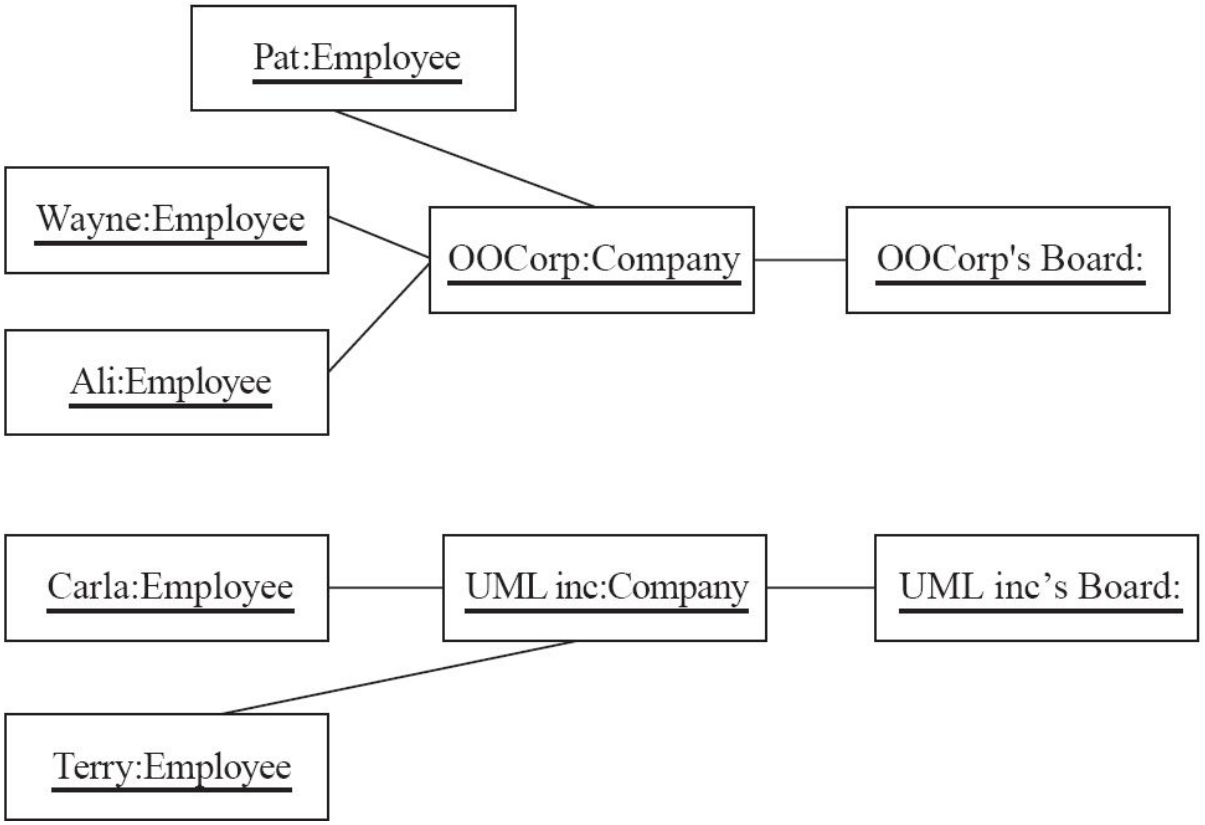
SWE 2022-23

Object Diagrams

- A *link* is an instance of an association
  - In the same way that we say an object is an instance of a class

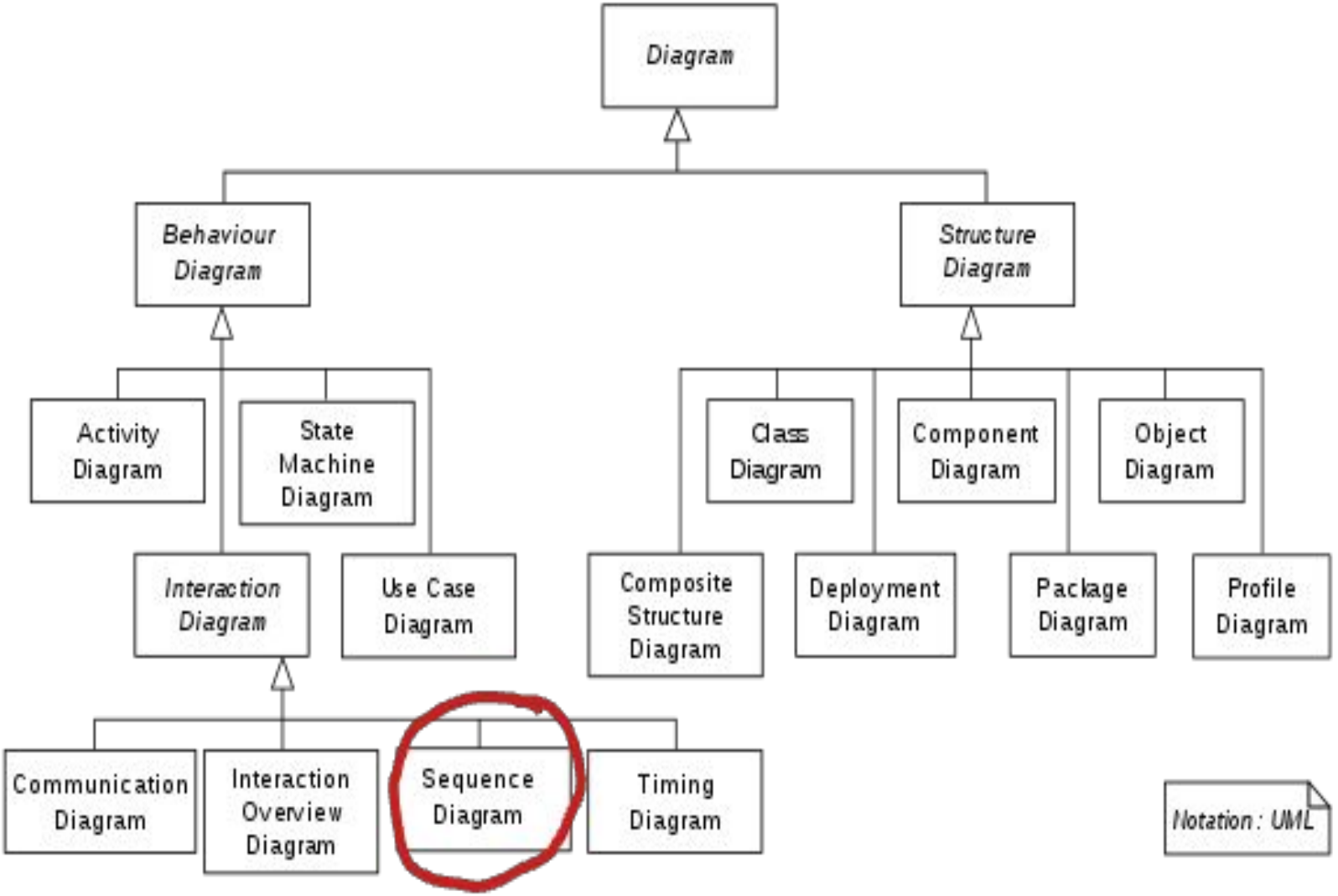
A class diagram can generate an infinite number of object diagrams



- Associations describe the relationships that will exist between *instances* at run time.
  - When you show an instance diagram generated from a class diagram, there will be an instance of *both* classes joined by an association

- Generalizations describe relationships between *classes* in class diagrams.
  - They do not appear in instance diagrams at all.
  - An instance of any class should also be considered to be an instance of each of that class's superclasses

Interaction Diagram



Interaction Diagrams

- Interaction diagrams are used to model the **dynamic aspects** of a software system
  - They help you to visualize **how the system runs**.
  - An interaction diagram is often built from a use case and a class diagram.
    - The objective is to show how a set of objects accomplish the required interactions with an actor.

Interactions and messages

- Interaction diagrams show **how a set of actors and objects communicate with each other to perform**:
  - The steps of a use case, or
  - The steps of some other piece of functionality.

- The set of steps, taken together, is called **an *interaction***.

- Interaction diagrams can show several different types of communication.
  - E.g. method calls, messages send over the network
  - These are all referred to as *messages*.

