Activity Diagrams

Object-Oriented Analysis and Design

CSCI 6448 - Fall 1998

Kenneth M. Anderson

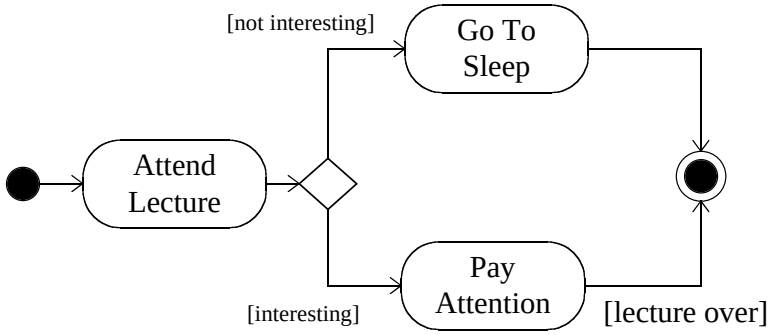
Goals of this Lecture

- Explain Activity Diagrams
  - Present background
  - Explain notation
  - Discuss when to use
- Present Simple Example
  - More complex examples next lecture

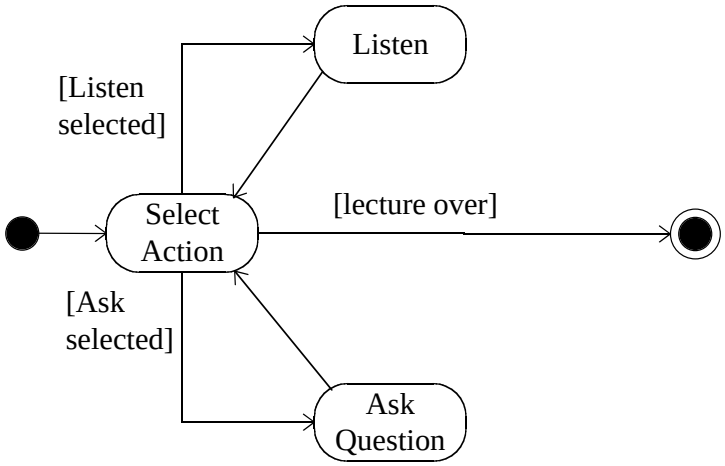
Activity Diagrams

- 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

Brief Example



Decompose "Pay Attention"



Background

- Activity Diagrams did not originate from the work of the "three amigos"
- Jim Odell's event diagrams are most closely related
  - Events represent changes of state
  - We will cover events from Odell's perspective next week

Also related

- Activity Diagrams are also related to state diagrams and Petri-Nets
- UML Website states
  - "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."

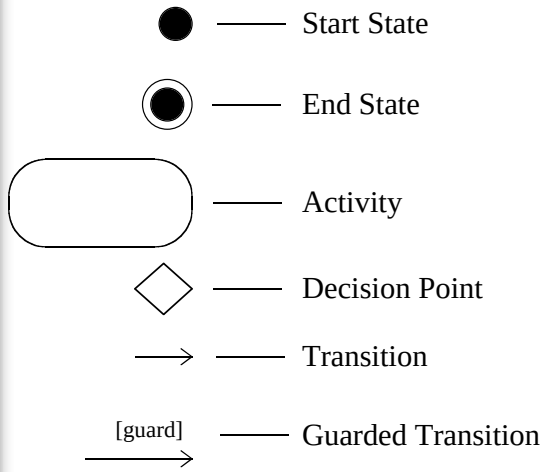
Implications

- Anything that can be used in state diagrams can be used in activity diagrams
  - However, by sticking with the constraints (action-only states, simple triggers) activity diagrams more easily convey high-level workflow
- We cover state diagrams next week

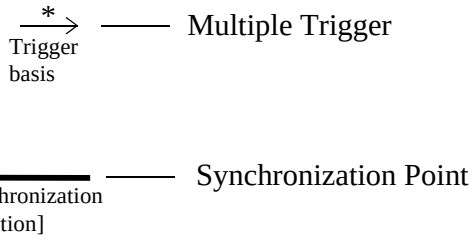
Petri-Nets

- Used to model and analyze concurrent software
- Notation consists of places, transitions, and tokens
- Semantic rules allow Petri-Nets to be “executed” in order to simulate concurrent behavior

Notation



Additional Notation



Multiple Trigger

- Invokes a particular activity multiple times
- The trigger basis bounds the number of times the activity is invoked
- The activities are assumed to occur in parallel; stereotypes could be used to express alternate semantics
  - I.e. <<iterate>>

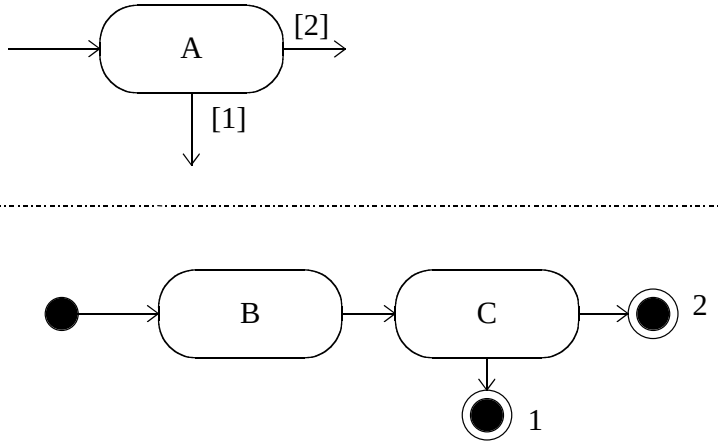
Synchronization Condition

- Each time a trigger arrives at a synchronization bar, the bar's condition is evaluated
- Once it is true, the outgoing activities are initiated
- If no condition is specified, all incoming triggers must have occurred

Ending an Activity Diagram

- An end state does not have to be indicated
  - In such cases, the diagram is finished when no more activities can be triggered
- A sub-activity diagram much have end states that correspond to the parent activities outgoing transitions

Example



Reminder

- Activity Diagrams are a special case of State Diagrams
  - As a result
    - The notation of state diagrams can also be used in activity diagrams
    - We will cover state diagram notation next week

When to Use Activity Diagrams

- Analyzing a use case
  - Capture the flow of the use case's required actions
- Understanding workflow across multiple use cases
  - Clearly represent parallel workflows and synchronization conditions
- Understanding multi-threaded code