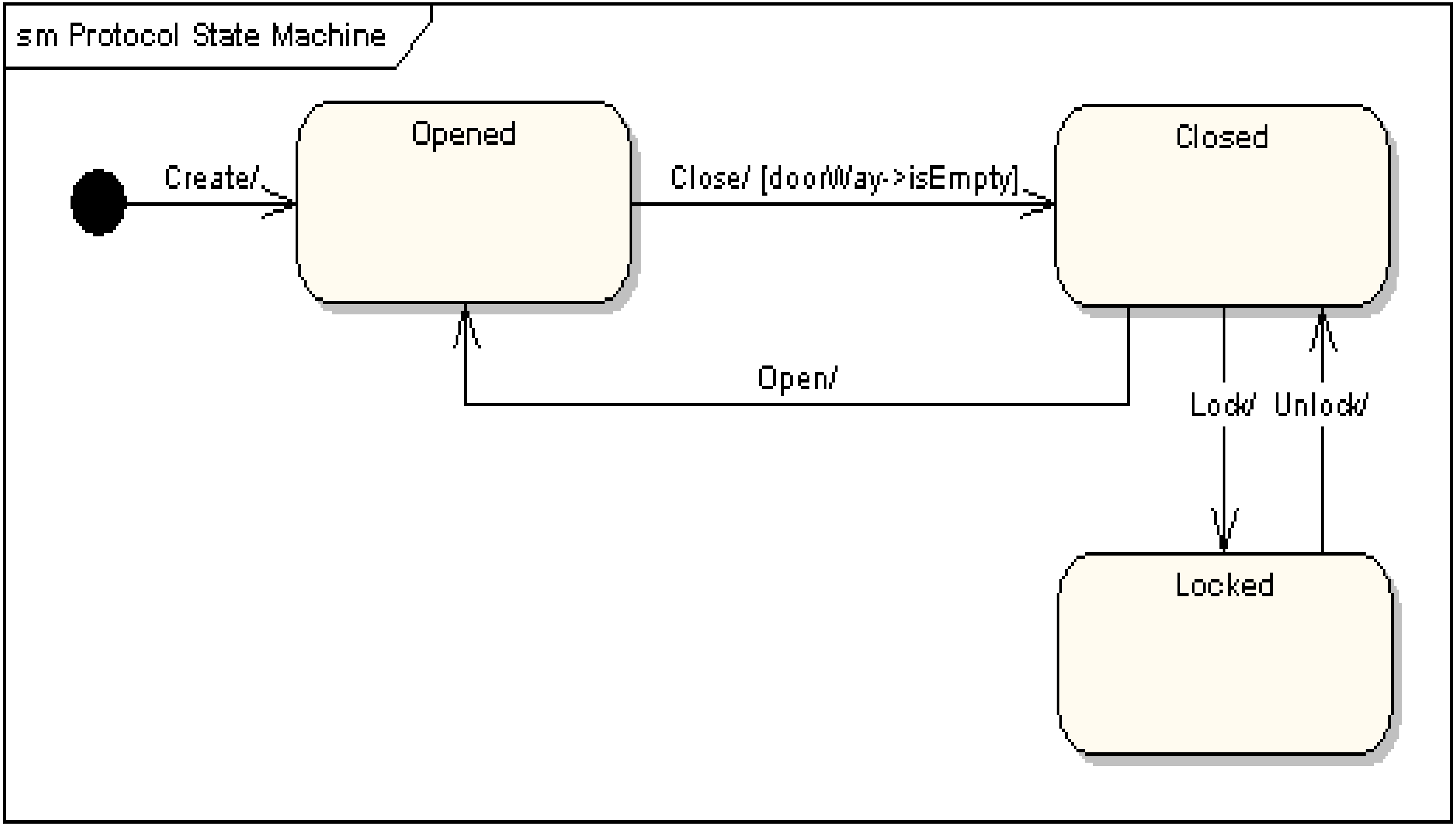
State Machine Diagrams

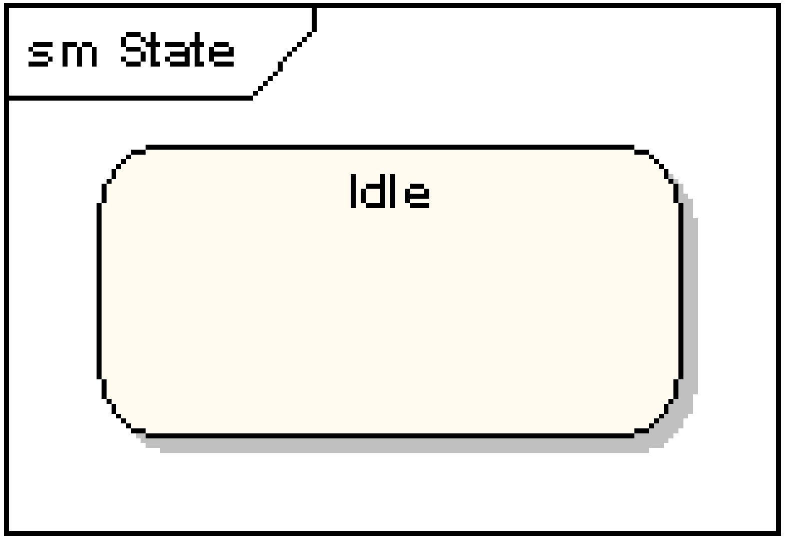
State Machine Diagrams

- A state machine diagram models the behavior of a single object, specifying the sequence of events that an object goes through during its lifetime in response to events.
