CT503/CT871 Lecture 6: UML 5: State Model

- Lecture 5 Overview:
  - Systems Analysis: UML State Model
    - State Diagrams
    - Nesting of State Diagrams
    - Consistency with other Models
    - ATM Example: State Diagram
  - Systems Analysis: UML Diagrams & Models

State Model

- The UML State Model models aspects of the dynamic behaviour of the system
- A State Model has many state diagrams
- The state diagram models the life cycle history of a single reactive object
- The three key elements of state diagrams are:
  - State
  - Event
  - Transition

State Diagram

- A state diagram shows the life history of a single class
- A state diagram is a network of states and events: the enumeration of all the possible states of an object
- A stimulus from one object to another is an ***event***
- The response to an event depends on the ***state*** of the object receiving it
- Shows all the possible ***states*** objects can have and which ***events*** cause them to change state

Which Object Classes?

- Which object classes need state diagrams?
- During analysis, concentrate on those classes with significant dynamic behaviour:
  - Respond to external events
  - Generate and respond to internal events
  - Progress through different states during lifecycle
  - Could have current behaviour that depends on past behaviour

Which Object Classes?

- Which object classes need state diagrams?
- **Describe the type of class that is best represented by a behavioral state machine. Give two examples of classes that would be good candidates for behavioral state machine.**
- A class that responds to multiple events and that has significant changes in state over time.
- Examples include: invoices and purchase orders; inventories; bookings; accounts

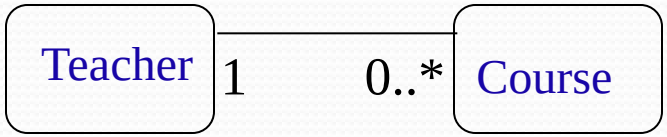
State

- One of the possible conditions in which an object may exist
- Object state changes as a result of events handled by the object
- The response of an object to an event may include an action or a change of state by the object
- A state specifies the response of the object to input events
- A state has duration: it occupies an interval of time; continuous activity: bell ringing or flying NY to Paris

State

- States may be distinguished by the value of certain attributes:
  - The Book object may only be borrowed if the value of the onShelf attribute is true

- States may also be distinguished by the existence of certain links
- Instances of class Teacher can have two states:
  - teaching
  - on sabbatical when no link



Events

- Something that happens at a point in time
- An event is a one-way transmission of information from one object to another that has no duration
- The state of the object determines the response to different events: e.g. adding a student to a course, creating a new course
- Two events that are causally unrelated are said to be concurrent: no effect on each other
- Most event classes have attributes indicating the information they convey

Events

- Attributes are shown in parentheses after the event class name
- Events include error conditions as well as normal occurrences: e.g. time-out, transaction aborted

airplane flight departs (airline, flight number, city)
mouse button pushed (button, location)
input string entered (text)
phone receiver lifted
digit dialed (digit)
engine speed enters danger zone

Event Classes and Attributes

Types of Events

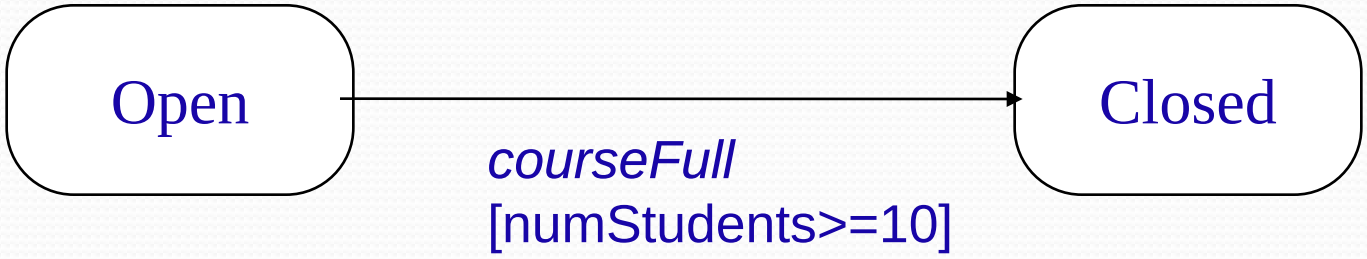
- There are several types of events, the most common are the signal, change and time event:
  - ***Signal*** event: the event of sending or receiving a signal, a one-way transmission of information
  - ***Change*** event: caused by the satisfaction of a boolean expression, whenever the expression changes from false to true, the event happens
  - ***Time*** event: caused by the occurrence of an absolute time (when) or the elapse of a time interval (after)

