as a rounded rectangle.
- They should have a name that begins with a verb and ends with a noun (e.g., Get Patient Information)
- Names should be short, yet contain enough information so that the reader can easily understand exactly what they do.
- The only difference between an action and an activity is that an activity can be decomposed further into a set of activities and/or actions, whereas an action represents a simple non-decomposable piece of the overall behavior being modeled.
- In most cases, each activity is associated with a use case.

Control Flows

- *Control flows* model the paths of execution through a business process.
- A control flow is portrayed as a solid line with an arrowhead on it showing the direction of flow.
- Control flows can be attached only to actions or activities.

Initial node

- Portrays the beginning of a set of actions or activities.
- Is shown as a small filled-in circle.



final-activity node

- Is used to stop the process being modeled.
- Any time a final-activity node is reached, all actions and activities are ended immediately, regardless of whether they are completed.
- Is represented as a circle surrounding a small, filled-in circle.

Final-flow node

- Is similar to a final-activity node, except that it stops a specific path of execution through the business process but allows the other concurrent or parallel paths to continue.
- Is shown as a small circle with an X in it.

Decision node

- Is used to represent the actual test condition that determines which of the paths exiting the decision node is to be traversed.
- In this case, each exiting path must be labeled with a guard condition.
- A *guard condition* represents the value of the test for that particular path to be executed.

Merge node

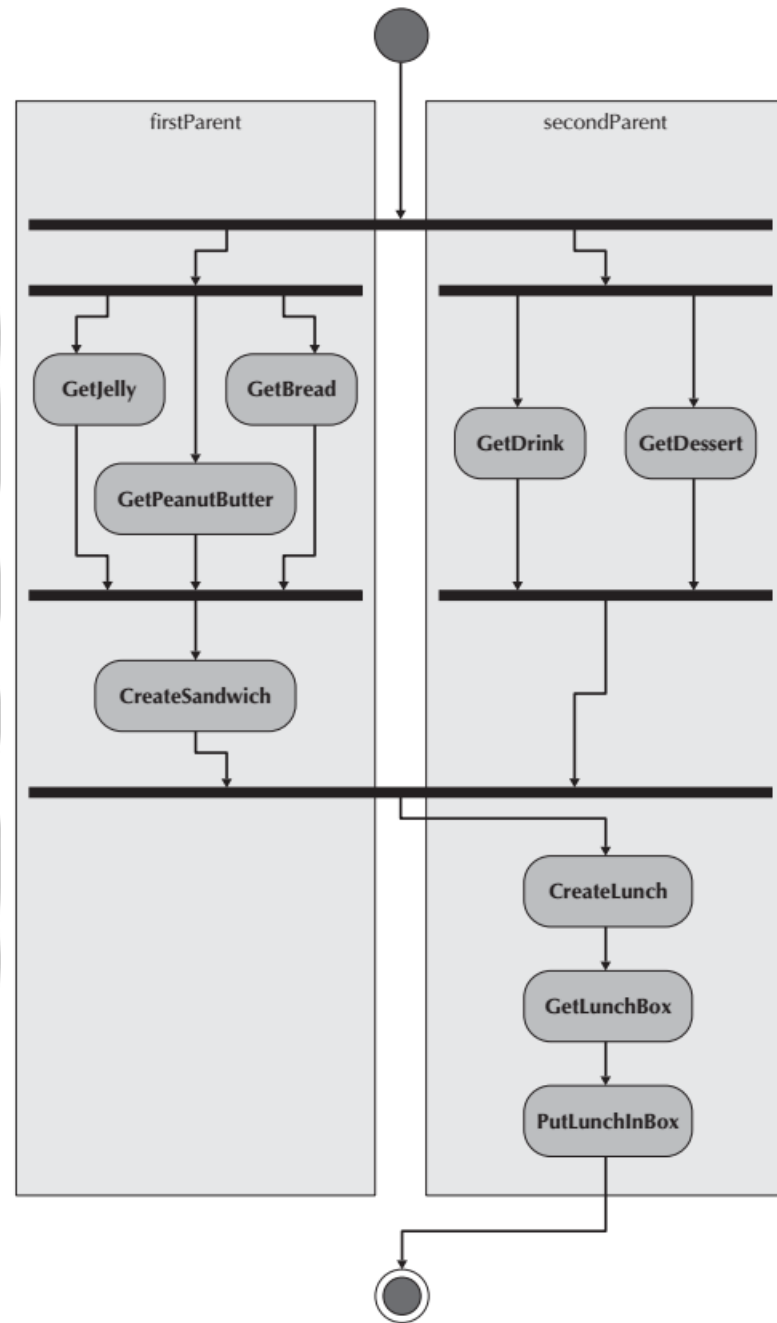
- Is used to bring back together multiple mutually exclusive paths that have been split based on earlier.
- However, sometimes, for clarity, it is better not to use a merge node.

