

State Modeling

COMP 3700.002
Software Modeling and Design

Shehenaz Shaik

OO Models

- Class Model

- Static structure of objects and relationships
- Class diagram

- State Model

- Changes over time or on events
- State diagram

- Interaction Model

- Interaction among objects
- Use case diagram
- Sequence diagram
- Activity diagram

State Model

- What it is?
 - State model describes the sequences of operations that occur in response to external stimuli.
- What it is not?
 - What the operations do?
 - How they are implemented?
 - What they operate on?

State Model (Contd.)

- State diagram
 - Graphical representation of relationship between states and events
- Multiple state diagrams
 - One for each class with temporal behavior
- States
 - Values of objects
- Events
 - External stimuli

Events

- An occurrence at a point in time
- Appear as
 - Verbs in past tense
 - Onset of some condition
- Causally related, or unrelated (concurrent)
- Kinds of events
 - Signal event
 - Change event
 - Time event

1. Signal event

- Signal
 - An explicit one-way transmission of information from one object to another
 - Message between objects
- Signal event
 - Event of sending / receiving a signal
 - An occurrence in time

Signal class

- Common structure and behavior
- UML Notation: << >>
 - Signal class name
 - Signal attributes



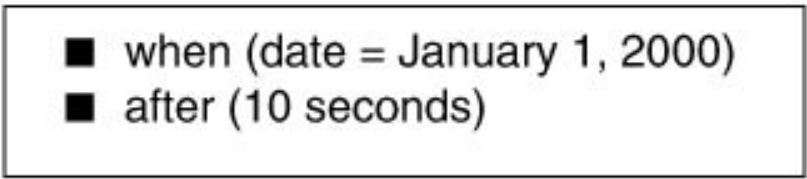
2. Change event

- Event caused by satisfaction of a Boolean expression
 - Expression is continually tested
- UML Notation: when (exp)

- when (room temperature < heating set point)
- when (room temperature > cooling set point)
- when (battery power < lower limit)
- when (tire pressure < minimum pressure)

3. Time event

- Event caused by
 - Occurrence of absolute time
 - Elapse of a time interval
- UML Notation:
 - when (time exp)
 - after (time duration)



■ when (date = January 1, 2000)
■ after (10 seconds)

The image shows a rectangular box with a thin black border containing two lines of text. Each line starts with a small black square followed by the text 'when (date = January 1, 2000)' and 'after (10 seconds)' respectively.

Event types

- Kinds of events
 - Signal event
 - Change event
 - Time event

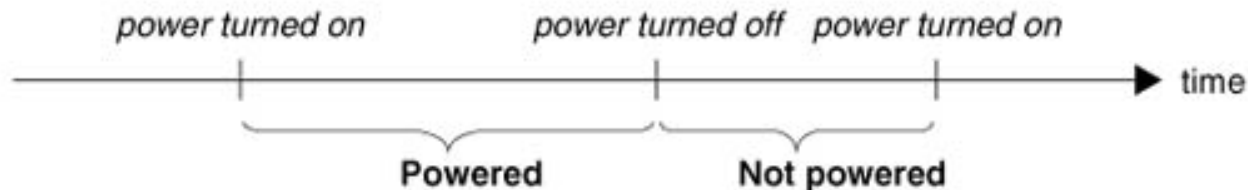
State

- Abstraction of values and links of an object
- Sets of values and links grouped together
 - All combinations of values and links with same response to events
→ Same state
- Attributes having no impact on sequence of control
 - Regard them as simple parameter values within a state
- Appear as
 - Verbs with 'ing' suffix
 - Duration of some condition
- UML Notation: Rounded box



Event Vs. State

- Event
 - Point in time
- State
 - Interval of time
- Both depend on level of abstraction



Characterization of a state

State: *AlarmRinging*

Description: alarm on watch is ringing to indicate target time

Event sequence that produces the state:

setAlarm (targetTime)

any sequence not including *clearAlarm*

when (*currentTime = targetTime*)

Condition that characterizes the state:

alarm = on, alarm set to *targetTime*, $targetTime \leq currentTime \leq targetTime + 20$ seconds, and no button has been pushed since *targetTime*

Events accepted in the state:

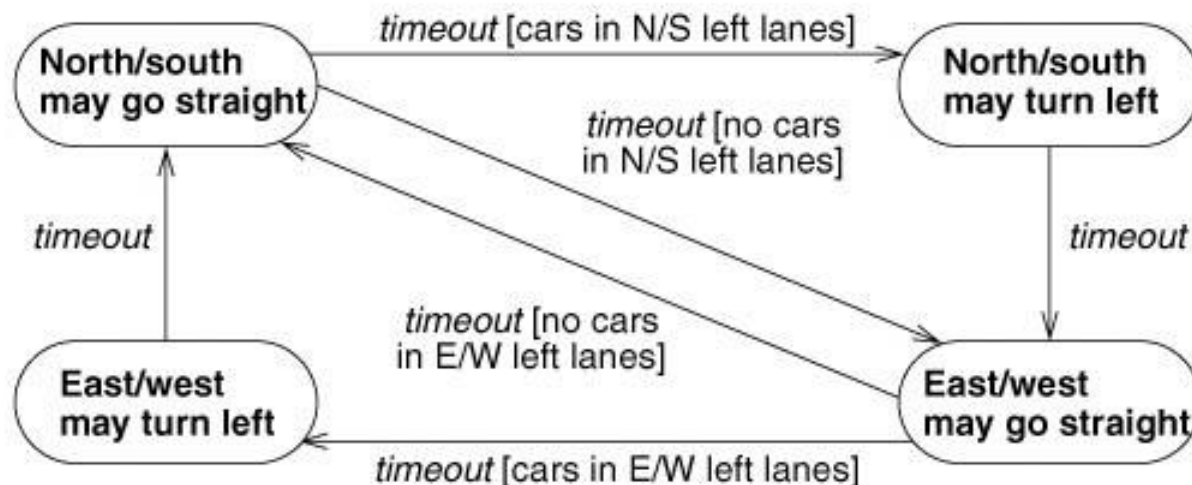
event	response	next state
when (<i>currentTime = targetTime + 20</i>)	<i>resetAlarm</i>	<i>normal</i>
<i>buttonPushed</i> (any button)	<i>resetAlarm</i>	<i>normal</i>

Transition

- Instantaneous change from one state to another
- Transition '*fires*' when its event occurs
- Next state depends on
 - Current state
 - Event received
- An event may cause multiple objects to transition concurrently
- UML Notation:
 - Line from origin state to target state
 - Arrowhead points to target state