Elements found in interaction diagrams

- **Instances of classes**
  - Shown as boxes with the class and object identifier underlined
- **Actors**
  - Use the stick-person symbol as in use case diagrams
- **Messages**
  - Shown as arrows from actor to object, or from object to object

Before creating an interaction diagram

Ideally, we would have

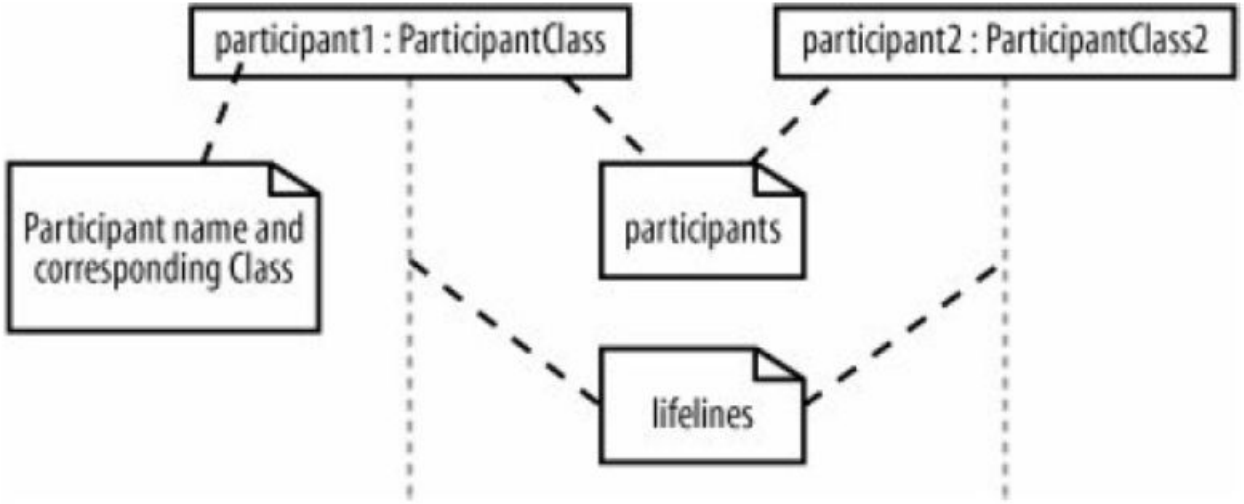
- Class Diagram
- Use Case Diagram

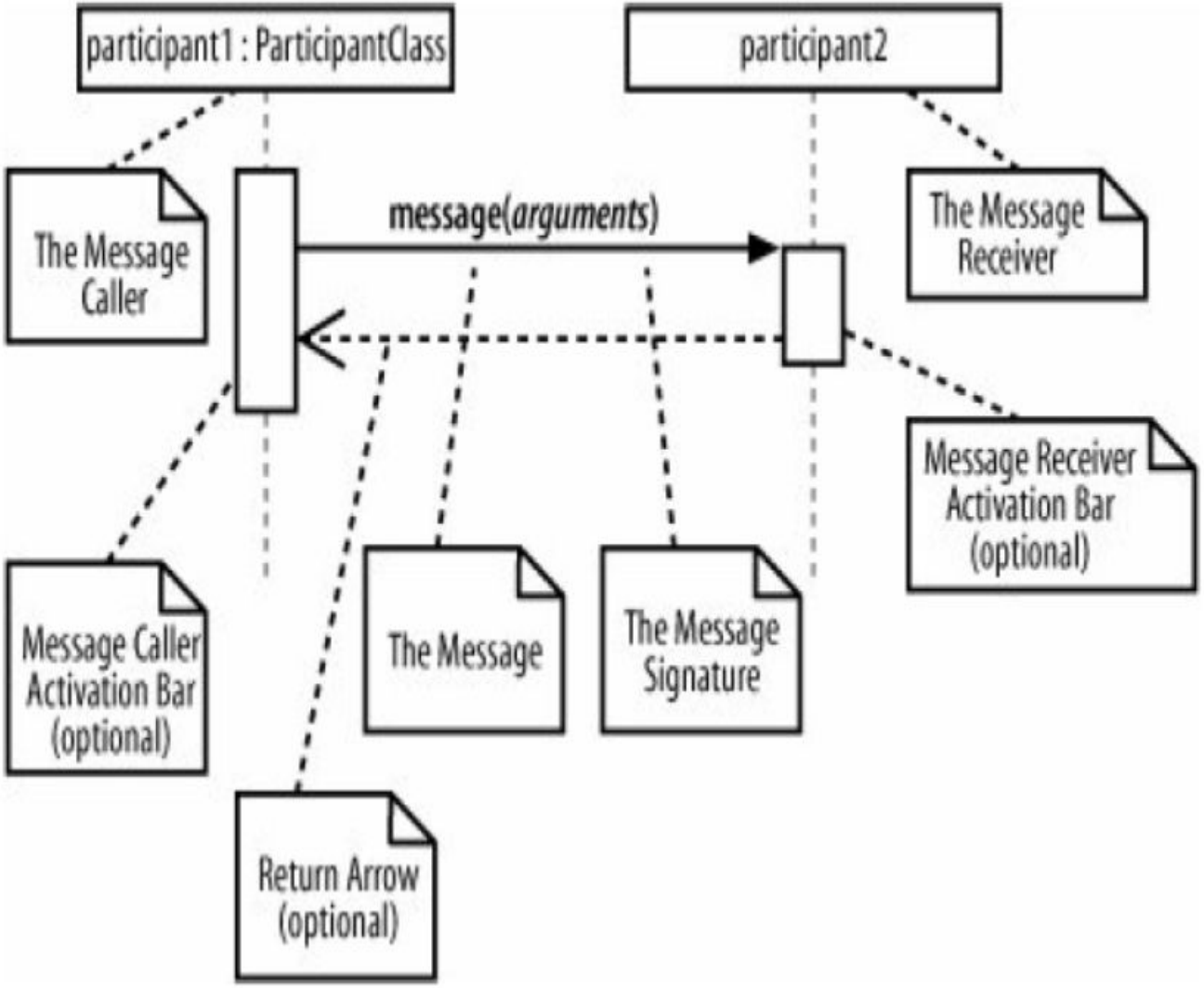
Three main types of Interaction Diagrams

- Sequence
- Timing
- Communication

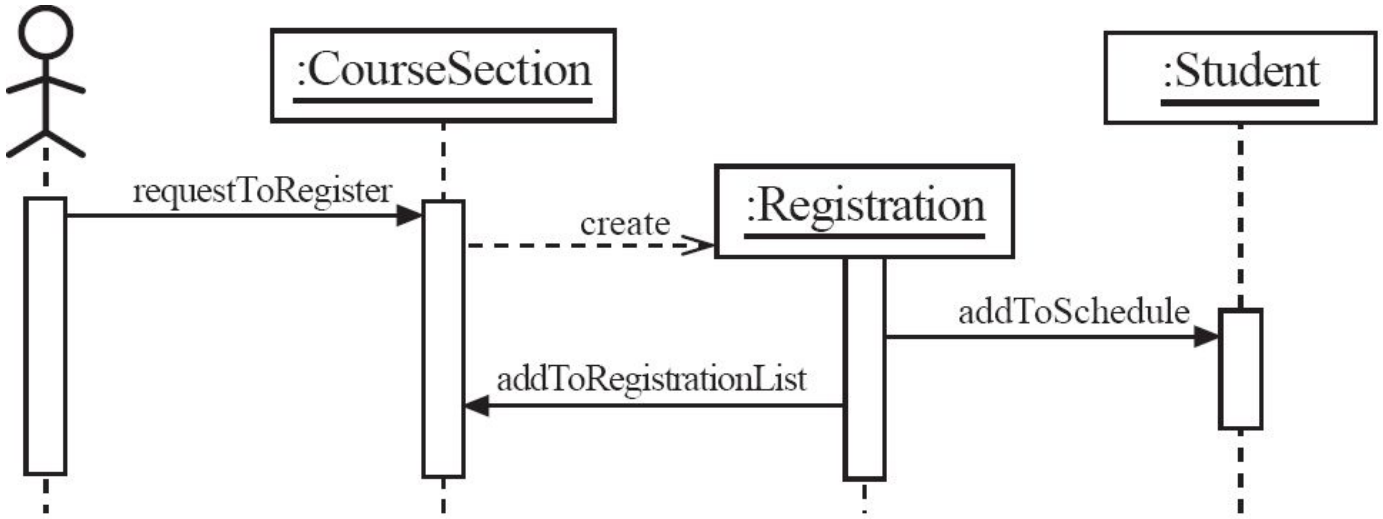
Sequence diagrams

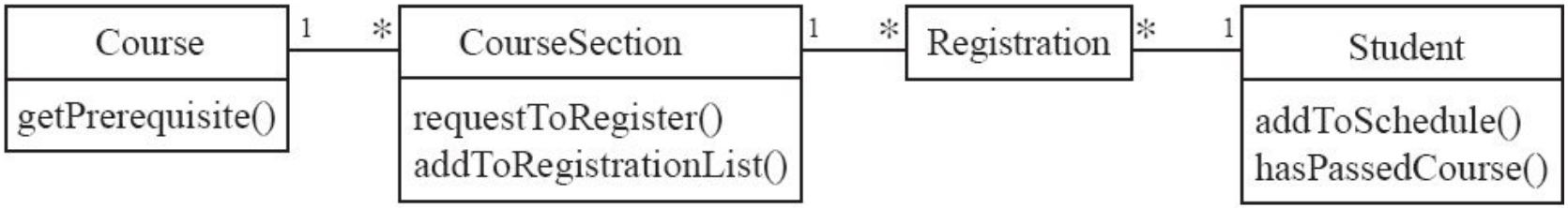
- A sequence diagram **shows the sequence of messages** exchanged by the set of objects performing a certain task
  - The **objects are arranged horizontally** across the diagram.
  - An actor that initiates the interaction is often shown on the left.
  - The **vertical dimension represents time**.
  - A vertical line, called a ***lifeline***, is attached to each object or actor.
  - The lifeline becomes a broad box, called an ***activation*** *box* during the *live activation* period.
  - A message is represented as an arrow between activation boxes of the sender and receiver.
    - A message is labelled and can have an argument list and a return value.