Transitions

- A transition: a change of state caused by an event
- Instantaneous
- A transition 'fires'
  - when its event occurs (except if guarded by a *guard condition*)
  - when changing from source state to target state
- UML notation for a transition is a line from the origin state to the target state

Guard Condition

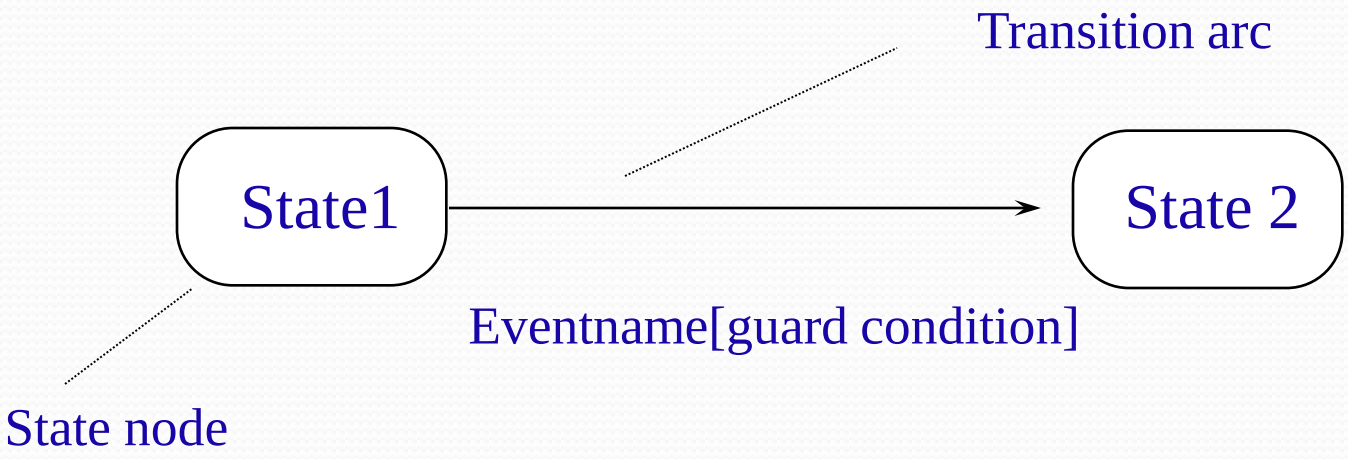
- A guard condition allows a transition between states only if the condition is true
- A guard condition is a Boolean function of attribute values: e.g “numStudents>= 10”
- Valid over an interval of time
- Can be used as guards on transitions: a Boolean in brackets after the event name (Example: Traffic lights))



State Diagram

- Relates events and states
- A state diagram: a graph whose nodes are states and whose arcs are transitions (event names) between states
- Specifies the state sequence caused by an event sequence
- A state is drawn as a round box containing a name
- A transition is drawn as an arrow from the receiving state to the target state

State Diagram Notation



One-Shot Lifecycle

- State diagrams can represent one-shot life cycles or continuous loops (phone line)
- One-shot diagrams have initial and final states
- An initial state is shown by a solid circle
- A final state is shown by a bull's-eye
- Both can be labeled to indicate different initial and final conditions
- A one-shot diagram can be considered a state diagram "subroutine" that can be referenced from various places in a high-level diagram

State Diagram Behaviour: Operations

- Events trigger operations
- **Operations**, attached to states or transitions, are performed in response to the corresponding states or events
- An **activity**: operation that takes time to complete
- Associated with a state: starts when the state is entered, can run to completion or can be interrupted by an outgoing transition

