UML Modeling II

Instructor: Yongjie Zheng
February 28, 2020

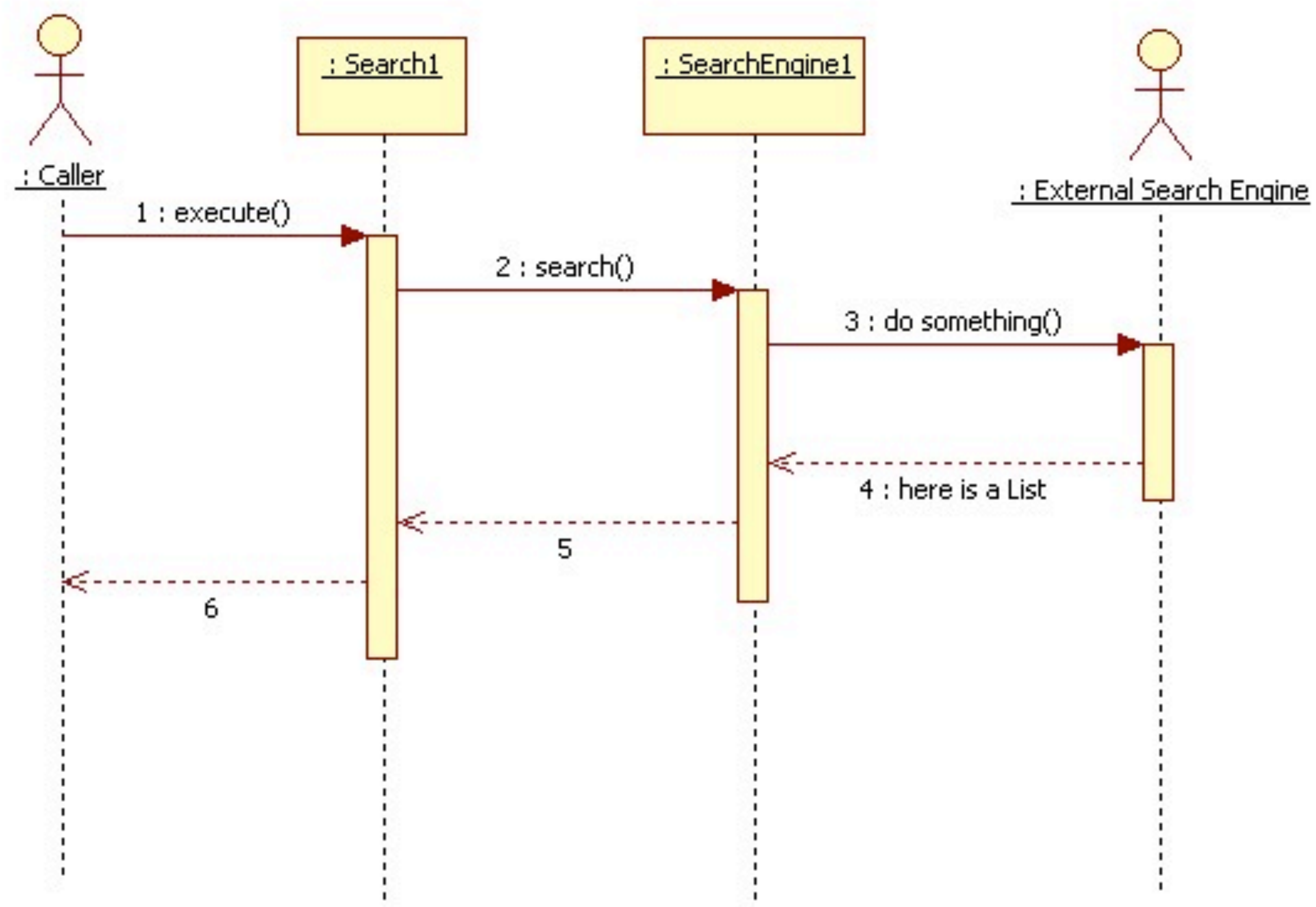
CS 441: Software Engineering

	Structure	Behavior
Requirements		Use Case Diagrams
Design	Class Diagrams Package Diagrams	Sequence Diagrams State Diagrams Activity Diagrams