Guard condition

- Boolean expression that must be true for a transition to occur
- Checked only once when event occurs
 - If true, transition fires
 - Unlike change event, which is checked continuously
- UML Notation: [] next to transition label



State Diagram

- Directed graph
 - Nodes: states
 - Arcs: Transitions between states
- Specifies state sequences caused by event sequences
- All objects in a class execute the state diagram for that class
 - Models their common behavior

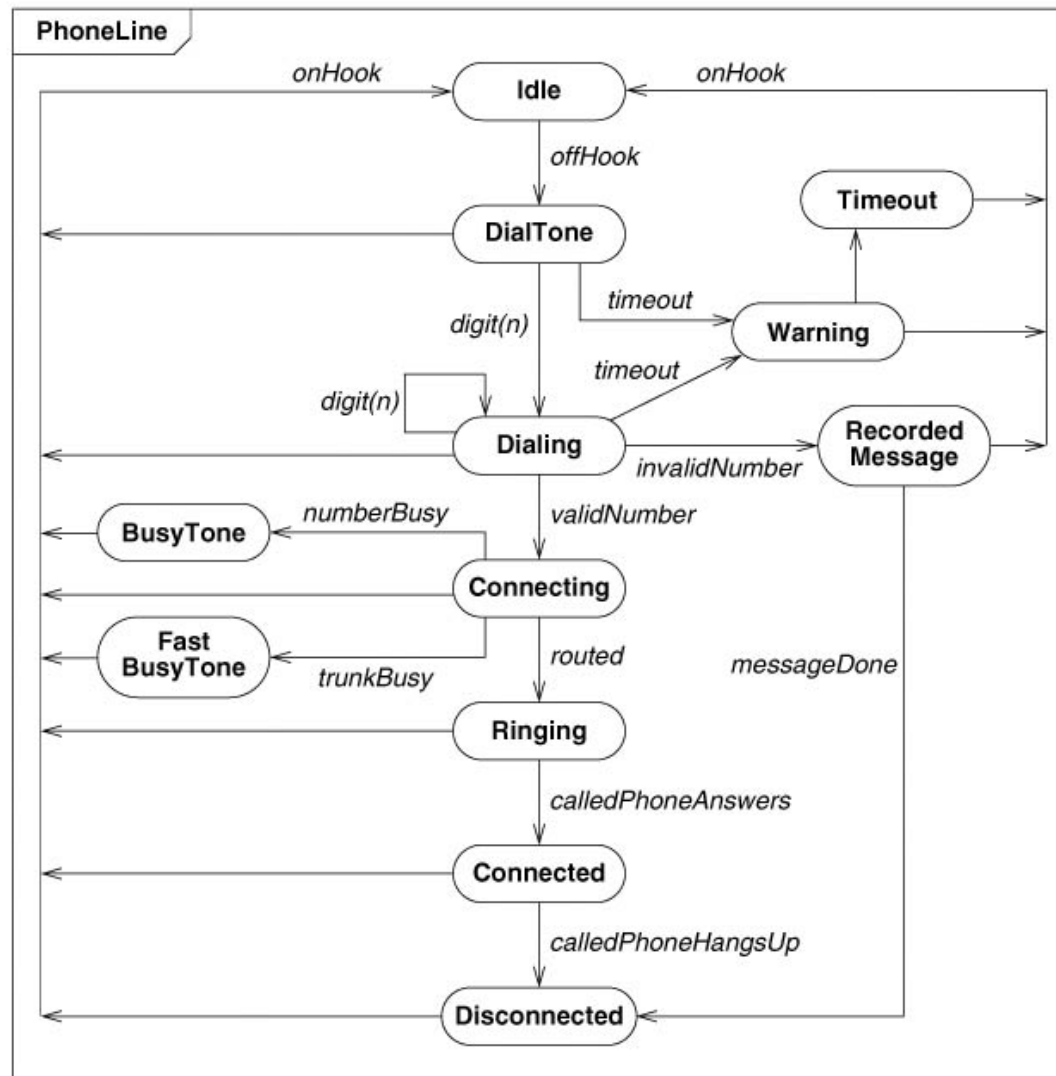
State Diagram (Contd.)

- State model
 - Multiple state diagrams
 - One per class with significant temporal behavior
 - More than one state
 - Single state with multiple responses to events
 - Stimulus / Response table may suffice
 - State diagrams must match on interfaces:
 - Events
 - Guard conditions

State Diagram (Contd.)

- UML Notation:
 - Rectangle with its name in a small pentagonal tag in upper left corner
 - States and transitions lie within the rectangle
- Two types
 - Continuous loops
 - One-shot life cycles