participant1 : ParticipantClass
participant2 : ParticipantClass2
Participant name and corresponding Class
participants
lifelines

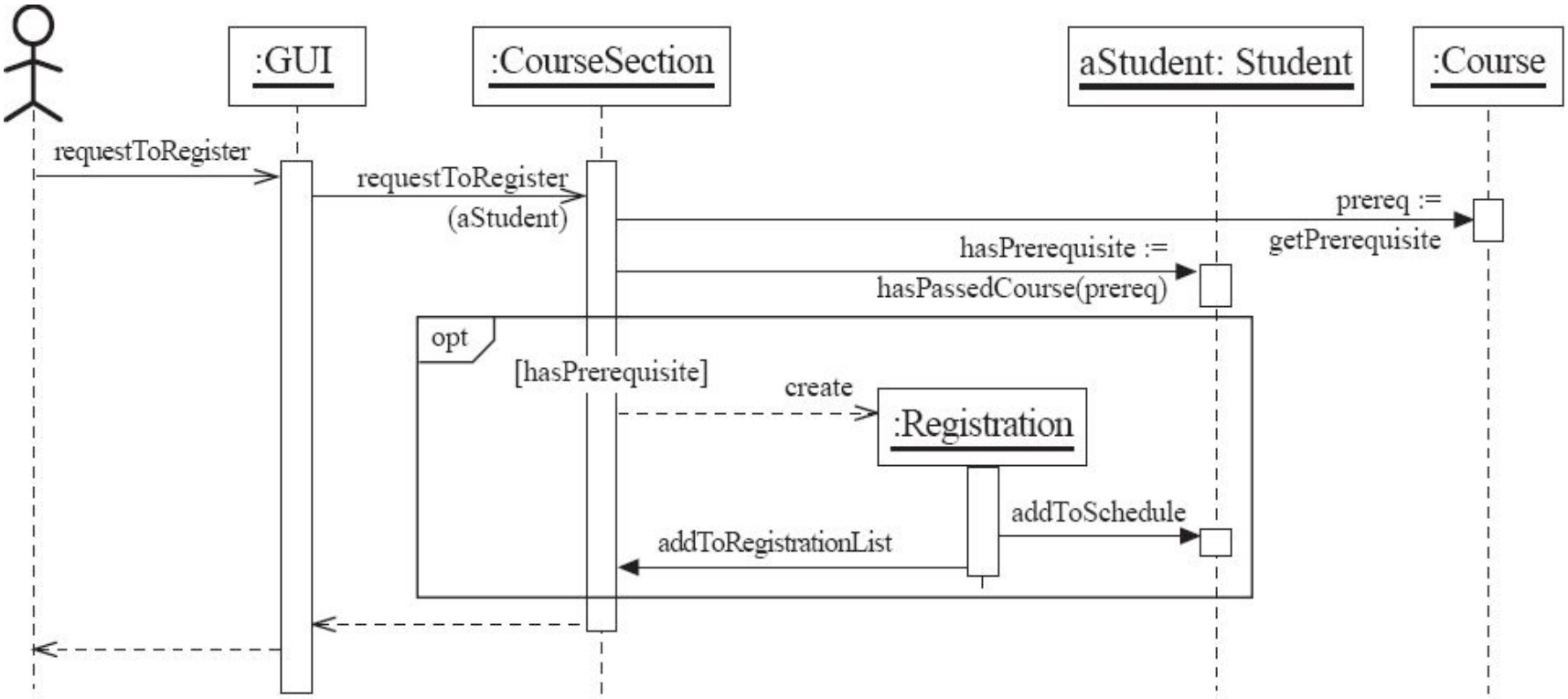

participant1 : ParticipantClass
participant2
The Message Caller
message(arguments)
The Message Receiver
Message Receiver Activation Bar (optional)
Message Caller Activation Bar (optional)
The Message
The Message Signature
Return Arrow (optional)