Sequence Diagram

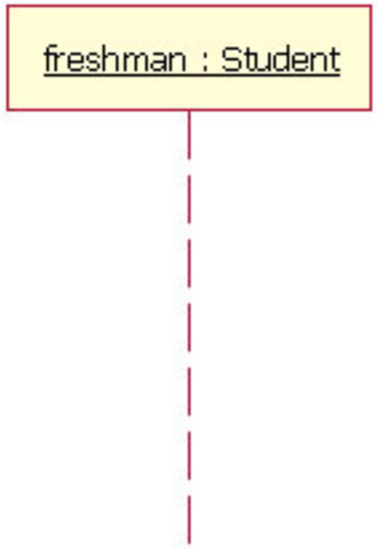
- A sequence diagram captures the behavior of a single scenario.
- The diagram shows a number of participating objects and the messages that are passed between these objects within the scenario.
- The diagram conveys information along the horizontal and vertical dimensions:
  - The vertical dimension shows, top down, the time sequence of messages/calls as they occur.
  - The horizontal dimension shows, left to right, the object instances that the messages are sent to.

An Example of Sequence Diagram



Sequence Diagram: Object

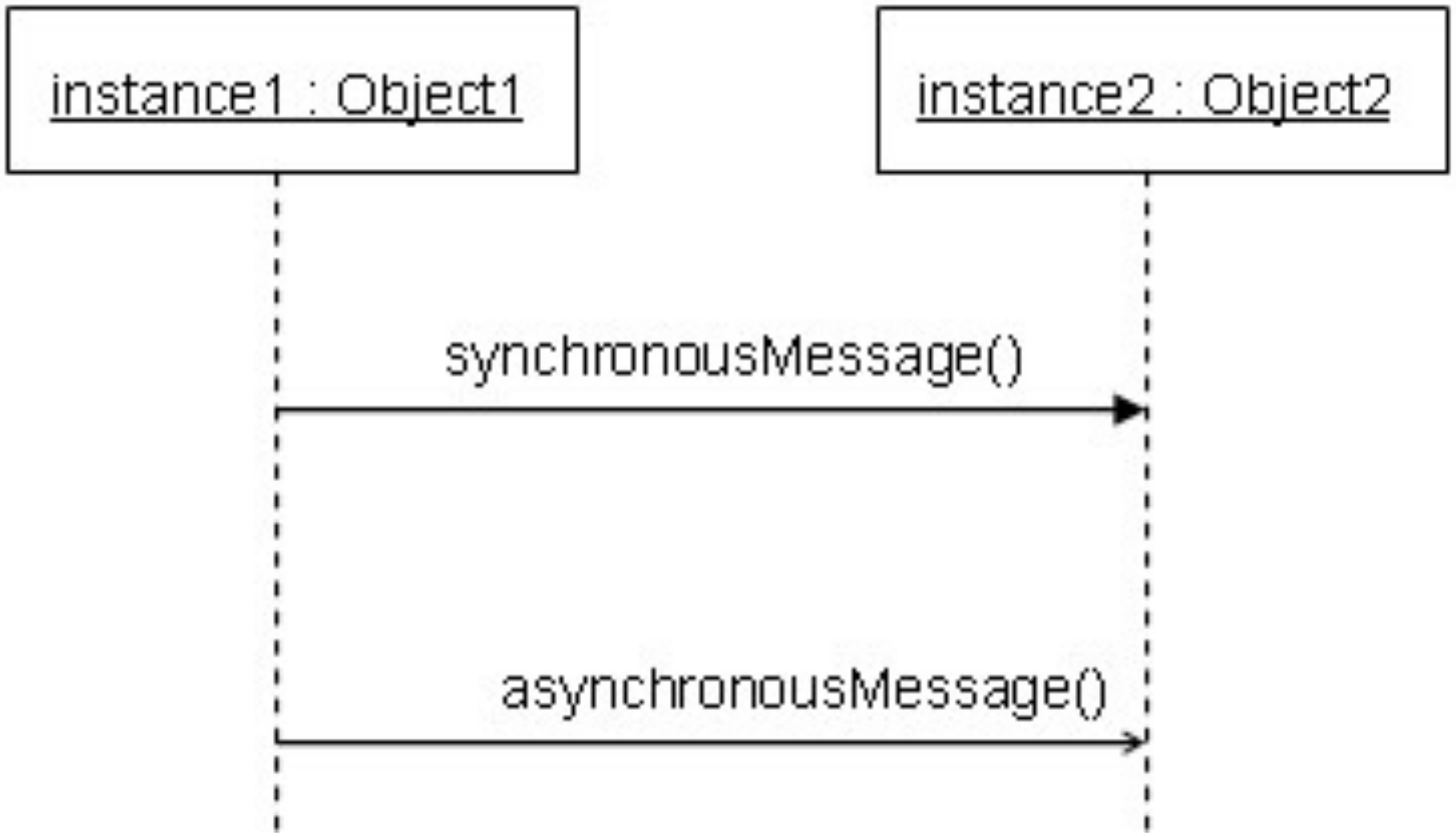
- Shown as a box at the top of a dashed vertical line.
  - Each box represents an instance of a class, or an object. (UML 1.x only)
- The vertical line is called the object's lifeline.
- An activation box is optional, and shows when an object is active.