Continuous Loop State Diagram



One-shot State Diagram

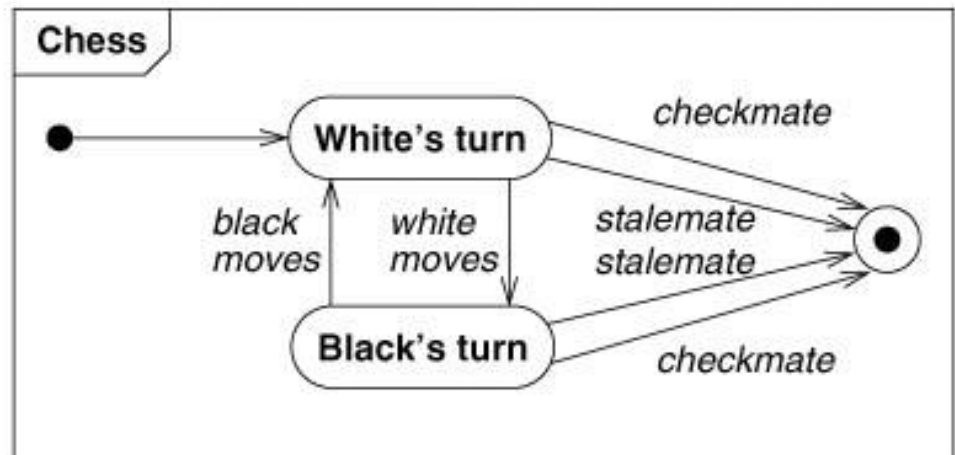
- Represents objects with finite lives

- Initial state

- Upon creation of object
- UML Notation: Solid circle

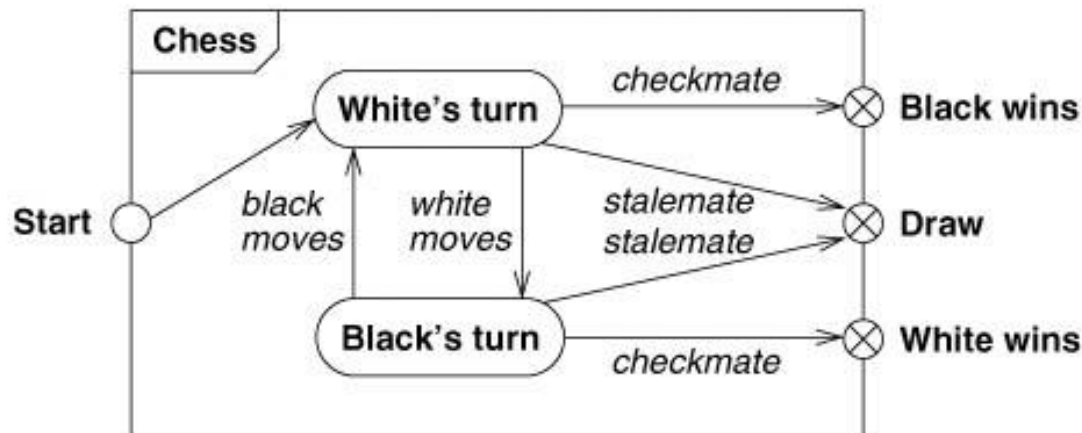
- Final state

- Implies destruction of object
- UML Notation: Bull's eye



One-shot State Diagram

- Alternate representation
 - Entry point
 - UML Notation: Hollow circle
 - Exit point
 - UML Notation: Circles enclosing an X
 - Entry/Exit points appear on state diagram's perimeter



Activity - Effect

- Effect

- Reference to behavior executed in response to an event

- Activity

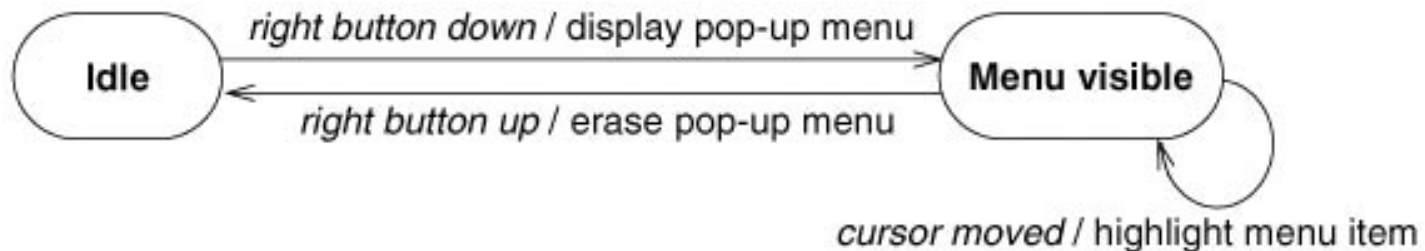
- Actual behavior invoked by any number of effects

- Activity performed:

- Upon transition to a different state
 - Upon entry to / exit from a state
 - Upon an event within a state

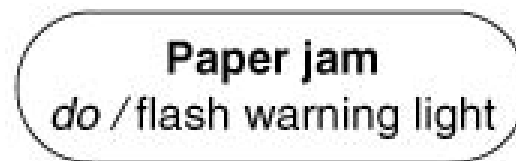
Activity – Effect (Contd.)

- UML Notation:
 - 'Event name / Activity name'



Do-Activity

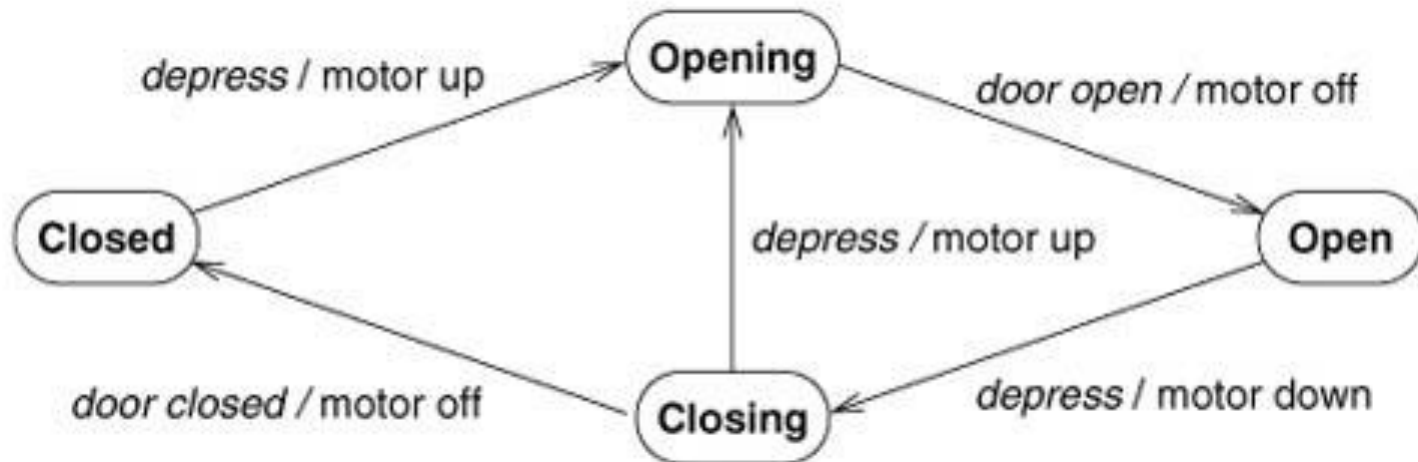
- Activity that continues for an extended time
- Can only occur within a state
- Cannot be attached to a transition
- Two types
 - Continuous operations, interrupted by an event
 - Sequential operations that terminate by themselves after an interval of time
- UML Notation:
 - 'do / Activity name'