- As an example, the following state machine diagram shows the states that a door goes through during its lifetime.


sm Protocol State Machine
Opened
Closed
Locked
Create/
Close/ [doorWay->isEmpty]
Open/
Lock/
Unlock/

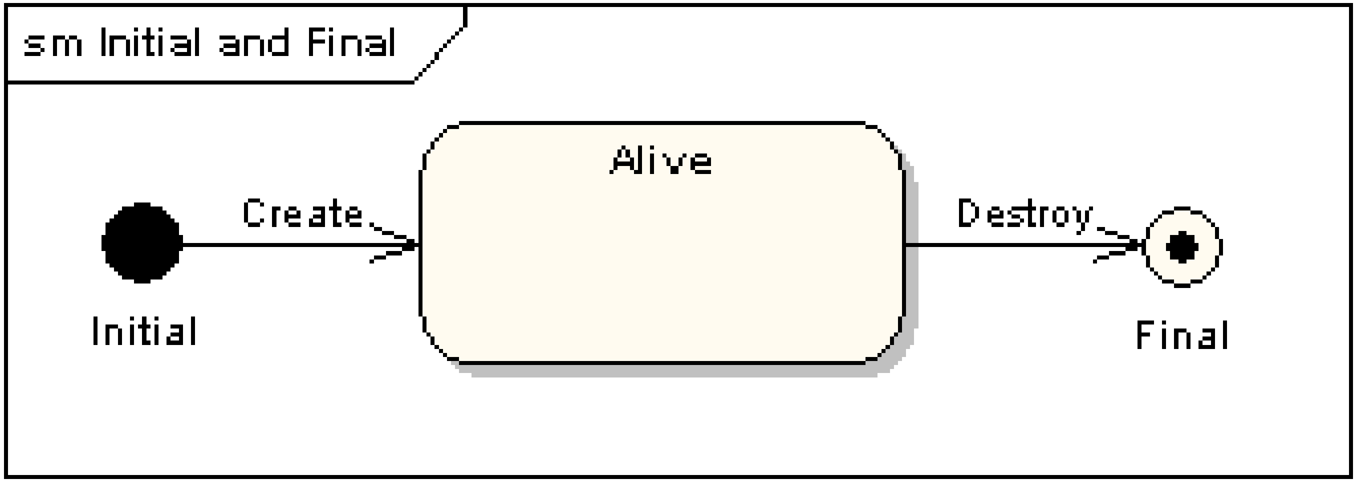
State Machine Diagrams

- ***States***
  - A state is denoted by a round-cornered rectangle with the name of the state written inside it.