Sequence diagrams – an example

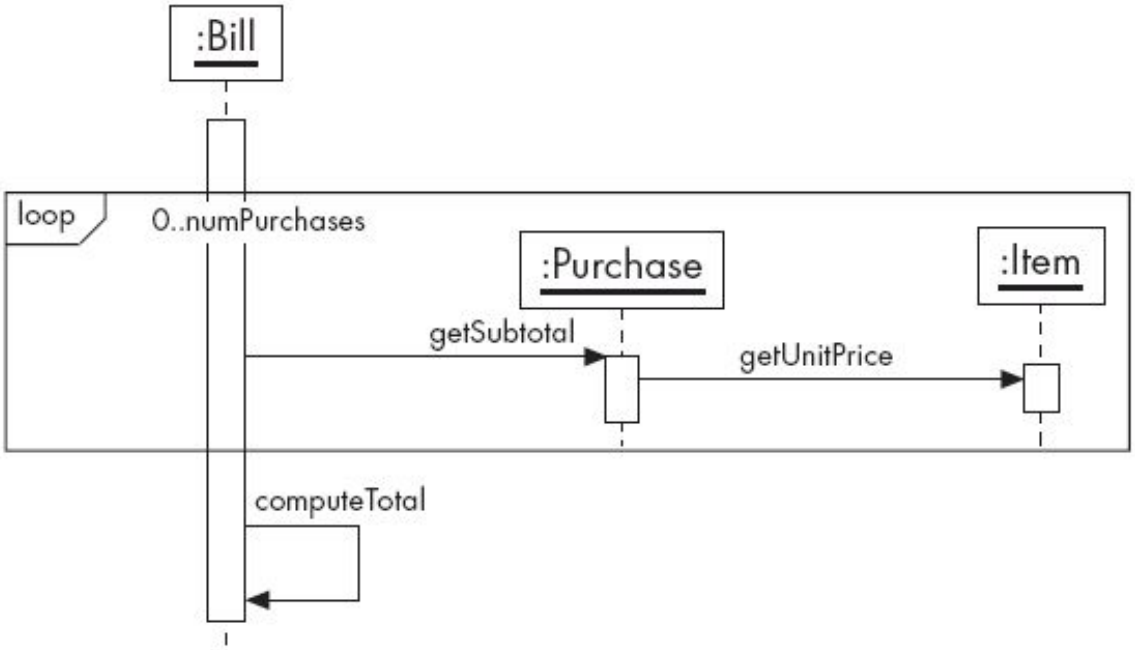


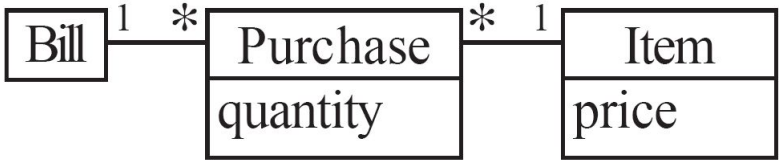


Sequence diagrams – same example, more details

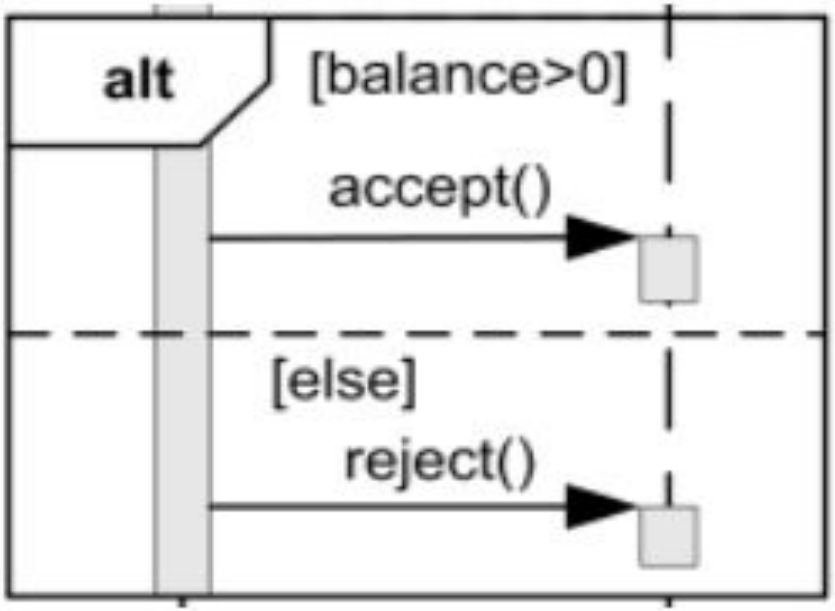


Sequence diagrams – an example with replicated messages



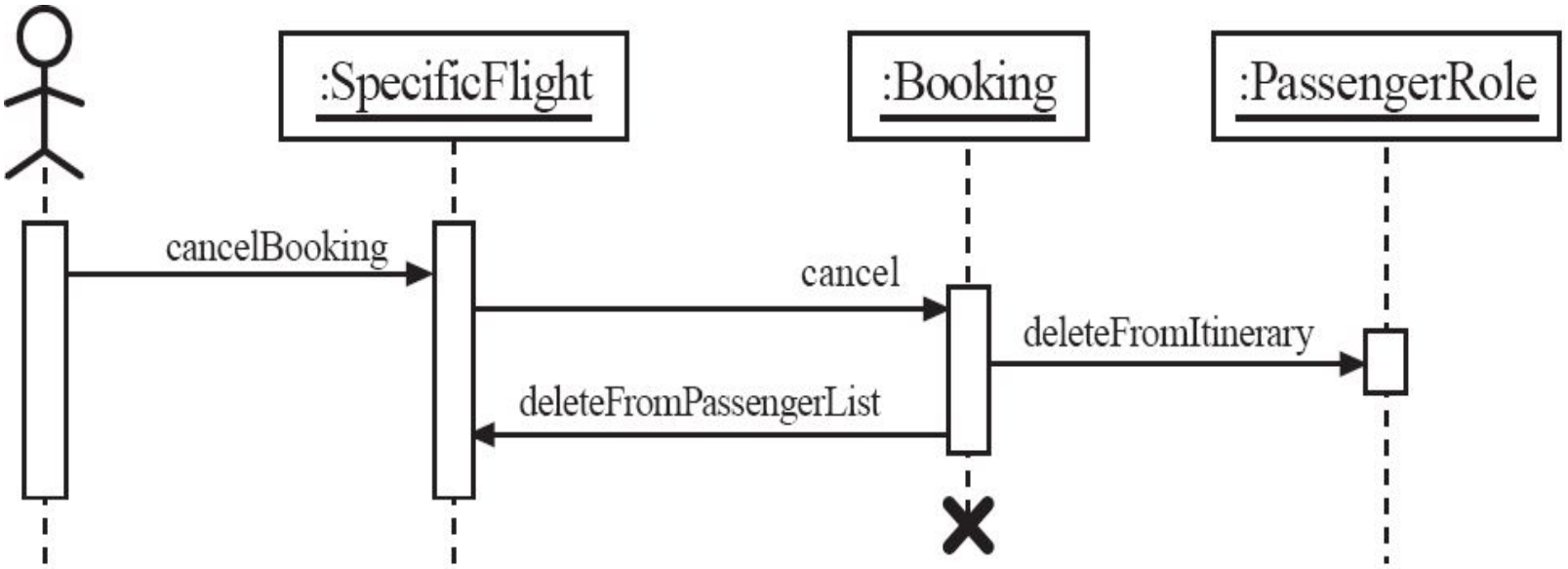


Alternative fragment



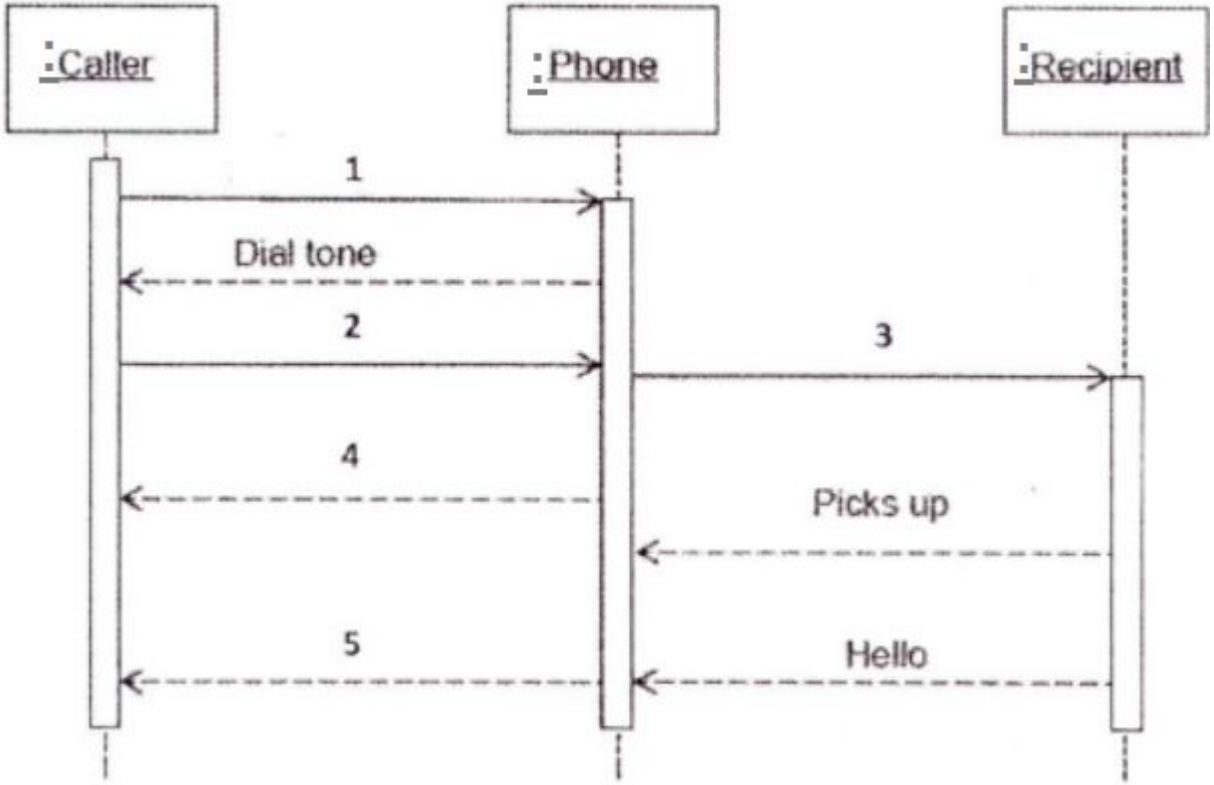
Sequence diagrams – an example with object deletion

- If an object's life ends, this is shown with an X at the end of the lifeline



Exercise:

A caller calls to his/her friend (Recipient): Complete the diagram by choosing the appropriate messages.



Options
Hello
Picks up
Ring
Dial number
Ring notification

UML - STATE DIAGRAM

State Diagrams

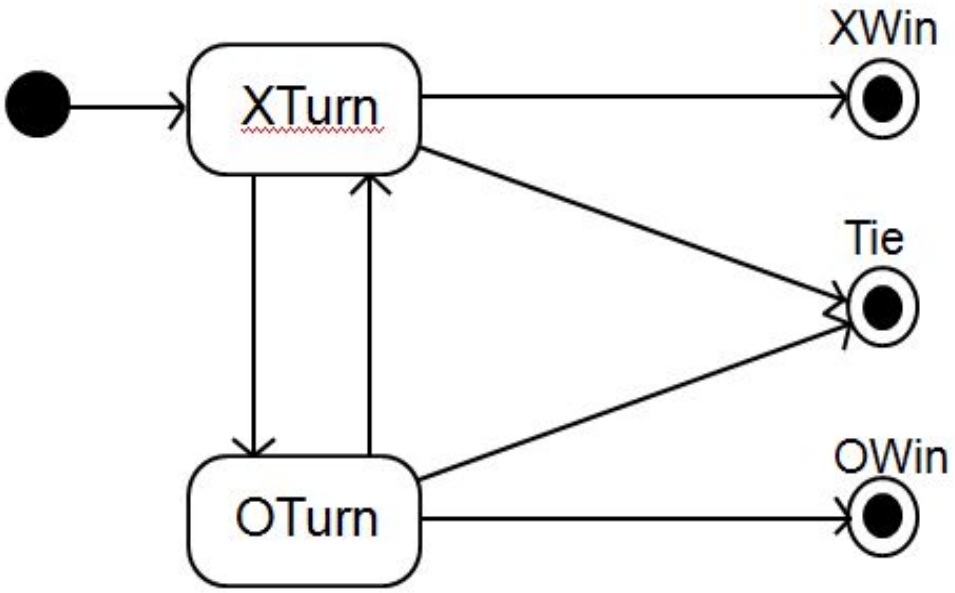
- **Describes the externally visible behavior of a system or of an individual object.**
- **Like directed graph where the nodes are states and the arcs are transitions.**
  - At any given point in time, the system or object is in a certain ***state***.
  - Some ***events*** will cause the system to change state.