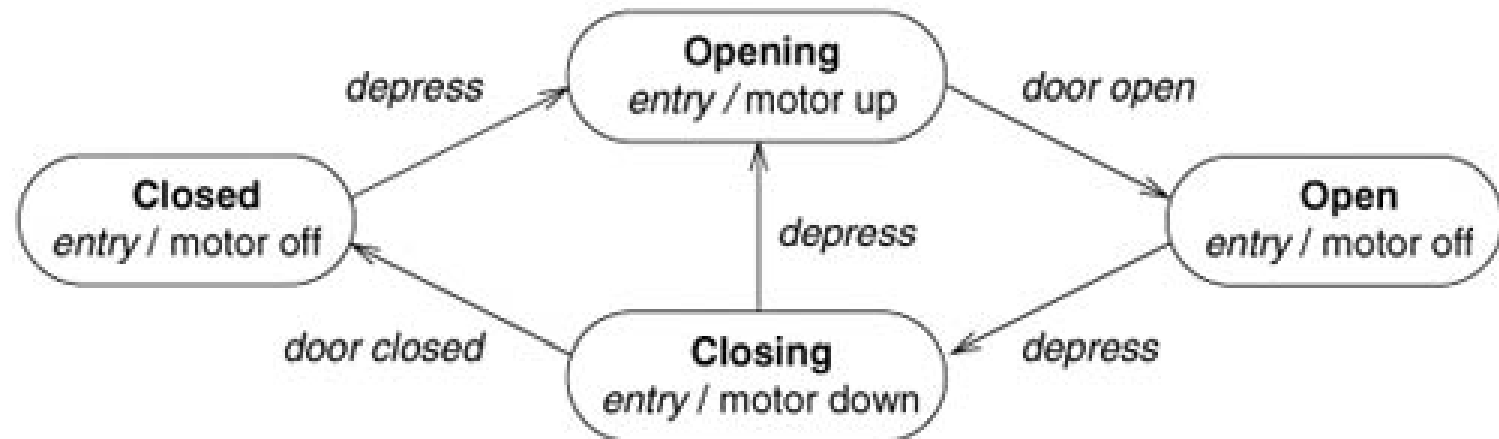
Entry / Exit Activities

- Bind activities to entry / exit from a state
 - Instead of showing activities on transitions
 - Entry/Exit activity → Attaching it to every incoming/outgoing transition respectively
 - If all transitions into a state perform same activity
 - Concise representation
- UML Notation:
 - 'entry / Activity name'
 - 'exit / Activity name'

Activities on transitions (e.g.)



Activities on entry to state (e.g.)



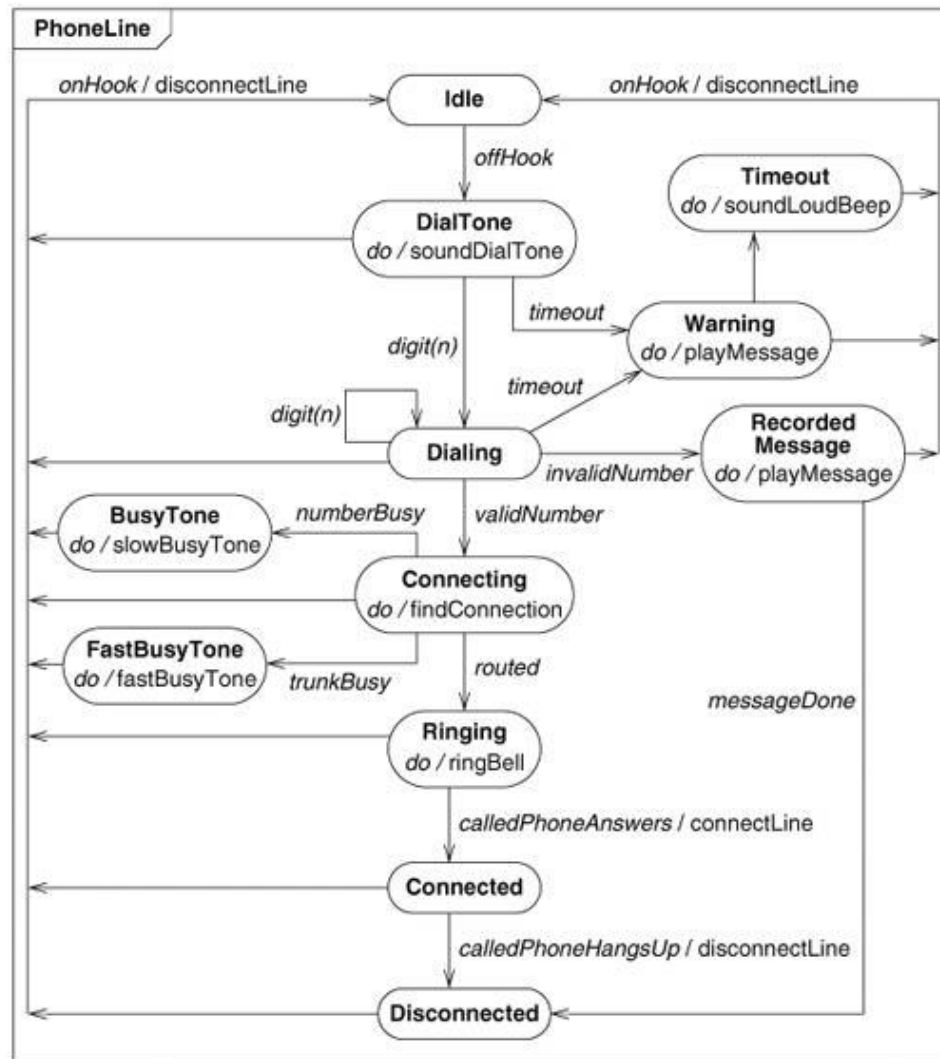
Execution order of Activities

- Execution order of activity types in a state
 - Activities on incoming transition
 - Entry activities
 - Do activities
 - Exit activities
 - Activities on outgoing transition

Completion Transition

- Automatic transition
 - Triggered by completion of activity in source state
 - UML Notation: Arrow without an event name
- Note:
 - Ensure at least one guard condition is satisfied
 - Otherwise, state remains 'stuck'
 - Use '*Else*' condition