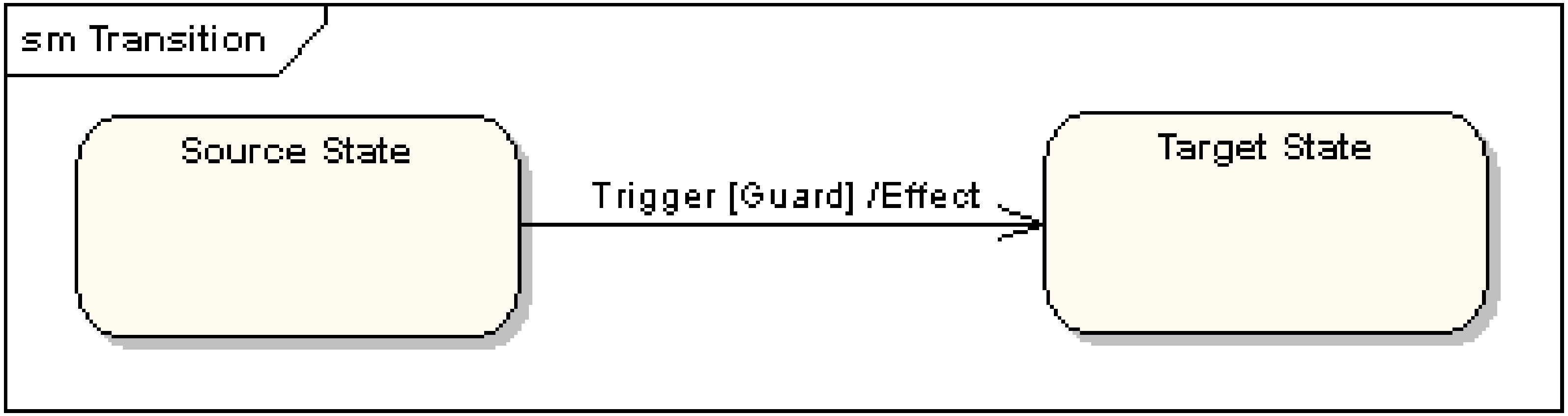
State Machine Diagrams

- ***Initial and Final States***
  - The initial state is denoted by a filled black circle and may be labeled with a name.
  - The final state is denoted by a circle with a dot inside and may also be labeled with a name.



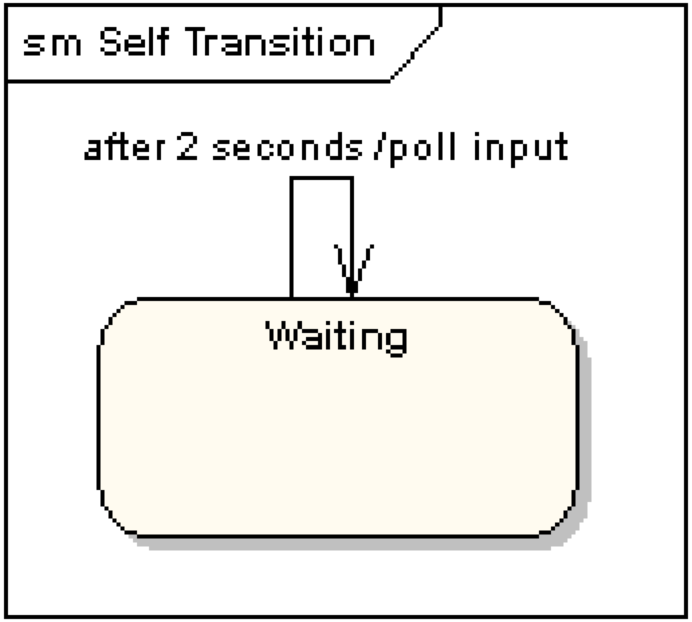
State Machine Diagrams

- ***Transitions***
  - Transitions from one state to the next are denoted by lines with arrowheads.
  - A transition may have a trigger, a guard and an effect, as below.



State Machine Diagrams

- ***Self-Transitions***
  - A state can have a transition that returns to itself.
  - This is most useful when an effect is associated with the transition.

