Lecture 18: Activity Diagrams

Kenneth M. Anderson
Object-Oriented Analysis and Design
CSCI 6448 - Spring Semester, 2001

Activity

- An OO system engages in a set of activities
  - Activities cause actions to occur
  - while activities themselves are non-atomic, they consist of a sequence of atomic computations
    - this means that an activity can be interrupted; but each of its computations (or individual actions) are atomic, and can't be interrupted

Activity Diagrams

- Activity diagrams show the flow of control between activities
  - They can model the sequential and concurrent steps in a computational process
  - They can also model the flow of an object as it moves from state tot state at different points in the activity

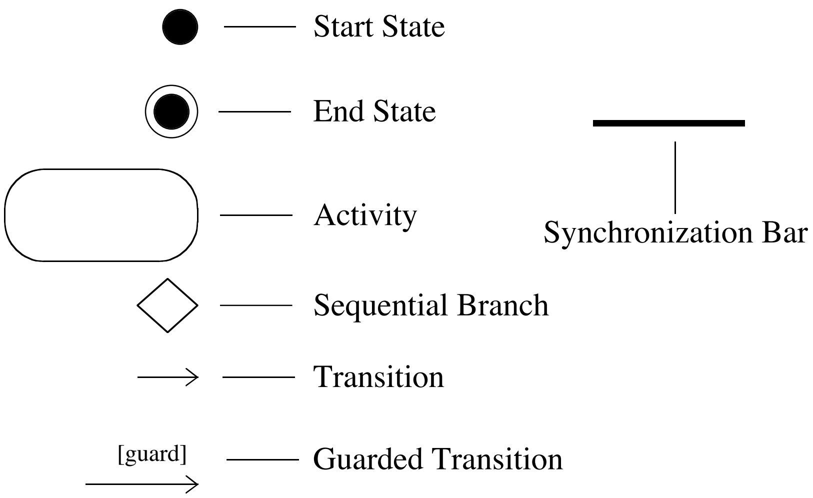
Activity Diagrams

- Specify the flow of a particular activity
  - An activity is decomposed into steps
  - Semantics that govern the flow of the activity can be included
- Activity Diagrams are hierarchical
  - I.E. a step in one diagram can be associated with another diagram that describes its sub-steps

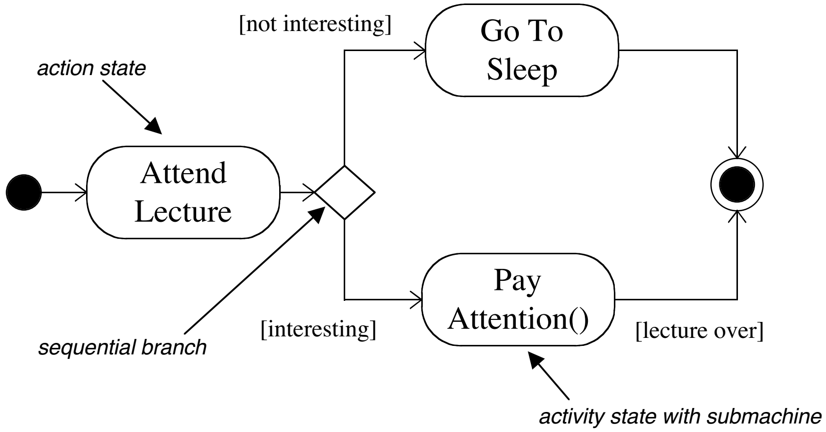
Activity Diagrams/State Diagrams

- Important Note: Activity diagrams are a type of state machine; so any notation that we covered in lecture 17 can be applied to activity diagrams
- UML User Guide, page 260
  - "An activity diagram is a special case of a state diagram in which all (or at least most) of the states are action states and in which all (or at least most) of the transitions are triggered by completion of the actions in the source states."

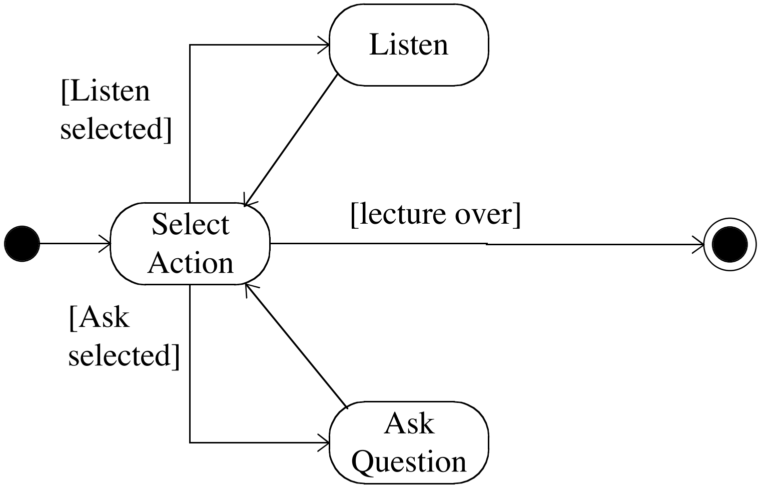
Notation


Start State
End State
Activity
Synchronization Bar
Sequential Branch
Transition
[guard]
Guarded Transition