State Diagram with Activities

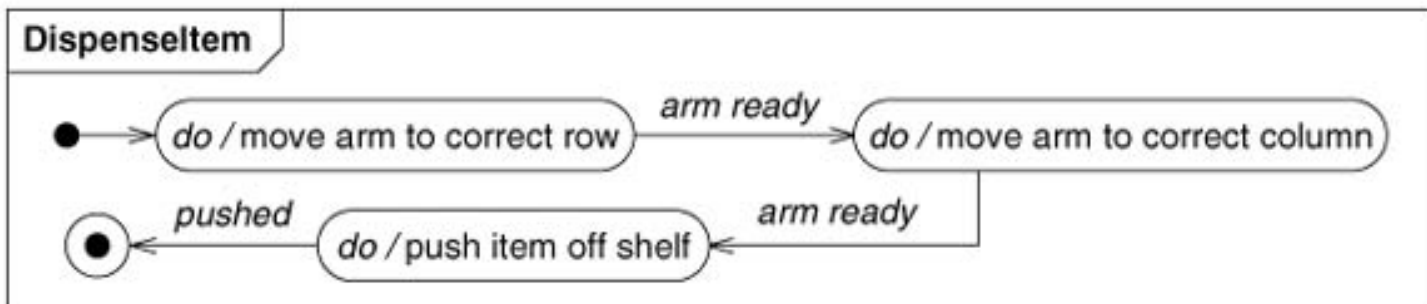


Flat State Diagrams: Issues

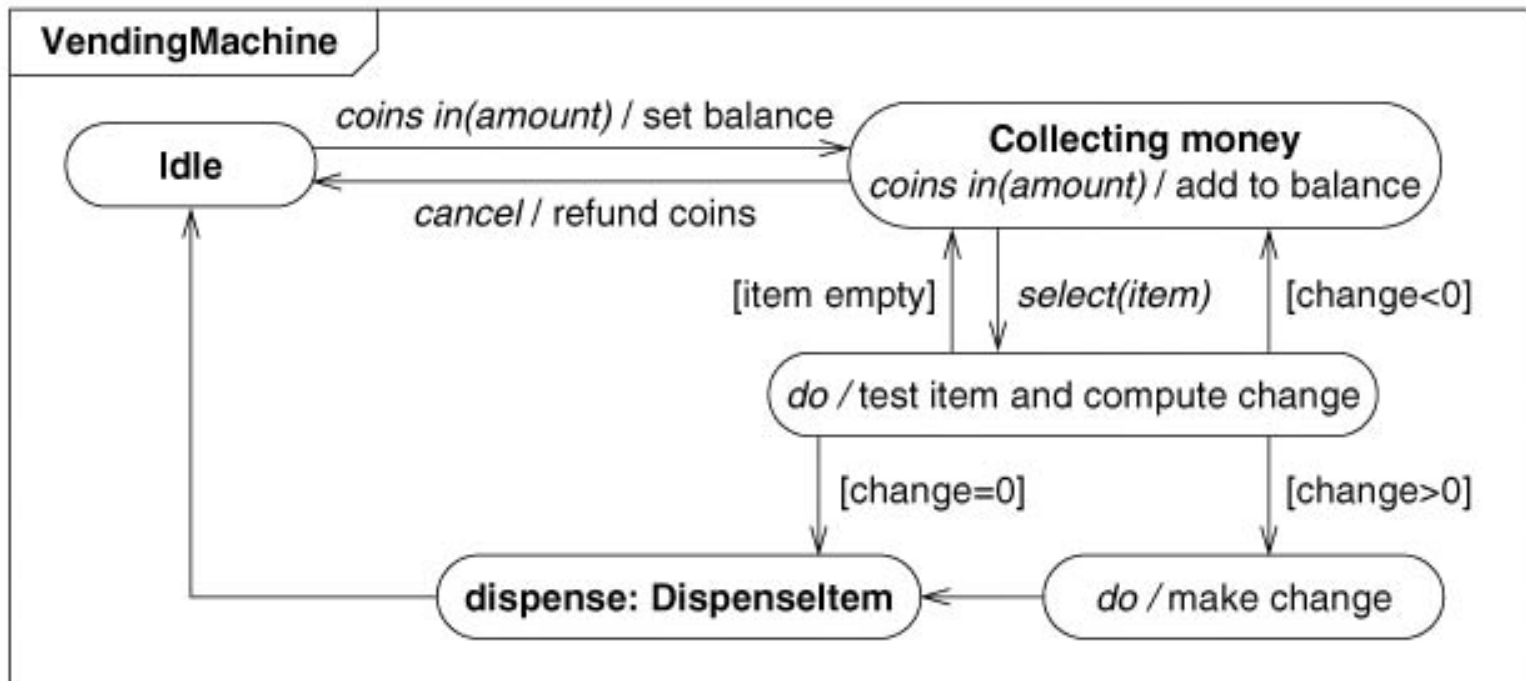
- Impractical for large problems
- Example
 - System with N Independent Boolean attributes that affect control
 - Single flat state diagram
 - 2^N states
 - N independent state diagram
 - 2n states only
 - 2 states per Boolean attribute (True / False)

Nested State Diagram

- Replacing a state with subdiagram
- Submachine
 - State diagram invoked as part of another state diagram
 - UML Notation
 - `local_state_name : submachine_name`



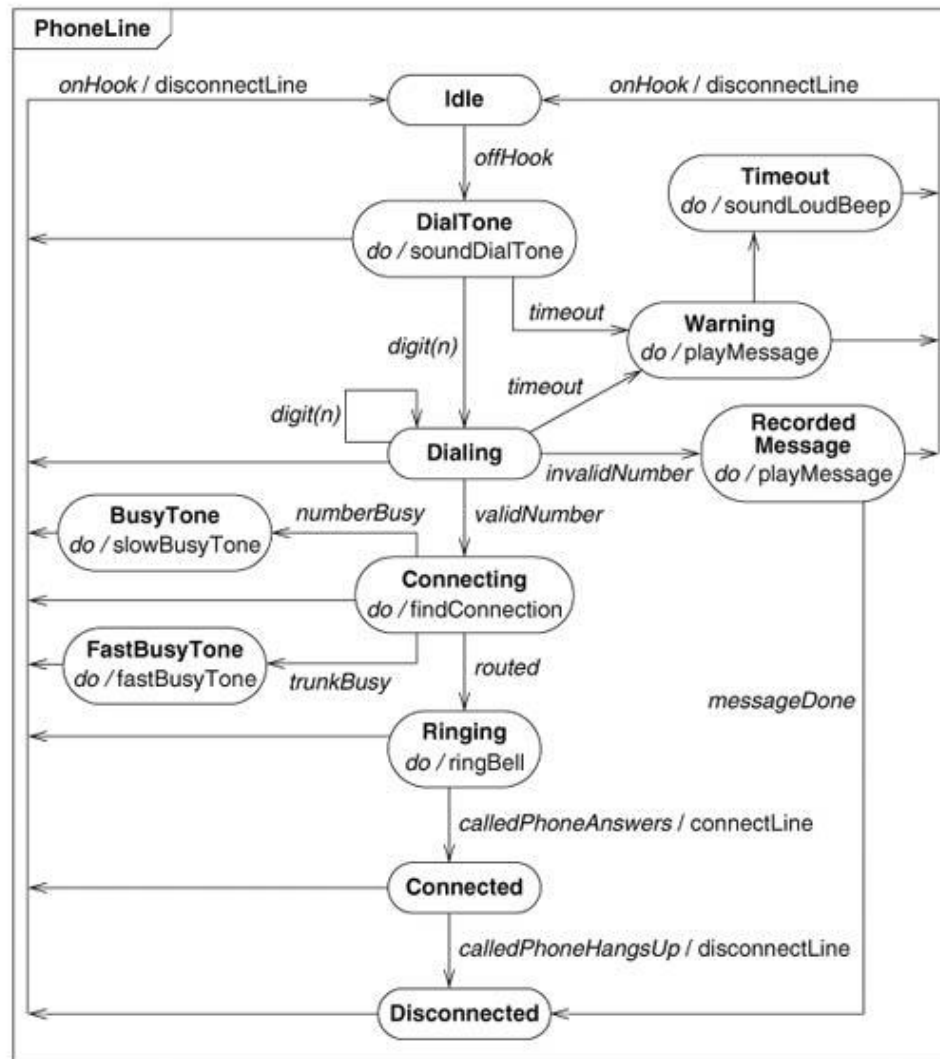
Nested State Diagram (Contd.)