Sequence Diagram: Message

- Analogous to method calls in a program
  - Can have parameters
- Labeled with the message name and parameters (optional).
- Synchronous messages
  - Calling object waits for call to complete
  - Indicated by a filled arrowhead (UML 2.x)
- Asynchronous messages
  - Calling object does not wait for call to complete
  - Indicated by a stick arrowhead (UML 2.x)

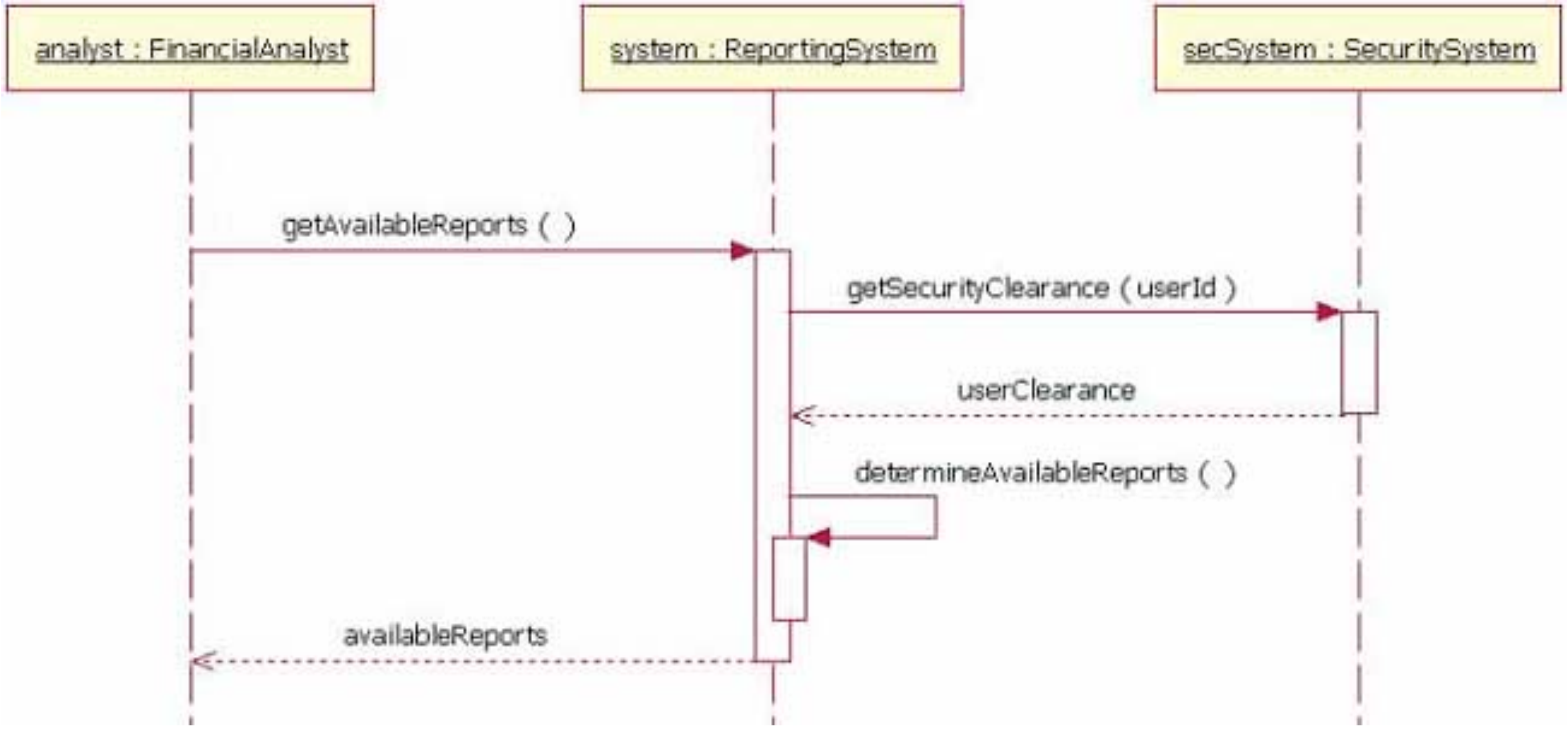
An Example


instance1 : Object1
instance2 : Object2
synchronousMessage()
asynchronousMessage()