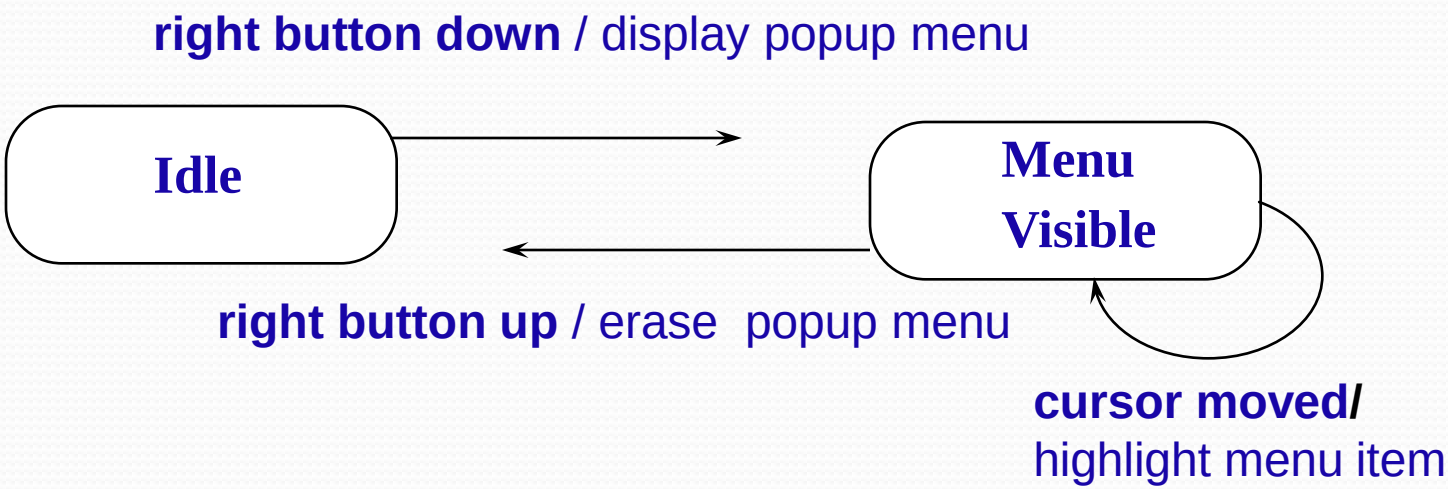
Operations: Activities

- Activities include both continuous and sequential operations
- The notation “do: A” within a state box indicates that activity A starts on entry to the state and stops on exit (continuous) or when complete (sequential)

Operations: Actions

- An action is an instantaneous operation that is associated with a transition; considered non-interruptible
- The notation for an action on a transition is a slash ('/') and the name of the action after the name of the event that causes it

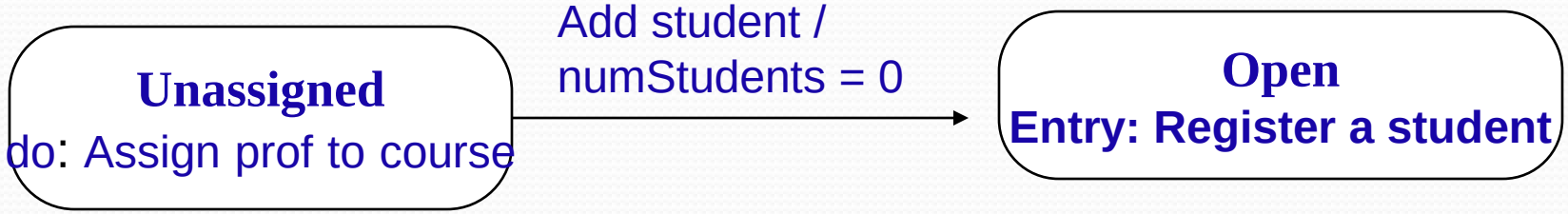
Actions: Example



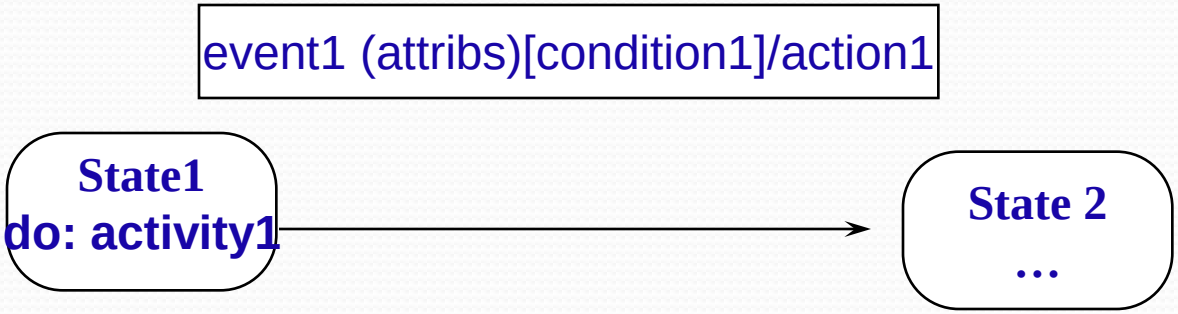
Actions for pop-up menu

Operations: Actions

- When an action must occur no matter how a state is entered or exited, the action can be associated with the state
- Action is associated with every transition entering or exiting the state
- The action is shown inside the state icon preceded by the keyword ***entry*** or ***exit***



State Diagram Notation Summary



Summary of Notation for State Diagrams

How to?