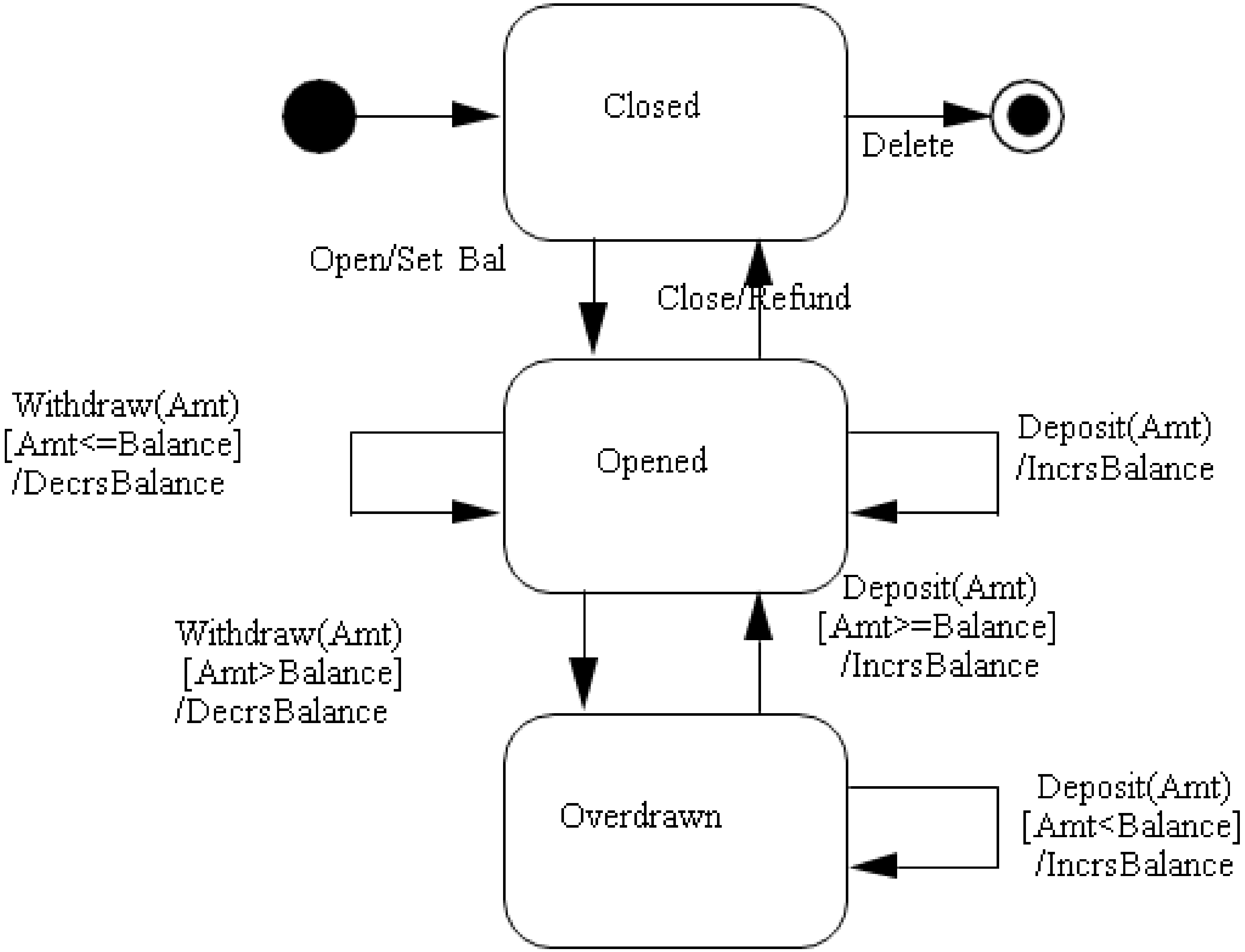


State Machine Diagrams

- "***Trigger***"
  - is the cause of the transition, which could be a signal, an event, a change in some condition, or the passage of time.