Sequence Diagram: Message

- Special messages
  - new — shown by the position of an object
  - delete — shown through big X
- Return messages
  - Optional
  - Drawn as a dotted line with an open arrowhead back to the originating lifeline, and above this dotted line you place the return value from the operation.
- Self-calls

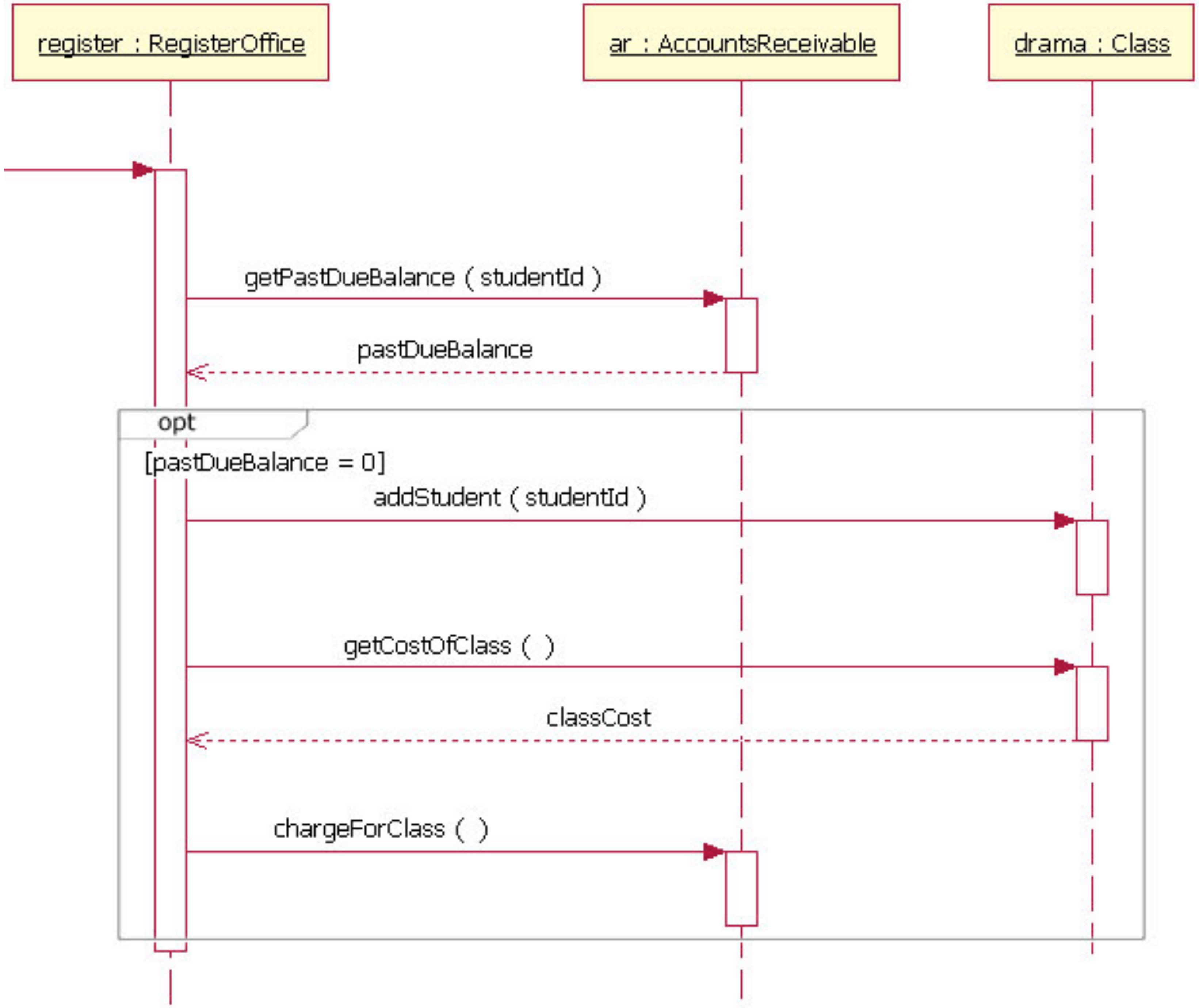
Another Example


analyst : FinancialAnalyst
system : ReportingSystem
secSystem : SecuritySystem
getAvailableReports ( )
getSecurityClearance (userId)
userClearance
determineAvailableReports ( )
availableReports