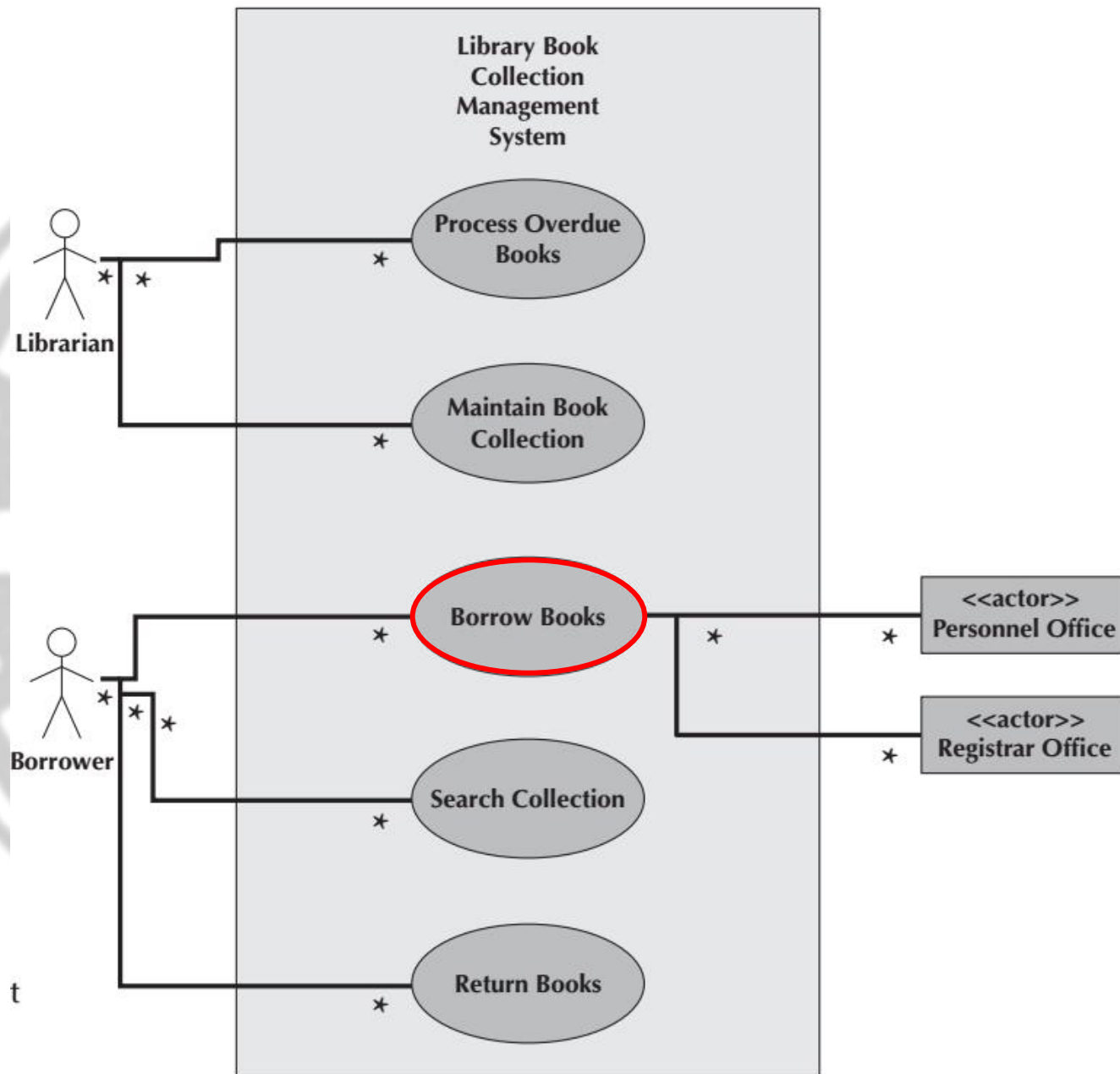
Fork and Join nodes

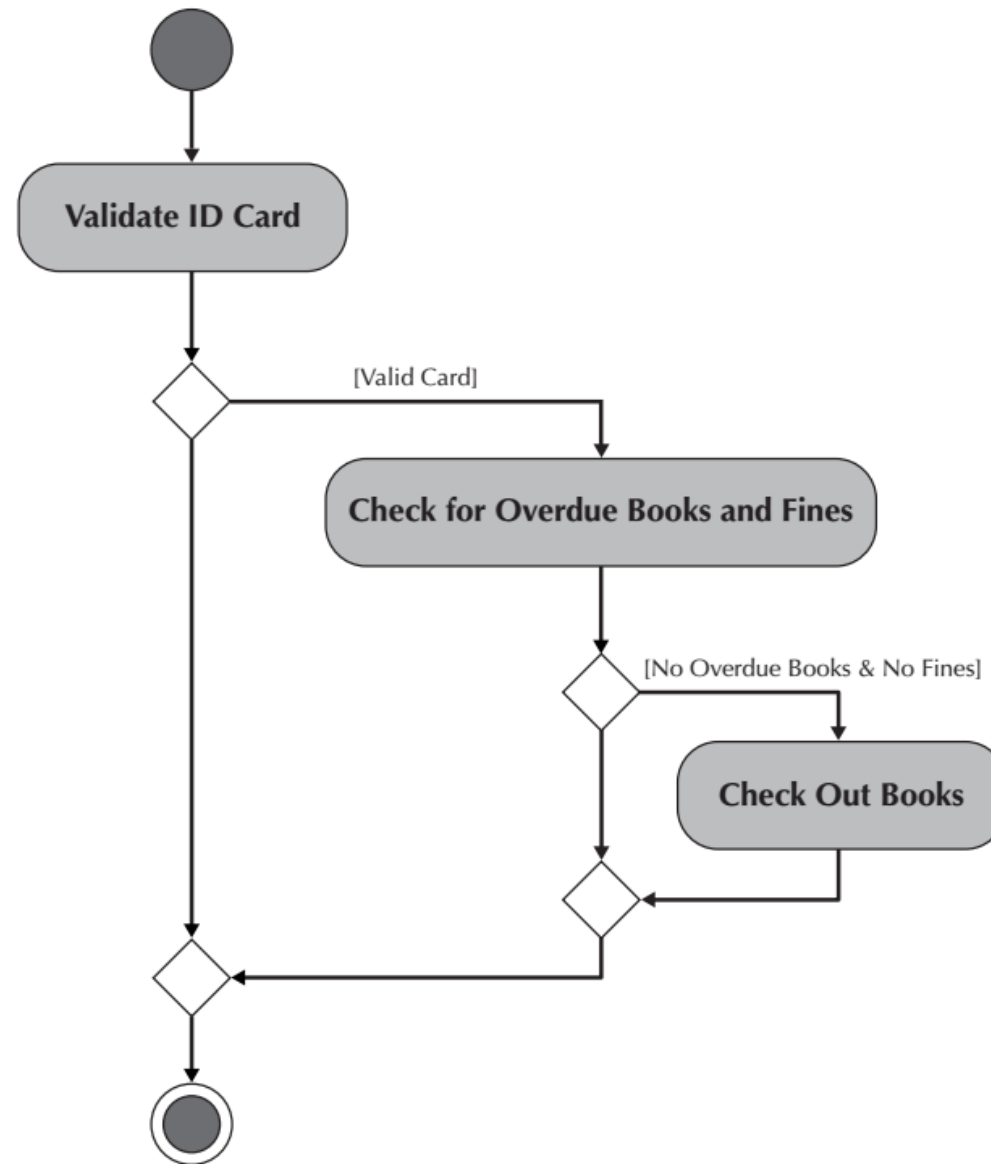
- The fork and join nodes allow parallel and concurrent processes to be modeled .
- The *fork node* is used to split the behavior of the business process into multiple parallel or concurrent flows. Unlike the decision node, the paths are not mutually exclusive.
- The *join node* simply brings back together the separate parallel or concurrent flows in the business process into a single flow.

Swimlanes

- However, there are times when it helps to break up an activity diagram in such a way that it can be used to assign responsibility to objects or individuals who would actually perform the activity. This is especially useful when modeling a business workflow and is accomplished through the use of *swimlanes*.