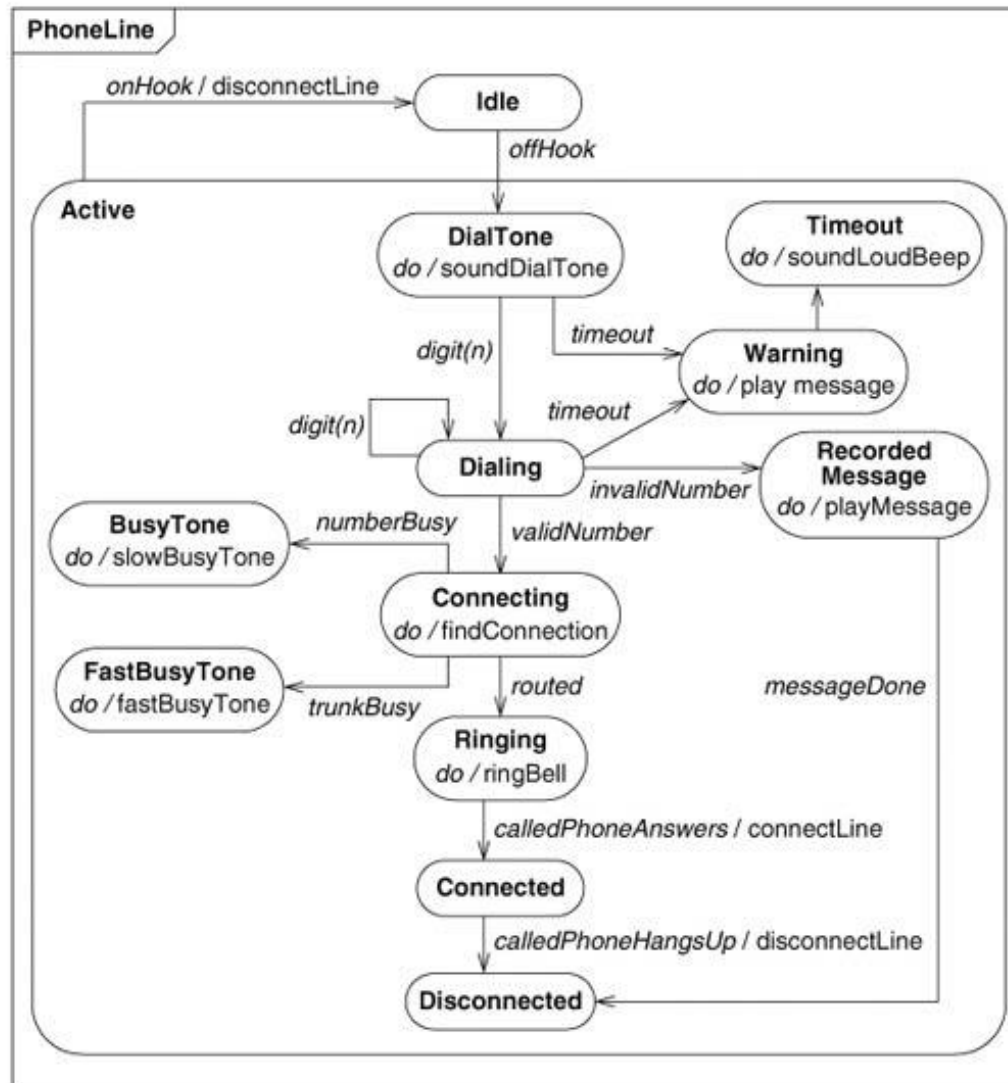
Nested States

- Show their commonality and share behavior
- Composite state
 - Encloses nested states
 - Arbitrary depth
- Receives outgoing transitions of its composite state

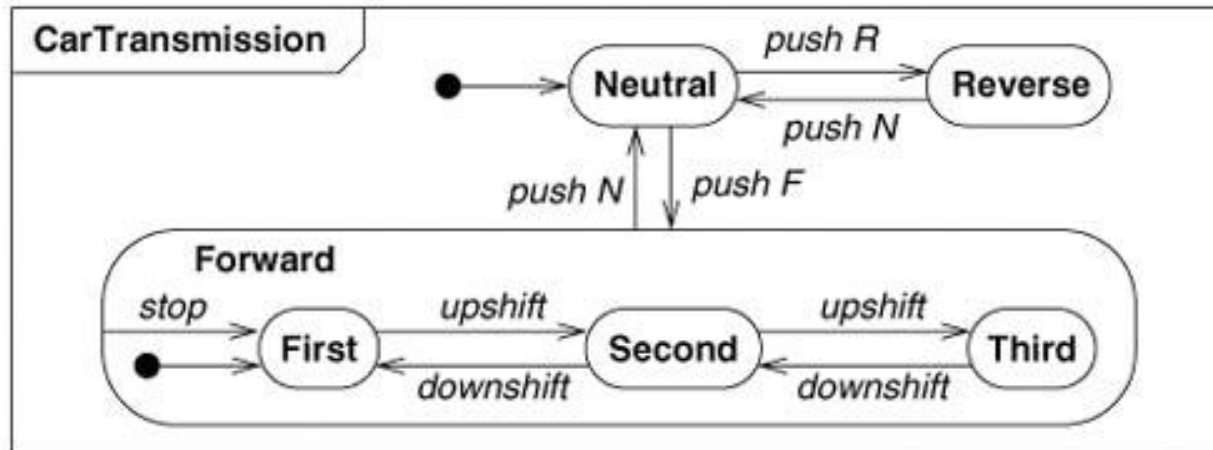
Nested States (E.g.)



Nested States (E.g. Contd.)