States

- At any given point in time, the system is in one state.
- It will remain in this state until an event occurs that causes it to change state.
- A state is represented by a **rounded rectangle** containing the name of the state.
- Special states:
  - A black circle represents the ***start state***
  - A circle with a ring around it represents an ***end state***

State diagrams – an example

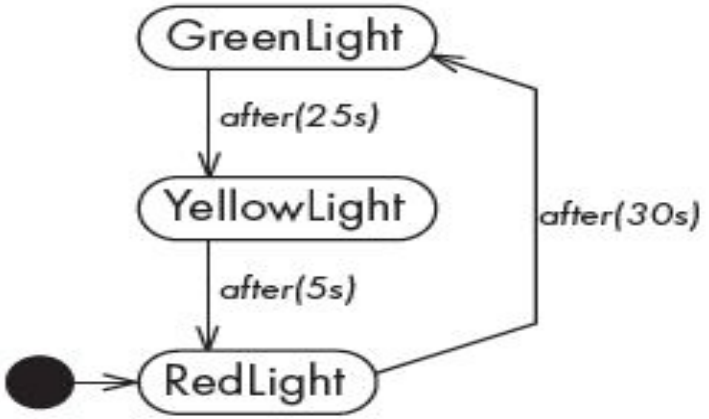
- tic-tac-toe game



Transitions

- A transition represents **a change of state** in response to an event.
  - It is considered to occur **instantaneously**.
- The label on each transition is the event that causes the change of state.
- A transition is rendered as a **solid directed line**.

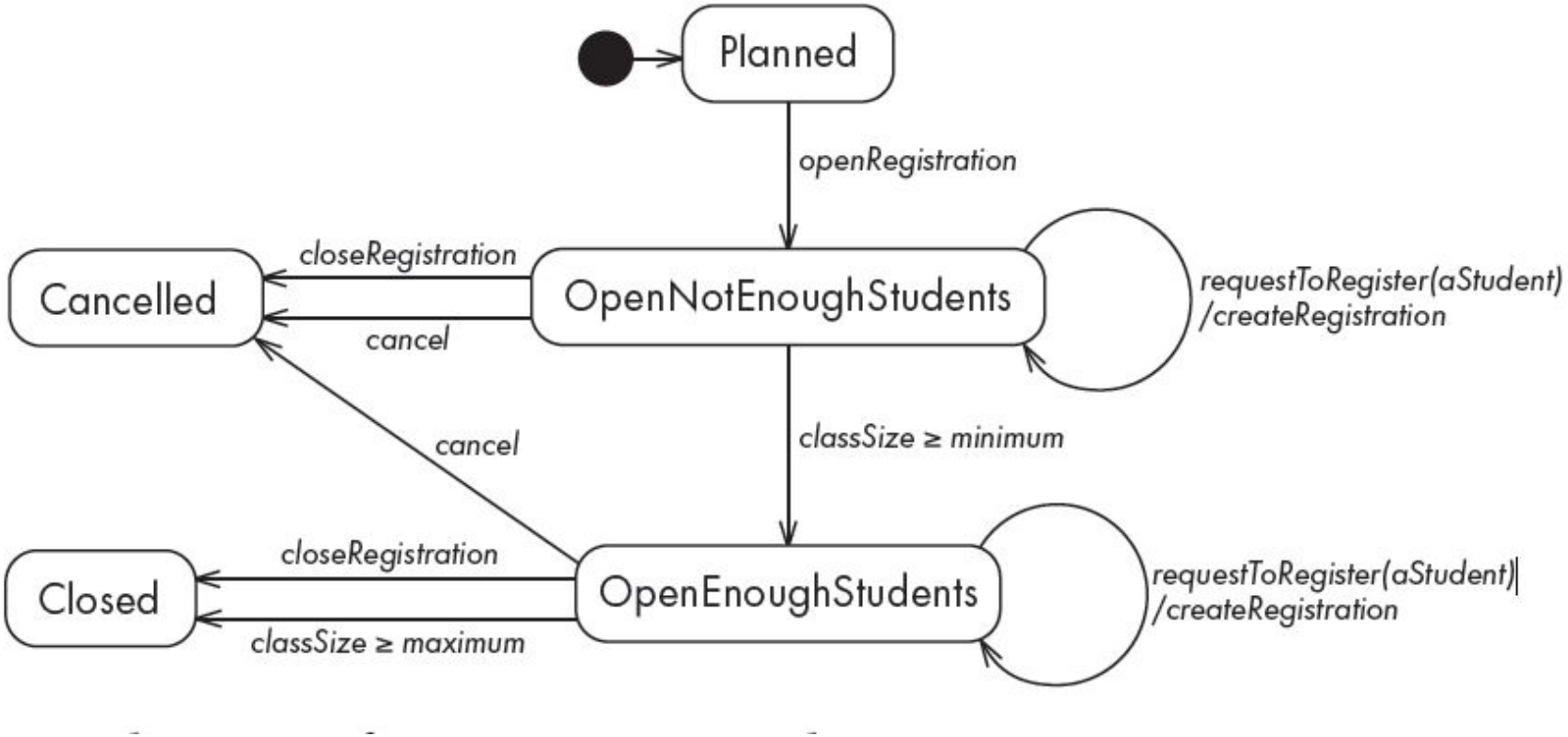
State diagrams of a simple traffic light, illustrating elapsed-time transitions



State diagrams – a … h

time-outs

State diagrams – an example with conditional transitions



State diagram of a CourseSection class

Computations in State Diagram

- Activities
- Actions

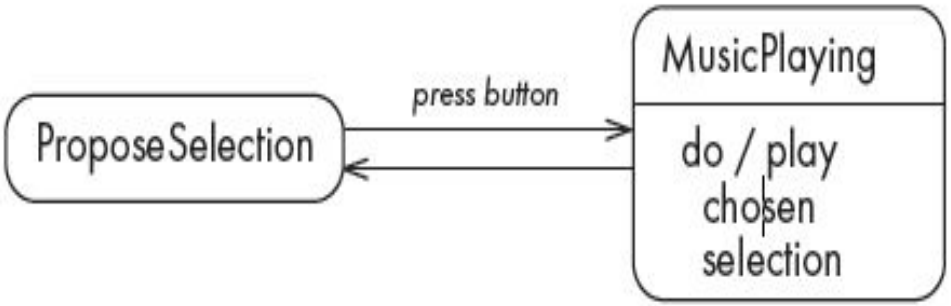
Activities in state diagrams

- An *activity* is something that **takes place while the system is *in* a state.**
  - It **takes a period of time**.
  - The system may take a transition out of the state in response to completion of the activity,
  - Some other outgoing transition may result in:
    - The interruption of the activity, and
    - An early exit from the state.

Activity representation

- An activity is shown textually within a state box by the word '**do**' followed by a '*/*' symbol, and a ***description*** of what is to be done.
- When you have details such as actions in a state, you draw a horizontal line above them to separate them from the state name.

State diagram – an example with activity



State diagram for a jukebox, illustrating an activity in a state

Actions in state diagrams

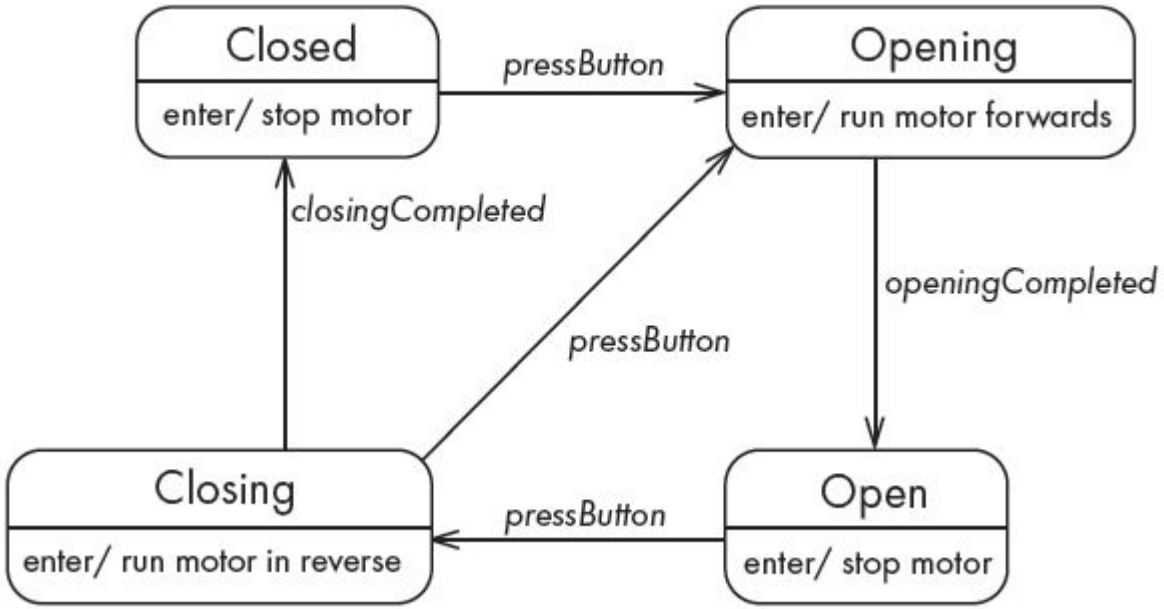
- An *action* is something that takes place effectively *instantaneously*
  - When a particular transition is taken,
  - Upon entry into a particular state, or
  - Upon exit from a particular state
- ***An action should consume no noticeable amount of time***

Representation of action

- An action is always shown preceded by a slash (‘/’) symbol.
- If the action is to be performed during a transition, then the syntax is **event/action**.
- If the action is to be performed when entering or exiting a state, then it is written in the state box with the notation **enter/action** or **exit/action**.