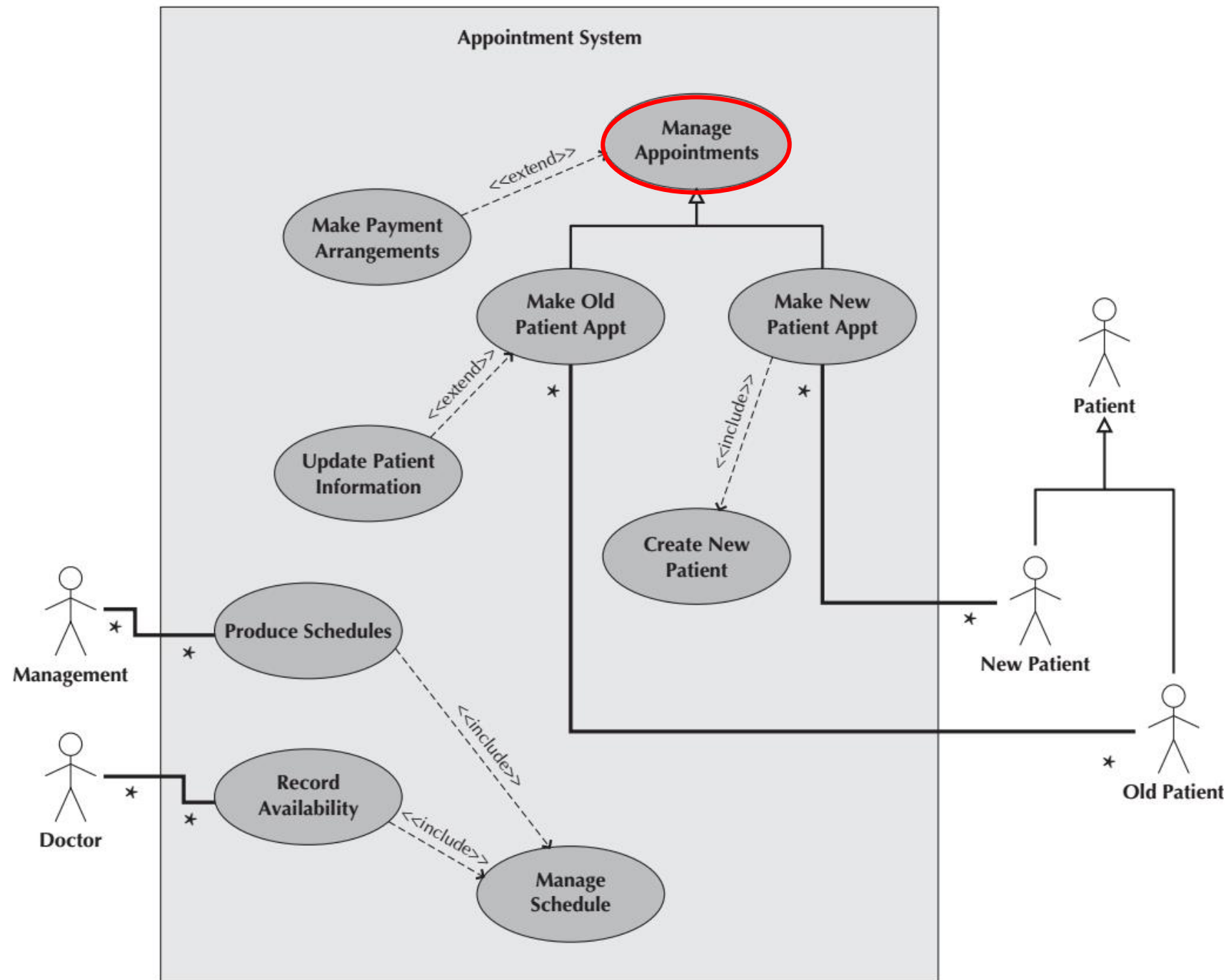


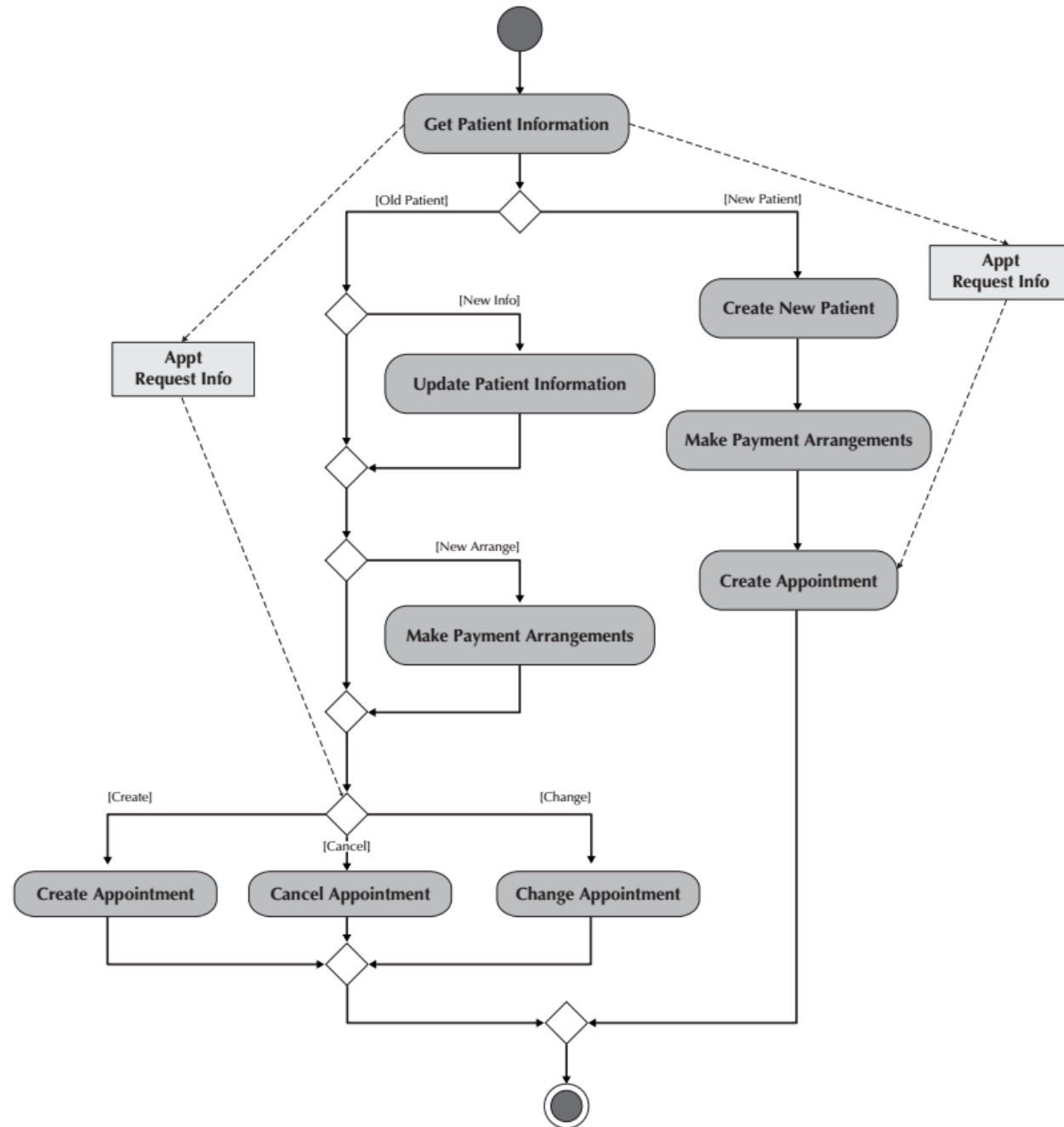
Object nodes

- Activities and actions typically modify or transform objects. *Object nodes* model these objects in an activity diagram.
- The name of the class of the object is written inside the rectangle.
- Essentially, object nodes represent the flow of information from one activity to another activity.

Object Flows

- *Object flows* model the flow of objects through a business process.
- Because activities and actions modify or transform objects, object flows are necessary to show the actual objects that flow into and out of the actions or activities.
- An object flow is depicted as a dashed line with an arrowhead on it showing the direction of flow.
- An individual object flow must be attached to an action or activity on one end and an object node on the other end.





Control Nodes

- Initial,
- Final-activity,
- Final-flow,
- Decision,
- Merge,
- Fork,
- Join



What should you do for your project?

1. Create Activity diagram.

We will work in the lab.

Reference

- **Dennis, Wixon, Tegarden**, “System Analysis and Design, An Object Oriented Approach with UML”, 5th Edition, 2015.
- **Valacich, J. S., J. F. George**, “Modern systems analysis and design”, 8th Edition, 2017.