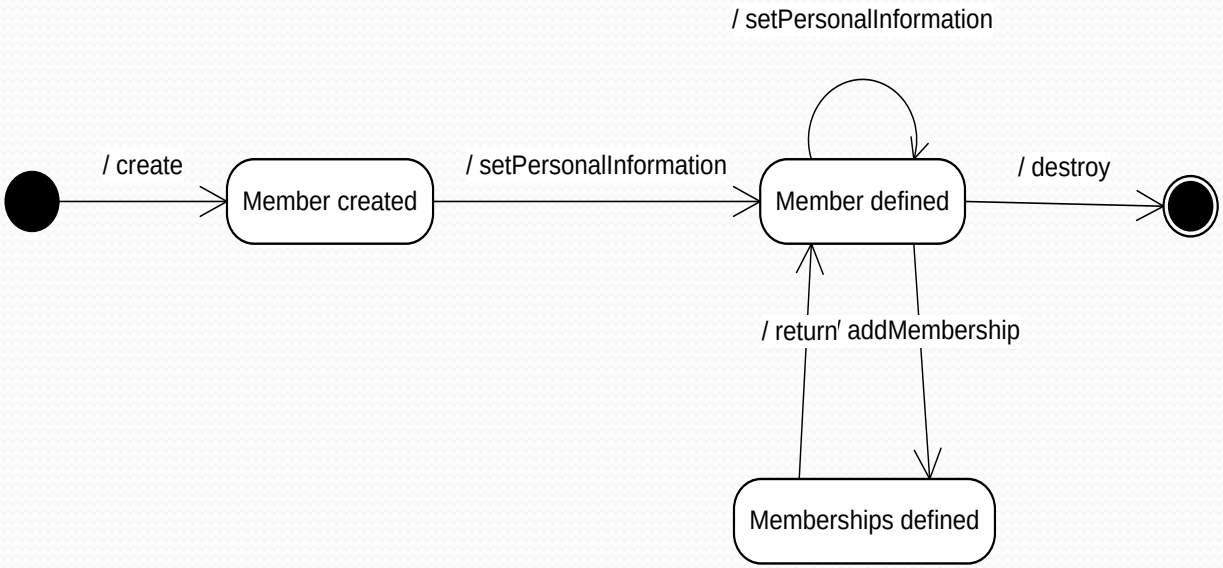
- It can be hard to identify all necessary states
- State diagrams can be developed incrementally
  - consider individual sequences of events received by an object
  - these might be specified on interaction diagrams
  - start with a statechart for one interaction
  - add states as required by additional interactions

Example Problem

A simple digital watch has a display and two buttons to set it, the A button and the B button. The watch has two modes of operation, display time and set time. In the display time mode, hours and minutes are displayed, separated by a flashing colon. The set time mode has two submodes, set hours and set minutes. The A button is used to select modes. Each time it is pressed, the mode advances in the sequence: display, set hour, set minutes, display, etc. Within the submodes, the B button is used to advance the hours or minutes once each time it is pressed. Buttons must be released before they can generate another event.

- Prepare a state diagram for the watch.