State diagram – an example with actions

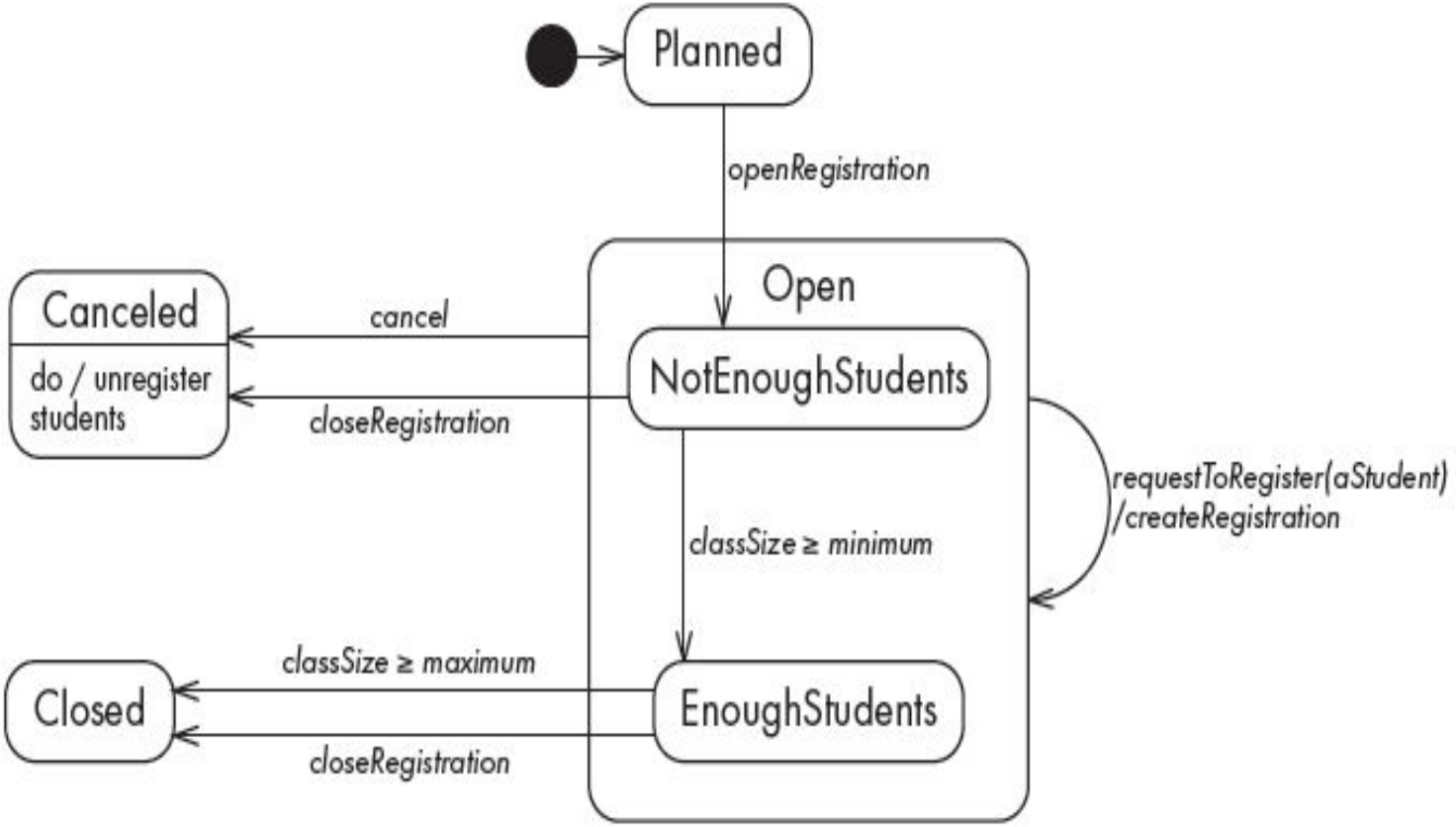
- **State diagram for a garage door opener,**



Nested substates

- A state diagram can be **nested inside a state**.
- The states of the inner diagram are called substates.

State diagram – an example with substates



UML - ACTIVITY DIAGRAM

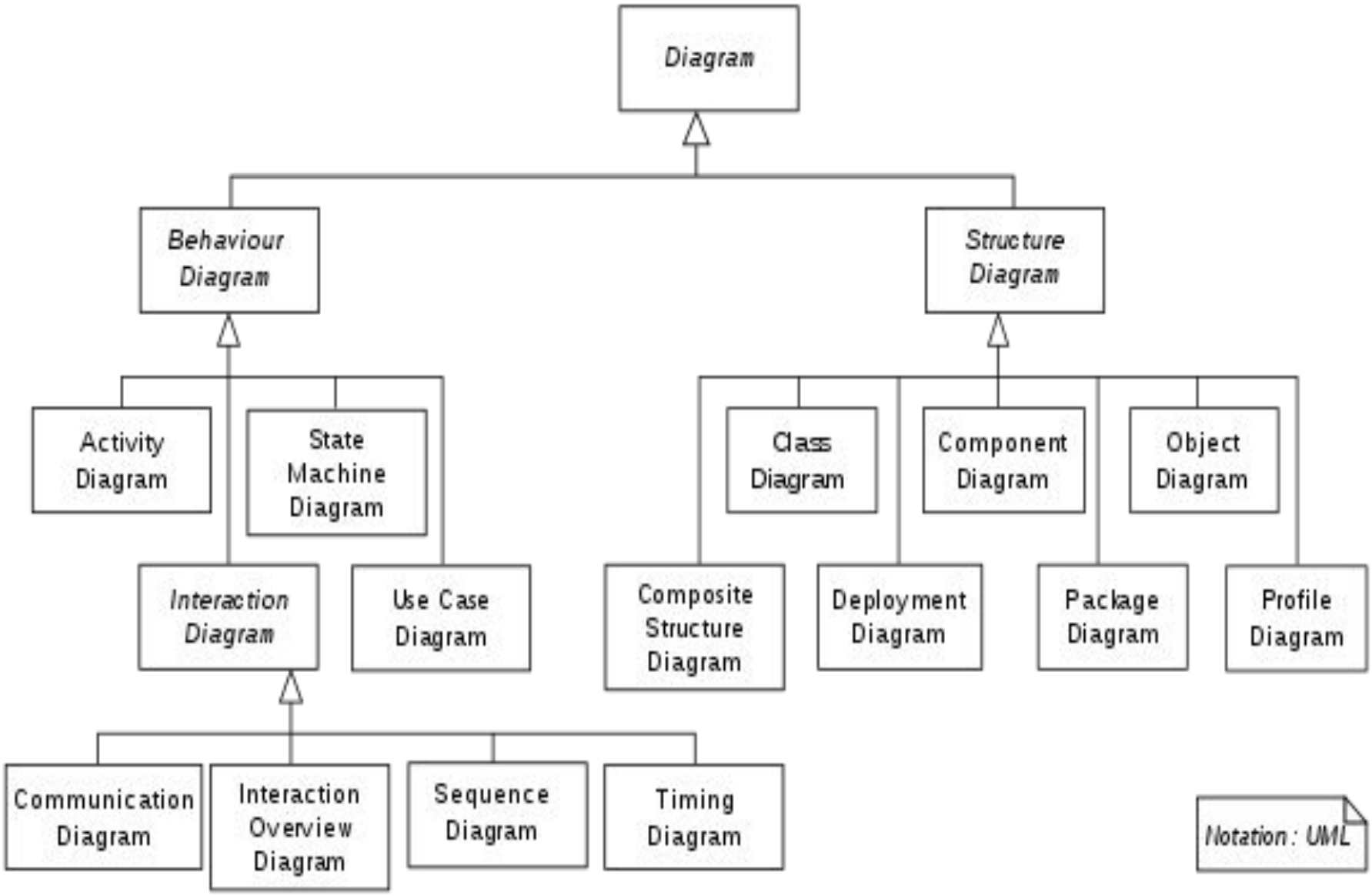


Diagram
Behaviour Diagram
Structure Diagram
Activity Diagram
State Machine Diagram
Interaction Diagram
Use Case Diagram
Class Diagram
Component Diagram
Object Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Profile Diagram
Communication Diagram
Interaction Overview Diagram
Sequence Diagram
Timing Diagram
Notation : UML