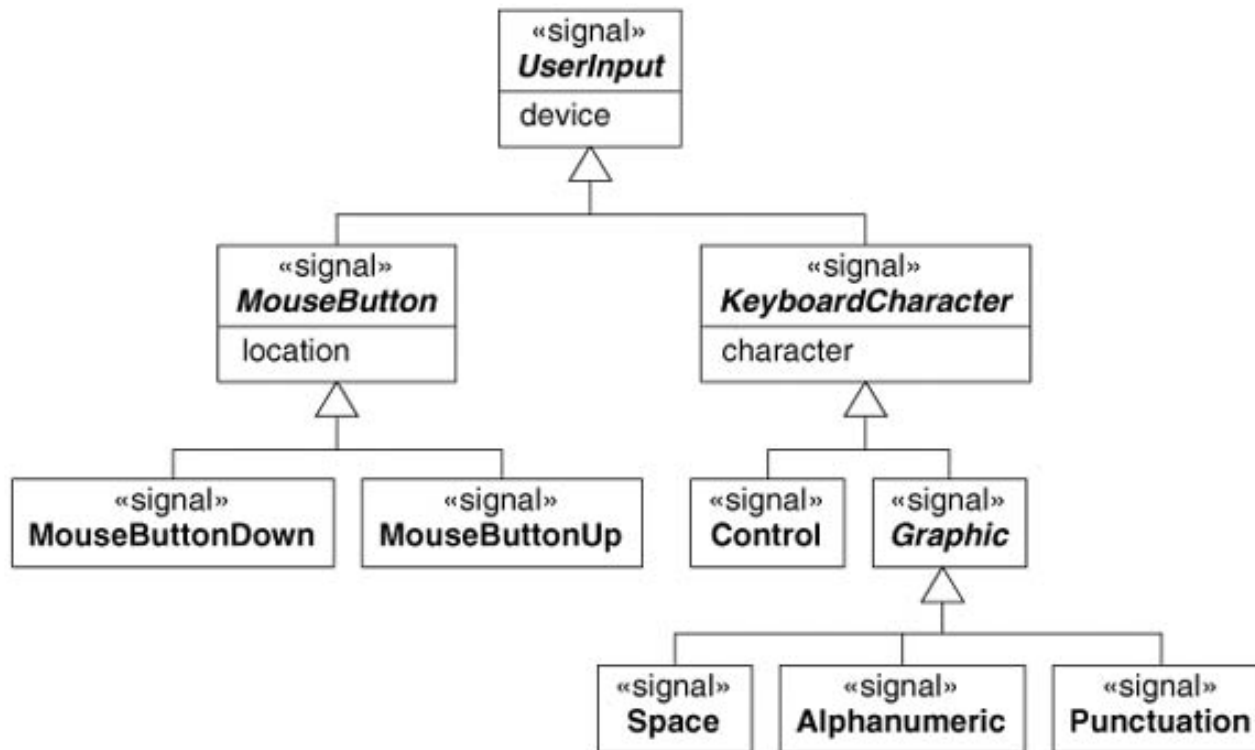


Nested States (Contd.)



Signal generalization

- Inheritance of signal attributes
- Received signal triggers transitions defined for any ancestor signal type



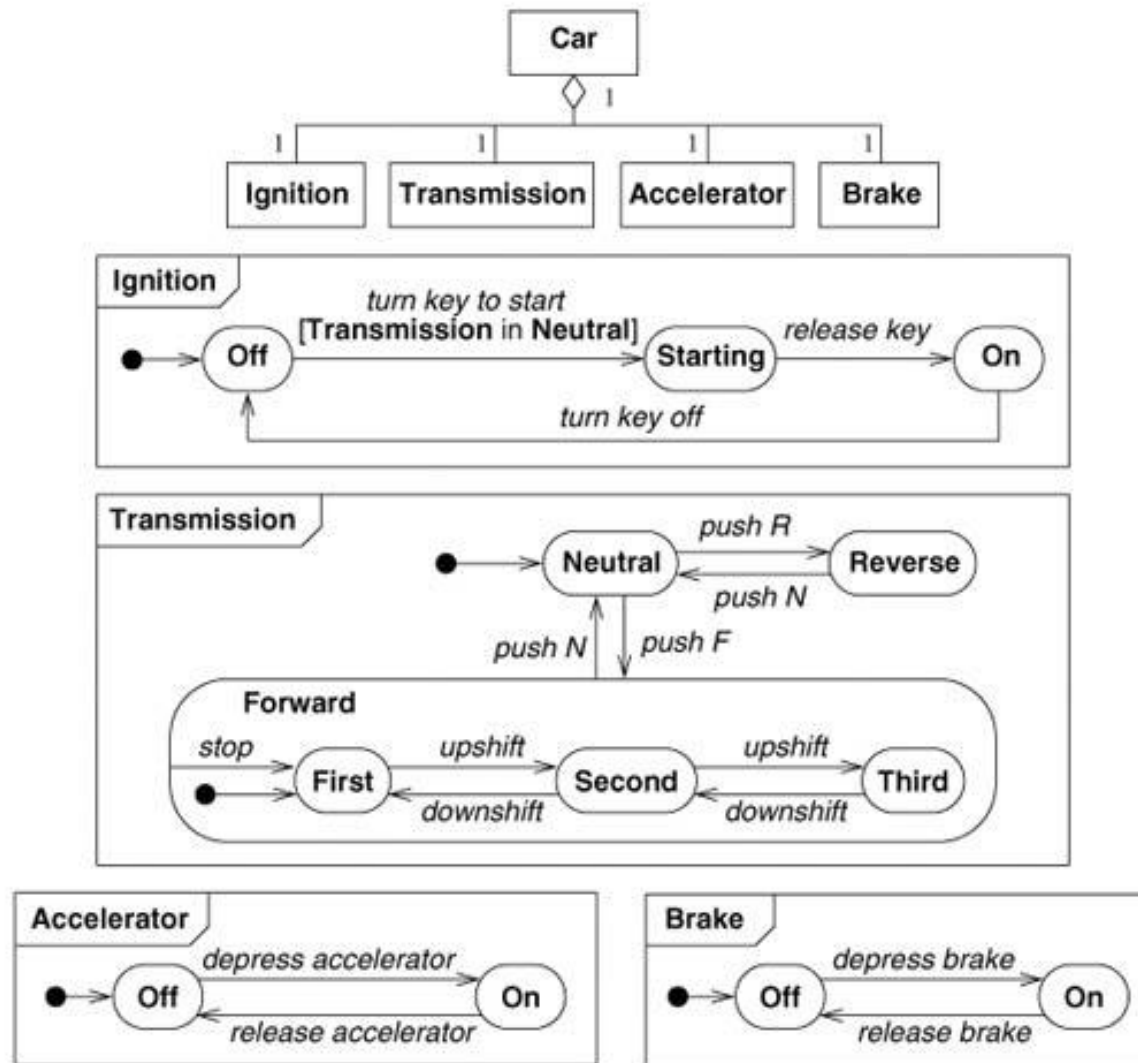
State Model: Concurrency

- Aggregation Concurrency
- Concurrency within an object
- Synchronization of concurrent activities