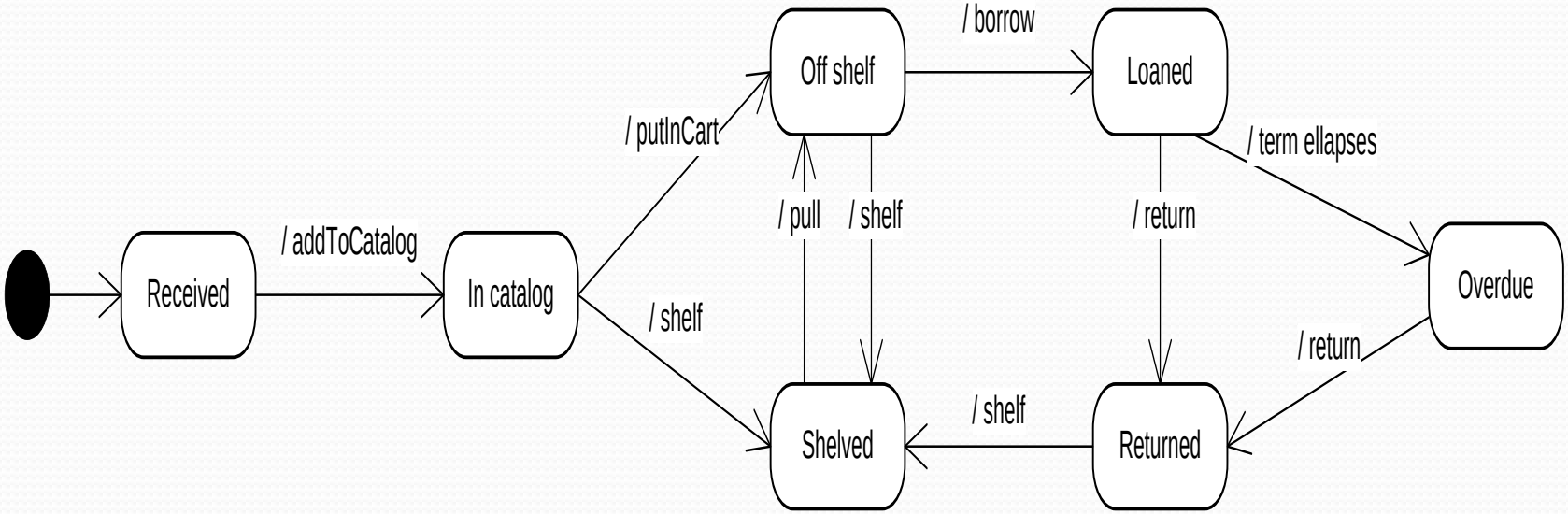
State Diagram Example



Video Example:

- Draw a state machine to describe the various states that a video goes through from the time it is purchased and placed on the display shelf through the rental and returns process
- Customers are not allowed to rent videos when they have overdue videos
- Fines must be paid before new videos can be rented