Sequence Diagram: Frame

- Frames are regions or fragments of the diagrams to support conditional and looping constructs. They have a frame operator and a guard.
- Frame Operator
  - loop – loop fragment while guard is true.
  - opt – execute fragment while guard is true
  - alt – alternative fragment for mutual exclusion conditional logic expressed in the guards
  - par – parallel fragments that execute in parallel
  - region – critical region within which only one thread can run
- Guard
  - [conditional clause]

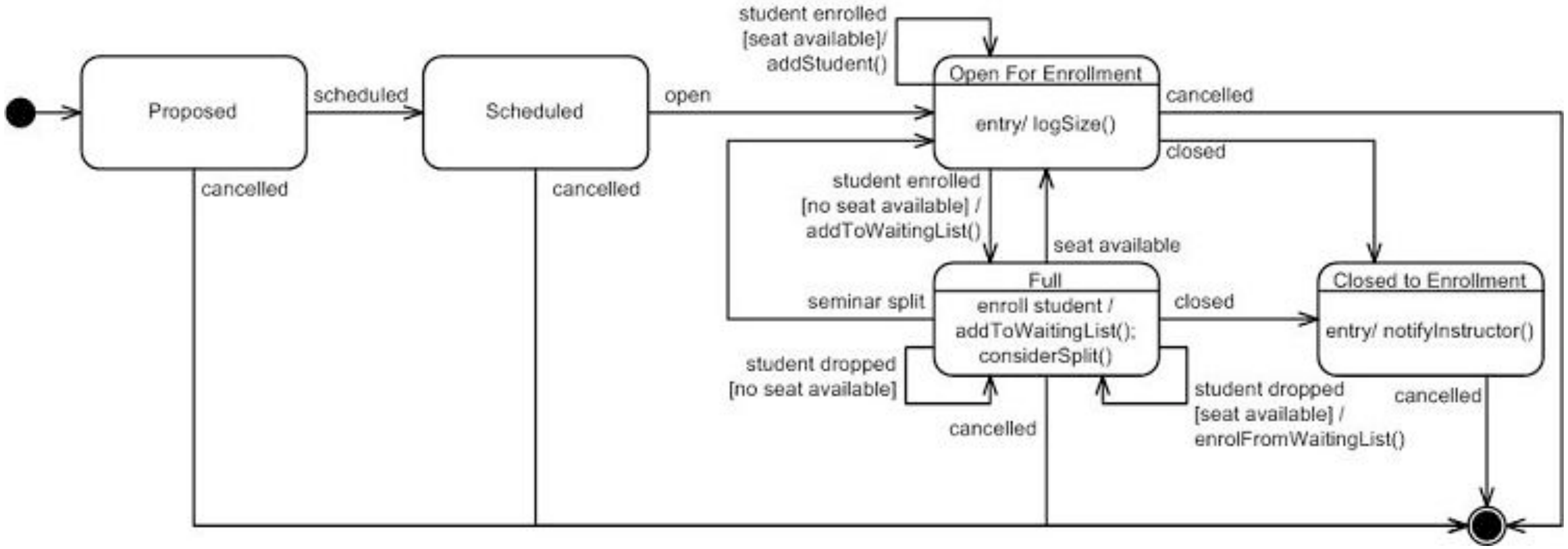
An Example of Frames



State Diagram

- A state diagram describes all of the possible states that a particular object can get into and how the object's state changes as a result of events that reach the object.
- State diagrams are good at describing the behavior of an object across several use cases.