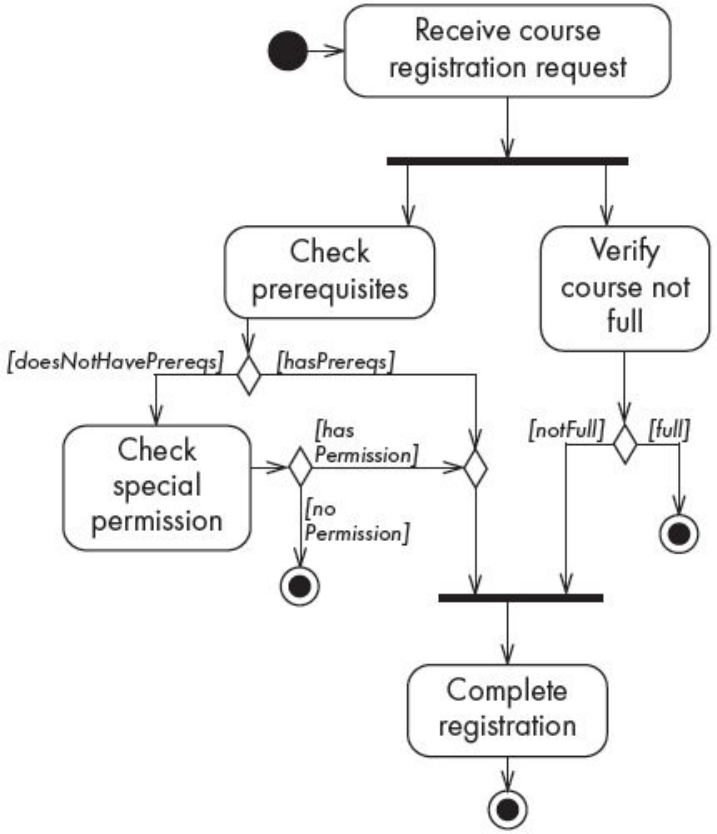
Activity Diagrams

- An *activity diagram* is like ***a state diagram***.
  - Except, most transitions are caused by ***internal* events**, such as the completion of a computation.
- Is used to understand the **flow of work that an object or component performs**
- Graphical representations of workflows (performed by object or component) of **stepwise activities and actions with support for choice, iteration and concurrency**.
- One of the strengths of activity diagrams is the representation of ***concurrent* activities**.

Activity Diagram Notation

- Activity diagram uses ***rounded rectangles*** to imply a specific system function
- ***Arrows*** to represent flow through the system.
- ***Decision diamonds*** to depict a branching decision (each arrow emanating from the diamond is labelled).
- ***Solid horizontal lines*** to indicate that parallel activities are occurring.

Activity diagram – an example



Activity diagram of the registration process

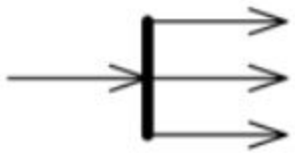
Representing concurrency

- Shown using **forks, joins and rendezvous.**

  - A ***fork*** has one incoming transition and multiple outgoing transitions.
  - A ***join*** has multiple incoming transitions and one outgoing transition.
  - A ***rendezvous*** has multiple incoming and multiple outgoing transitions.

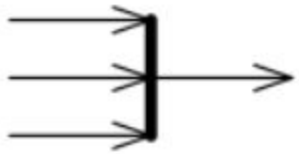
Representing concurrency

- A ***fork*** has one incoming transition and multiple outgoing transitions.
  - The execution splits into multiple concurrent threads.



Representing concurrency

- A ***join*** has multiple incoming transitions and one outgoing transition.
  - The **outgoing transition will be taken when all incoming transitions have occurred**.
  - The incoming transitions must be triggered in separate threads.
  - If **one incoming transition occurs, a wait condition** occurs at the join until the other transitions occur.



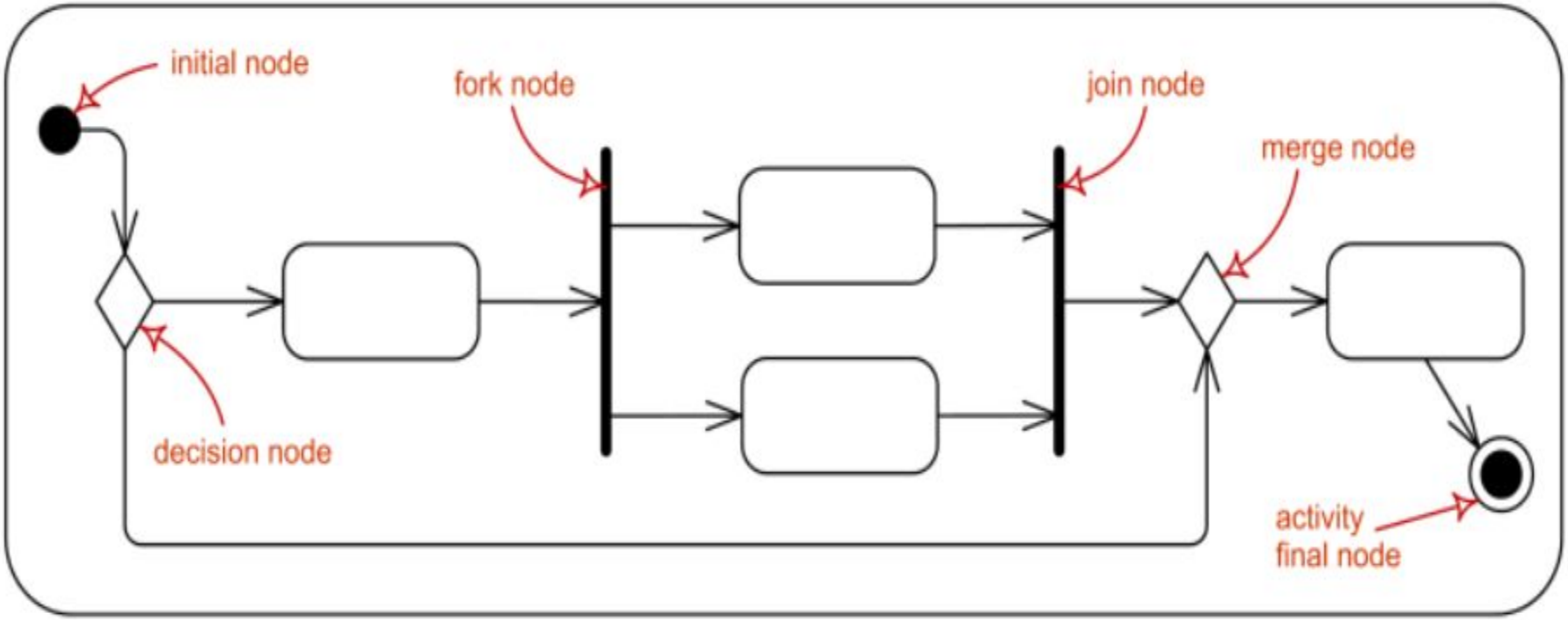
Representing concurrency

- A ***rendezvous*** has multiple incoming and multiple outgoing transitions.
  - Once all the incoming transitions occur all the outgoing transitions may occur.

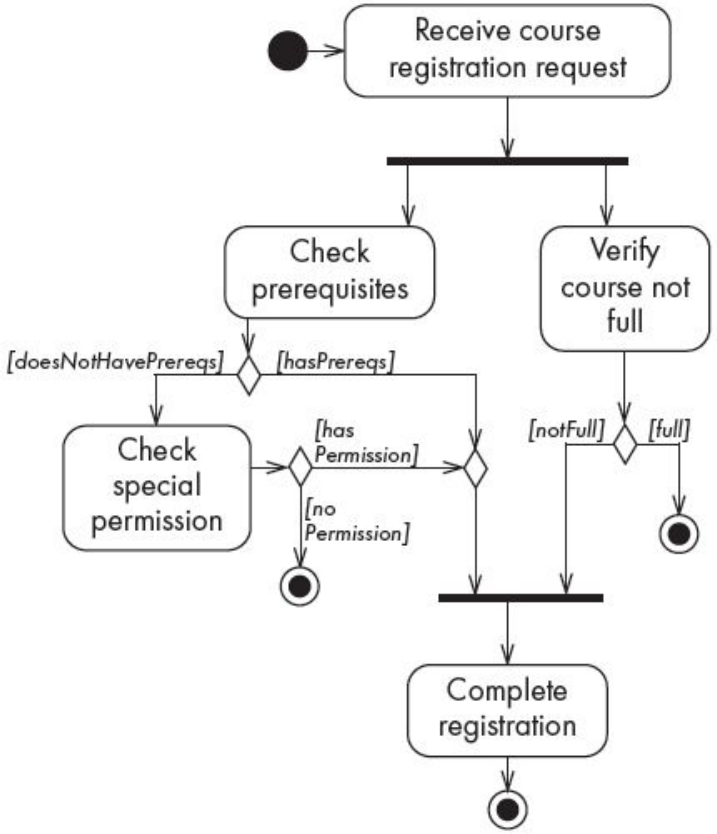
Decision nodes and merge nodes

- An activity diagram has two types of nodes for **branching within a single thread**. These are represented as small ***diamonds***:
  - ***Decision node***
    - *has one incoming transition and multiple outgoing transitions* each with a Boolean guard in square brackets. **Exactly one of the outgoing transitions will be taken**.
  - ***Merge node***
    - *has two incoming transitions and one outgoing transition. It is* used to bring together paths that had been split by decision nodes.