State Diagram Example

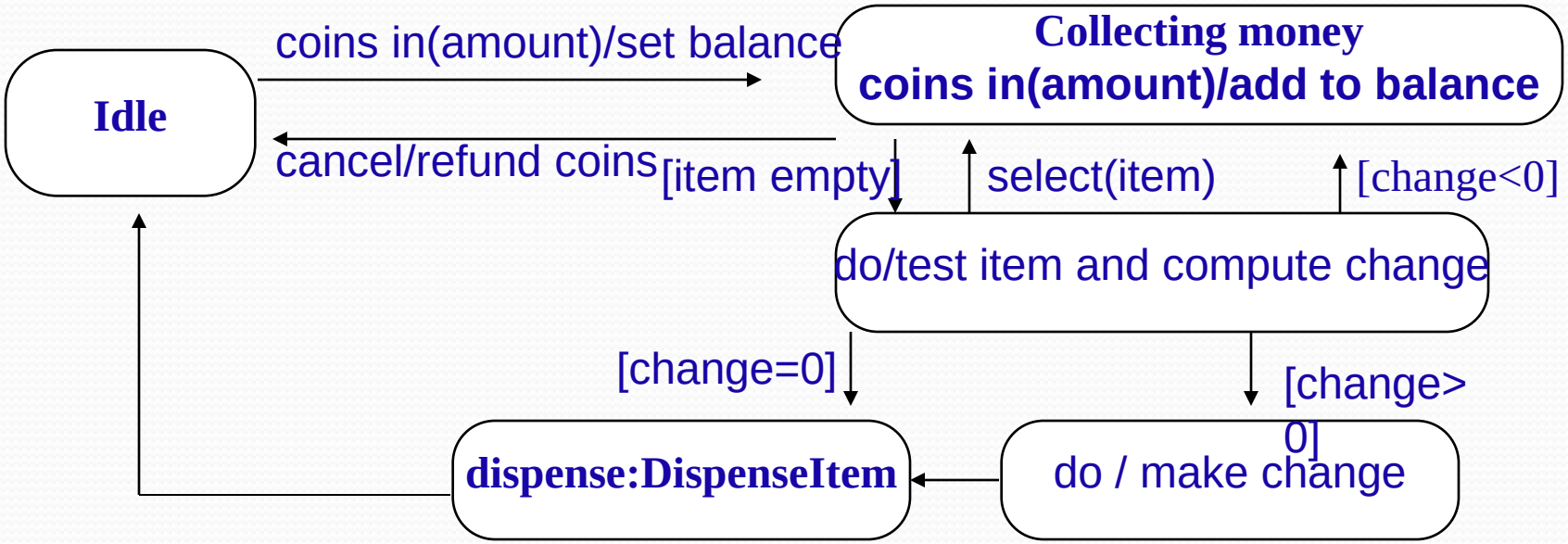


Nesting State Diagrams

- State diagrams can become very large & complex
- Nested states may be used to simplify complex diagrams
- Not generalisation: UML 2.0
- Composite state: state that contains nested states
- An activity in a state can be expanded as a lower-level state diagram: each state representing one step of the activity
- The states in the nested diagram are all refinements of the state in the high-level diagram

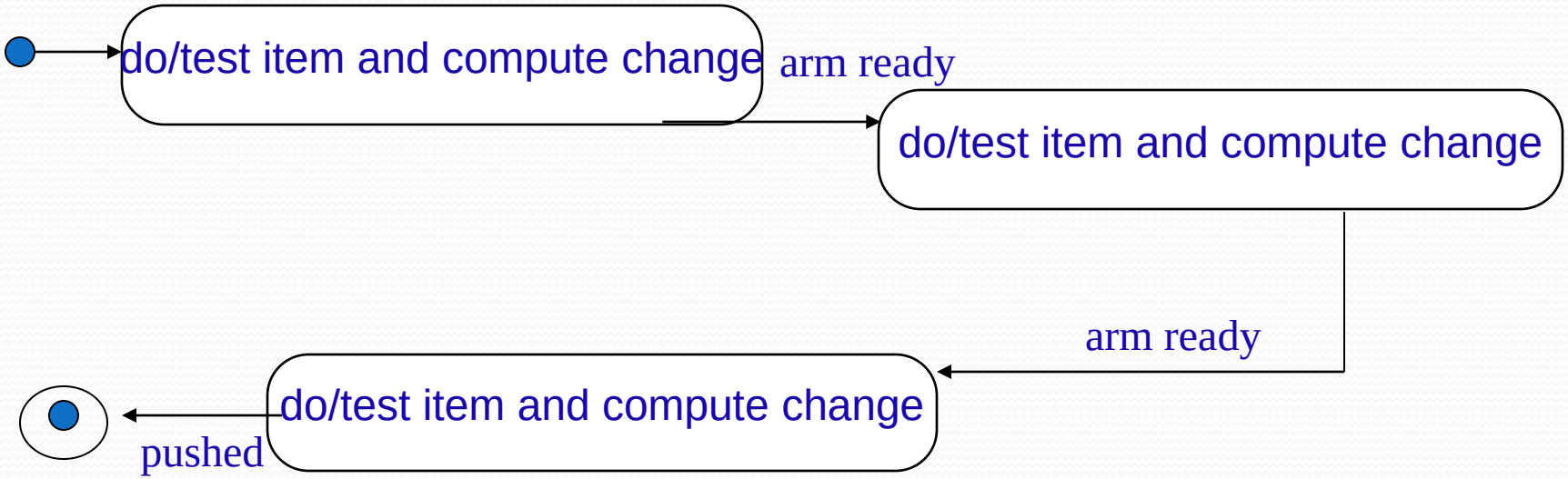
Nesting State Diagrams

- Example: Vending machine. Top level shows an activity dispense item and event select (item) that are expanded in more detail: nested state diagrams