An Example of State Diagram



State Diagram: State

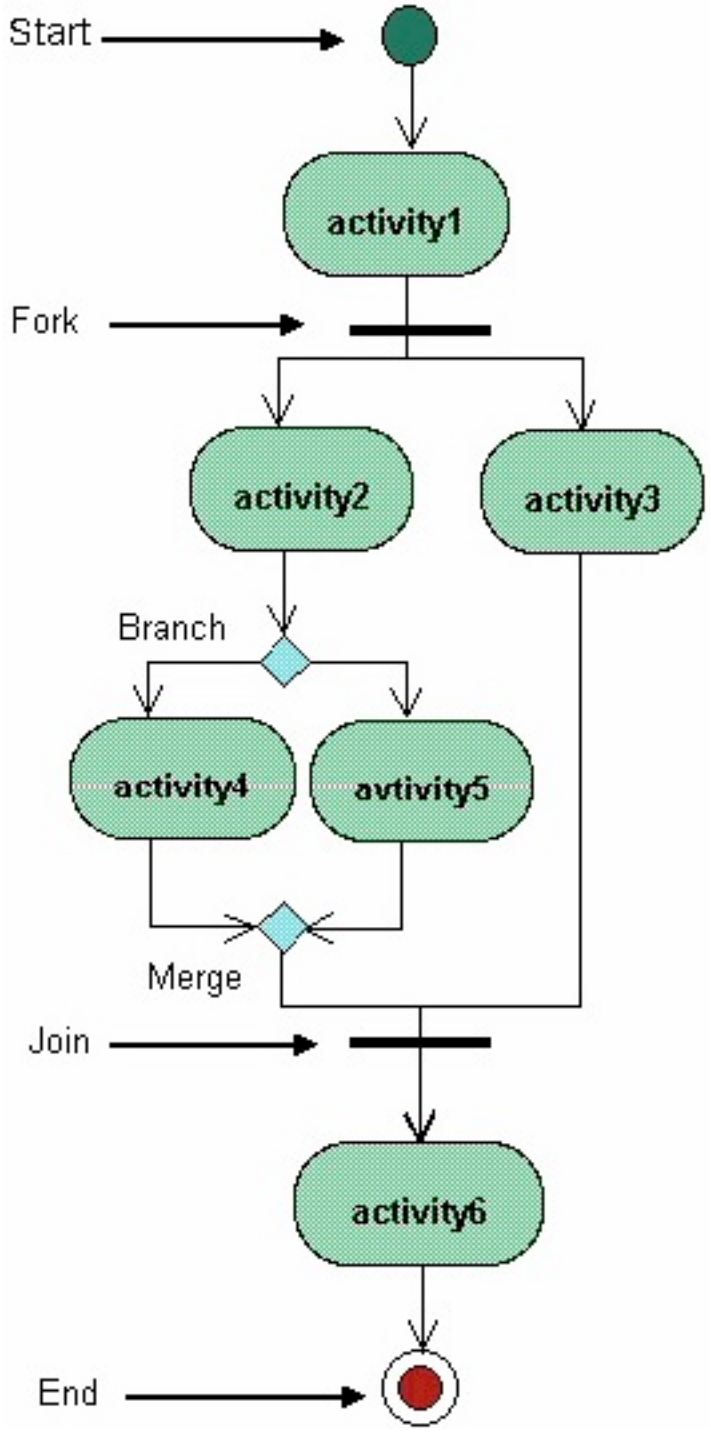
- Represented by rounded rectangles.
- An object starts in an initial state, represented by the closed circle, and can end up in a final state, represented by the bordered circle.
- If a state responds to an event that does not cause a transition, you can show this by putting text in the form of event/activity in the state box (i.e. internal activity).
  - Entry/activity: method to be invoked when the object enters the state
  - Exit/activity: method to be invoked when the object exits the state
  - do/activity: method to be run when the object is in the state (i.e. ongoing activity).

State Diagram: Transition

- Line with line arrowhead
- Represents movement from one state to another
- The format of a transition label is: *Event [Guard] / Action*.
    - Event (optional) causes the transition
    - Guard (optional) must be true for the transition to be triggered
    - Action (optional) is the invocation of a method
- An unlabeled transition occurs immediately or as soon as any activity associated with the given state is completed.
- <u>Only one transition can be taken out of a given state under any condition</u>.

Activity Diagram

- An activity diagram describes the sequencing of activities, with support for both conditional and parallel behavior.
- An activity diagram does not convey which class is responsible for each activity.



Activity Diagram

- Conditional behavior
  - Branch
    - When the incoming transition is triggered, only one of the outgoing transitions can be taken.
  - Merge
    - Marks the end of conditional behavior started by a branch.
- Parallel behavior
  - Fork
    - When the incoming transition is triggered, all of the outgoing transitions are taken in parallel.
  - Join
    - The outgoing transition is taken only when all the states on the incoming incoming transitions have completed their activities.

To summarize

- A **Use Case Diagram** presents an external view of the system.
- A **Sequence Diagram** presents the behavior of several objects within a single scenario.
- A **State Diagram** presents the behavior of an object across several use cases.
- An **Activity Diagram** presents the parallel behavior and concurrent programs.