Activity Diagram Notation …..



Activity diagrams – an example

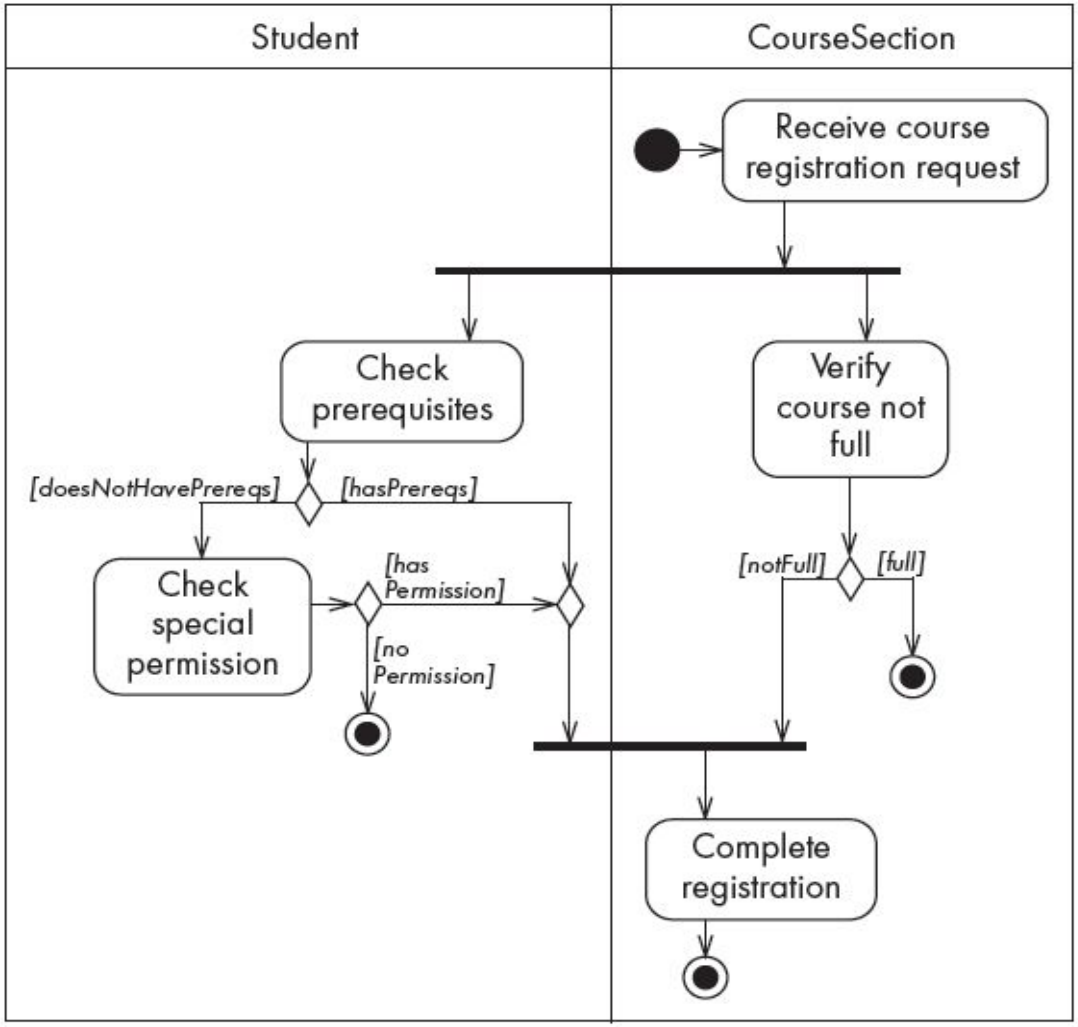


Activity diagram of the registration process

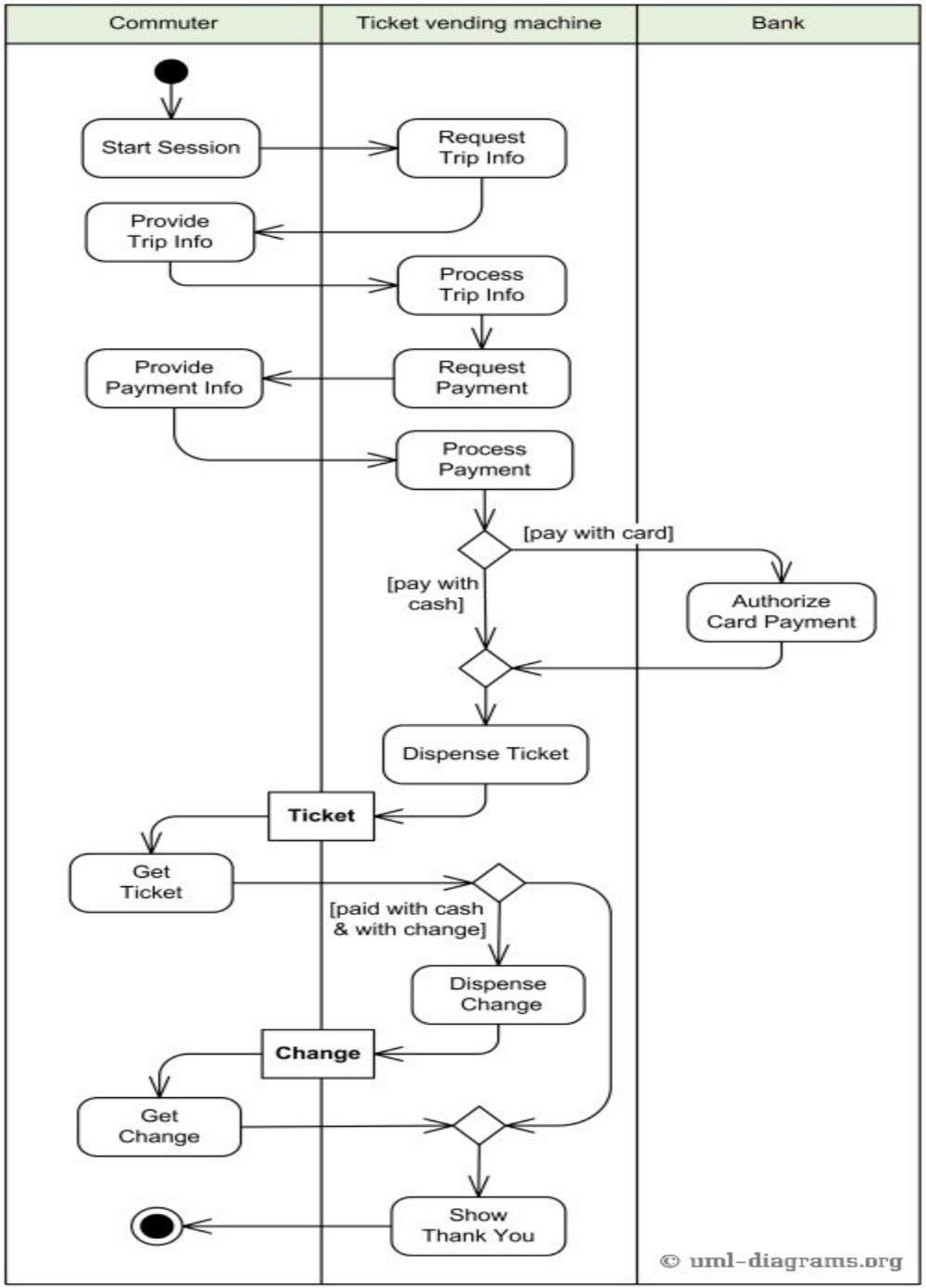
Swimlanes

- Activity diagrams are **most often associated with several classes**.
- The partition of activities among the existing classes can be explicitly shown in an activity diagram by the introduction of ***swimlanes.***
  - *Allows* you to represent the flow of activities described by the use case
  - indicate which actor or analysis class has responsibility for the action described by an activity rectangle.

- Responsibilities are represented as parallel segments that divide the diagram vertically, like the lanes in a swimming pool.



Activity diagram with swimlanes


Commuter
Ticket vending machine
Bank
Start Session
Request Trip Info
Provide Trip Info
Process Trip Info
Provide Payment Info
Request Payment
Process Payment
[pay with card]
[pay with cash]
Authorize Card Payment
Dispense Ticket
Ticket
Get Ticket
[paid with cash & with change]
Dispense Change
Change
Get Change
Show Thank You
© uml-diagrams.org

Surprise Test – 4 Marks

- Consider a bulb with a push down switch. The bulb initially remains off.

  When the switch is pushed down, the bulb is on. Again when the switch is pushed up, the bulb turns off. The lifecycle of the bulb continues in this way until it gets damaged.

Draw State Diagram – Give meaningful names for the states and the events. **Show the actions in each of the states.**