Nesting State Diagrams

DispenseItem

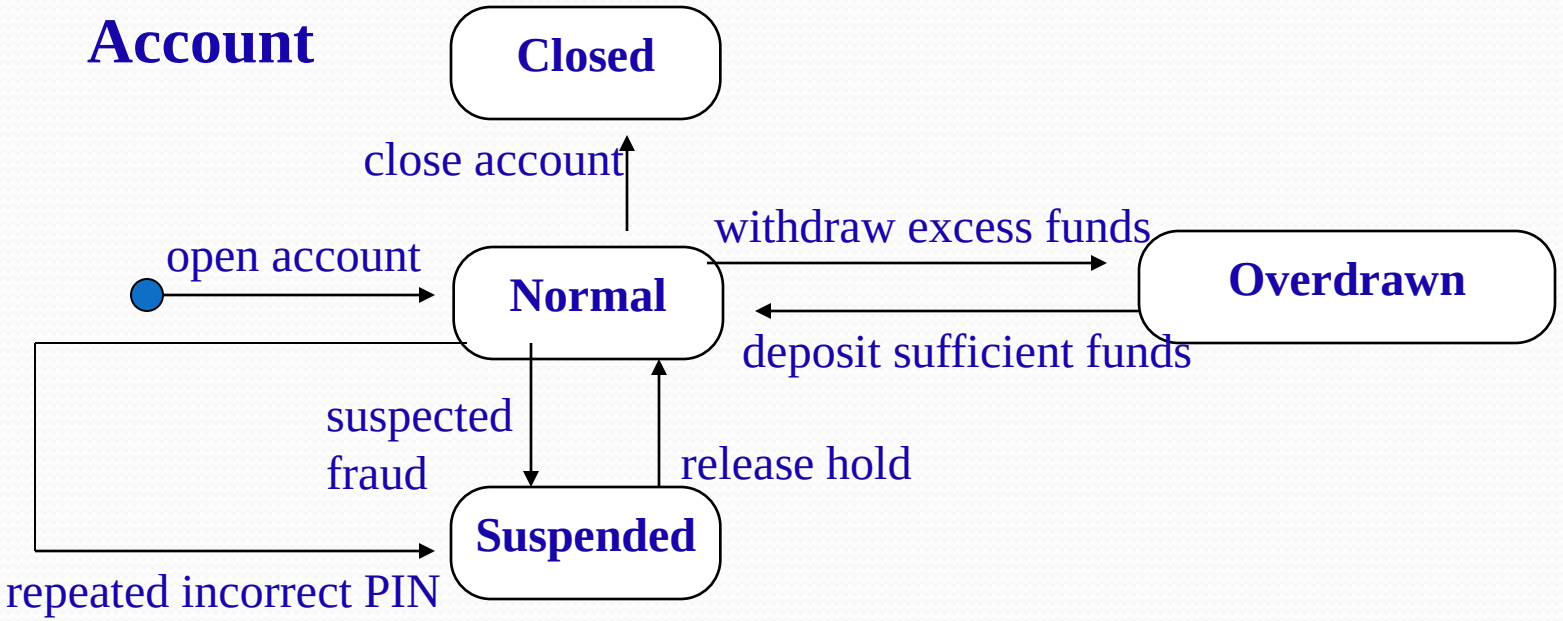


Consistency Across Models

- Consistency between UML state model and class model can be checked as follows:
  - Every action should correspond to the execution of an operation on the appropriate class
  - Every event should correspond to an incoming message / operation on the appropriate class
  - Every outgoing message sent from a state machine must correspond to an operation on another class

ATM Account Example

- Domain state model for ATM Account:



UML Diagrams & Models

CT871/CT503 Semester II UML Lecture 4

UML Diagrams

- Behaviour: describes system functionality
  - *Use Case*: how system behaves in use
  - *Activity*: business / operational behaviour of system components
  - *Interaction*: class interaction in a use case
  - *State*: single class behaviour
- Structure:
  - *Class*

UML Analysis Models

- Class Case Study next week & class assignment
- Domain Model:
  - Use Case Model
  - Domain Class Model
- Application Model:
  - Interaction Model
  - State Model
  - Application Class Model

UML CASE Tools

- EG: *Visual Paradigm*, Sparx Systems *Enterprise Architect*, *Eclipse* MDT (Model Development Tools), *Star UML*, *Altova*, *ArgoUML*
- MOF: Meta Object Facility: industry-standard environment
- XMI: OMG's XML-based standard format for model transmission & storage
- MDA: Model Driven Architecture
- BPMN
- UexCeler

UML Lecture 5 Review

- Systems Analysis: UML State Model
  - State Diagrams
  - Nesting of State Diagrams
  - Consistency with other Models
  - ATM Example: State Diagram
- Systems Analysis: UML Analysis Models
  - Domain Model
  - Application Model