Brief Example


[not interesting]
Go To Sleep
action state
Attend Lecture
sequential branch
[interesting]
Pay Attention()
[lecture over]
activity state with submachine

Decompose "Pay Attention"


Listen
[Listen selected]
Select Action
[lecture over]
[Ask selected]
Ask Question

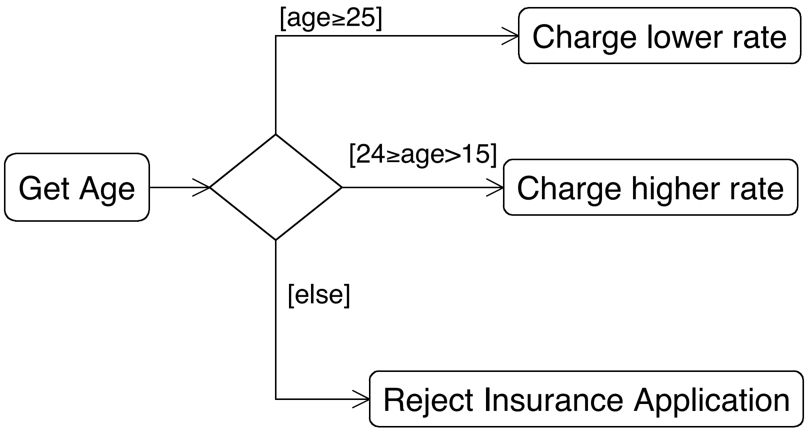
Action States / Activity States

- An action state in an activity does something
  - call an operation
  - send a signal
  - create or destroy objects
  - even evaluate expressions to assign values to attributes
- See UML User Guide, page 261 for examples
- Action states cannot be decomposed and cannot be interrupted
- Activity states, on the other hand, represent more complex actions, they can be interrupted, and can be decomposed (into other activity diagrams)

Sequential Branching

- A sequential branch is represented as a diamond
  - It may have one incoming transition and two or more outgoing transitions
  - Guards are associated with each transition; the guards are evaluated upon entering the branch, and the one which evaluates to true is then taken
    - Guards should therefore not overlap but cover all possibilities
    - You can label one transition with [else]; if all other guards evaluate to false, then the [else] transition is taken

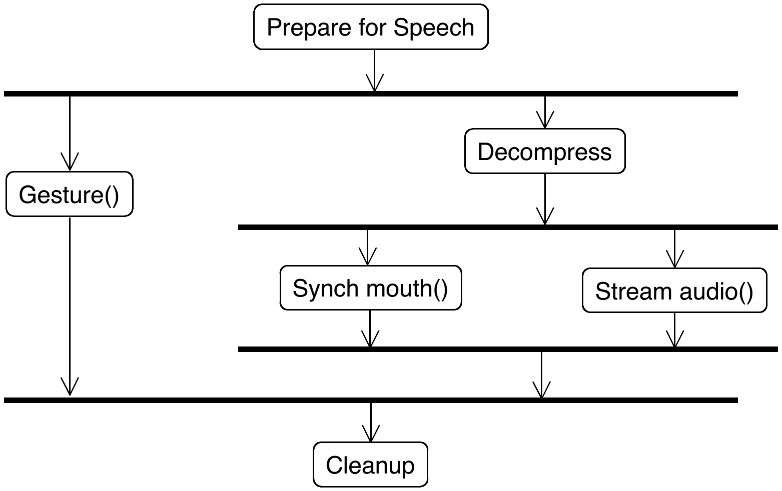
Example



Forking and Joining

- Concurrent activities within an activity diagram are modeled with the use of synchronization bars
- Synchronization bars are drawn as a thick horizontal or vertical line
- Joins and Forks should balance
  - the number of flows that leave a fork should equal the number of flows entering the corresponding join

Example



Swimlanes

- The activities of an activity diagram may be performed by different groups
  - You can indicate this by using swimlanes
- Each swimlane has a unique name and typically represents some real-world entity
- If swimlanes are used, each activity can belong to one and only one swimlane
- See example on page 266 of the UML User Guide

Object Flow

- Objects may be involved in the flow of control associated with an activity
  - For instance, an activity may create an object and pass it to some other activity
- These associations can be shown in activity diagrams by placing objects in the diagram and linking them to specific activities using a dependency link
- See example on page 267 and 270 of the UML User Guide