Aggregation concurrency

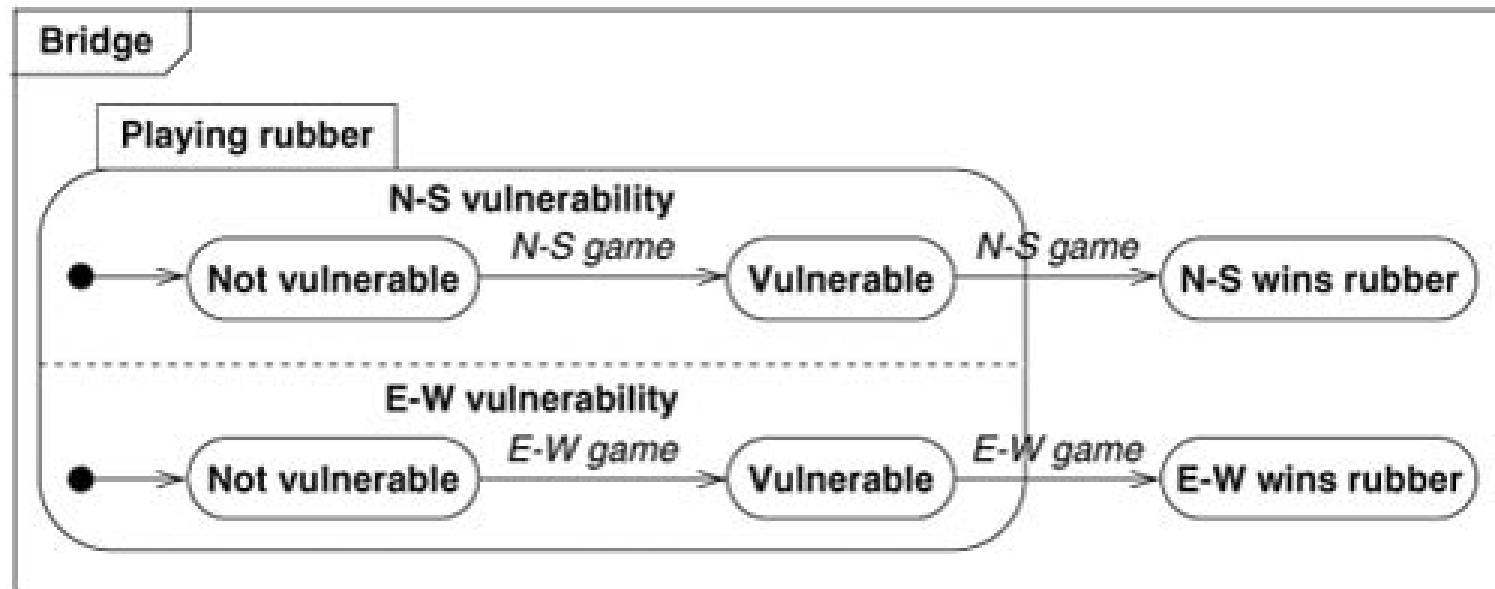
- Aggregation class
 - Whole / Part relationship
- Aggregate state diagram
 - Collection of state diagrams (one per part)
- Aggregate state
 - Combined states of all the parts (one from each)
- Optionally, part states have dependency on each other
 - Facilitates interaction between state diagrams

Aggregation concurrency (Contd.)



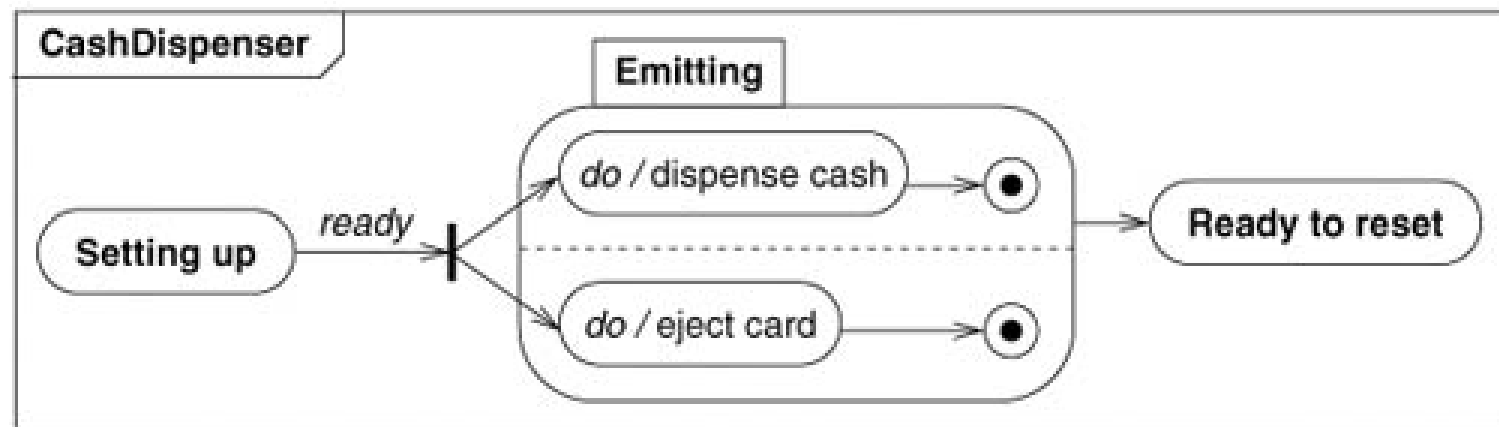
Concurrency within object

- Partition object into subsets of attributes/links
- State of object = one state from each subdiagram
- UML Notation:
 - Partitioning composite state into regions with dotted lines



Synchronization of concurrent activities

- Merge all concurrent activities
- Forked / Merged transition



State Model - Inheritance

- Subclasses
 - Inherit States & Transitions of ancestor class
 - May have own state diagrams
 - If disjoint attributes in state diagrams:
 - Composite state composed of concurrent state diagrams
 - Recommended approach
 - If same attributes in state diagrams:
 - Conflict exists
 - Subclass state diagram must be a refinement

Next sessions...

- Interaction Modeling

Reading assignment

- [Blaha] Chp 5: Pages 90-105
- [Blaha] Chp 6: Pages 110-126