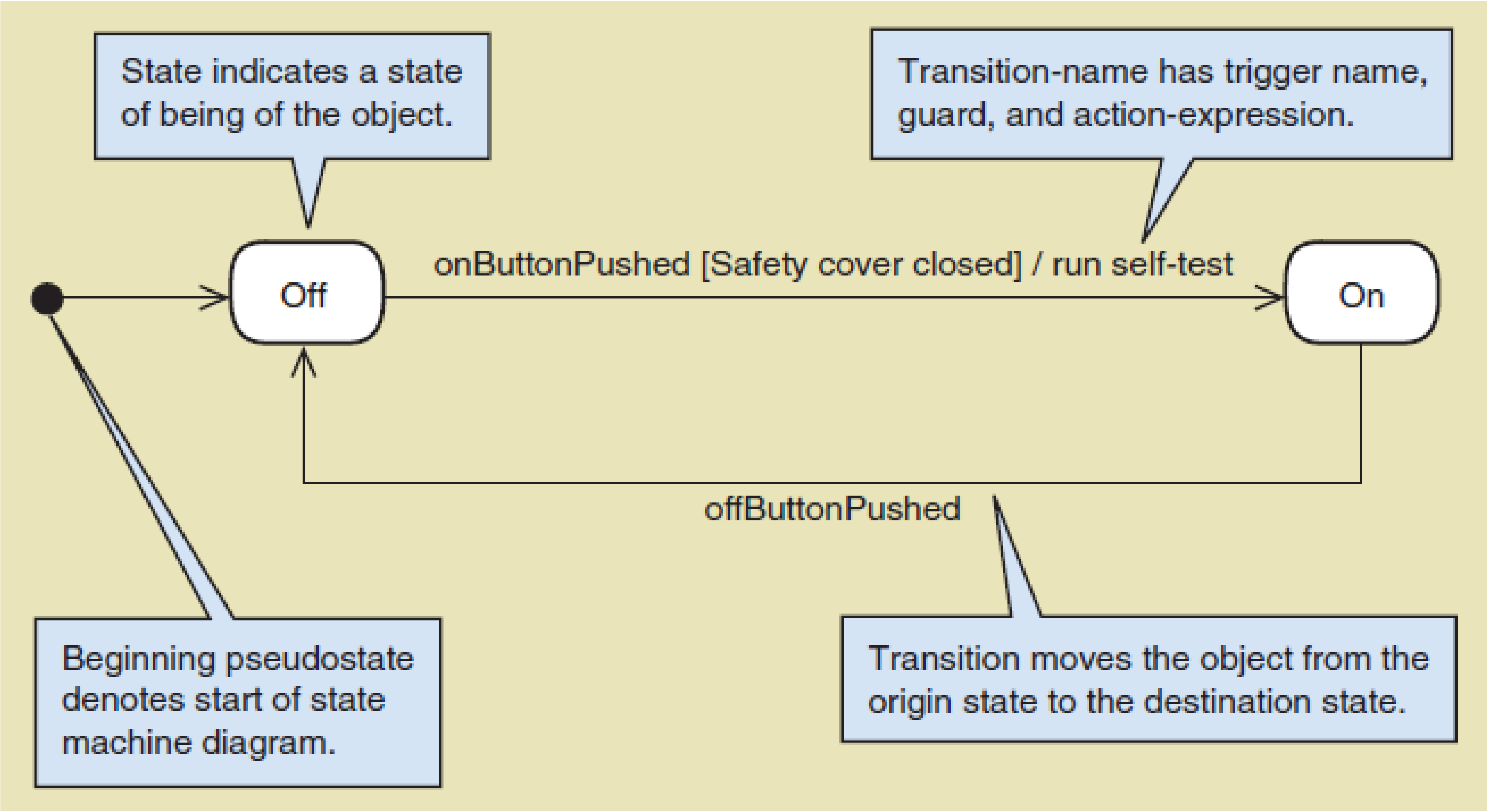
- "***Guard***"
  - is a condition which must be true in order for the trigger to cause the transition.

- ***"Effect"***
  - is an action which will be invoked directly on the object that owns the state machine as a result of the transition.


Closed
Delete
Open/Set Bal
Close/Refund
Withdraw(Amt)
[Amt<=Balance]
/DecrsBalance
Opened
Deposit(Amt)
/IncrsBalance
Withdraw(Amt)
[Amt>Balance]
/DecrsBalance
Deposit(Amt)
[Amt>=Balance]
/IncrsBalance
Overdrawn
Deposit(Amt)
[Amt<Balance]
/IncrsBalance

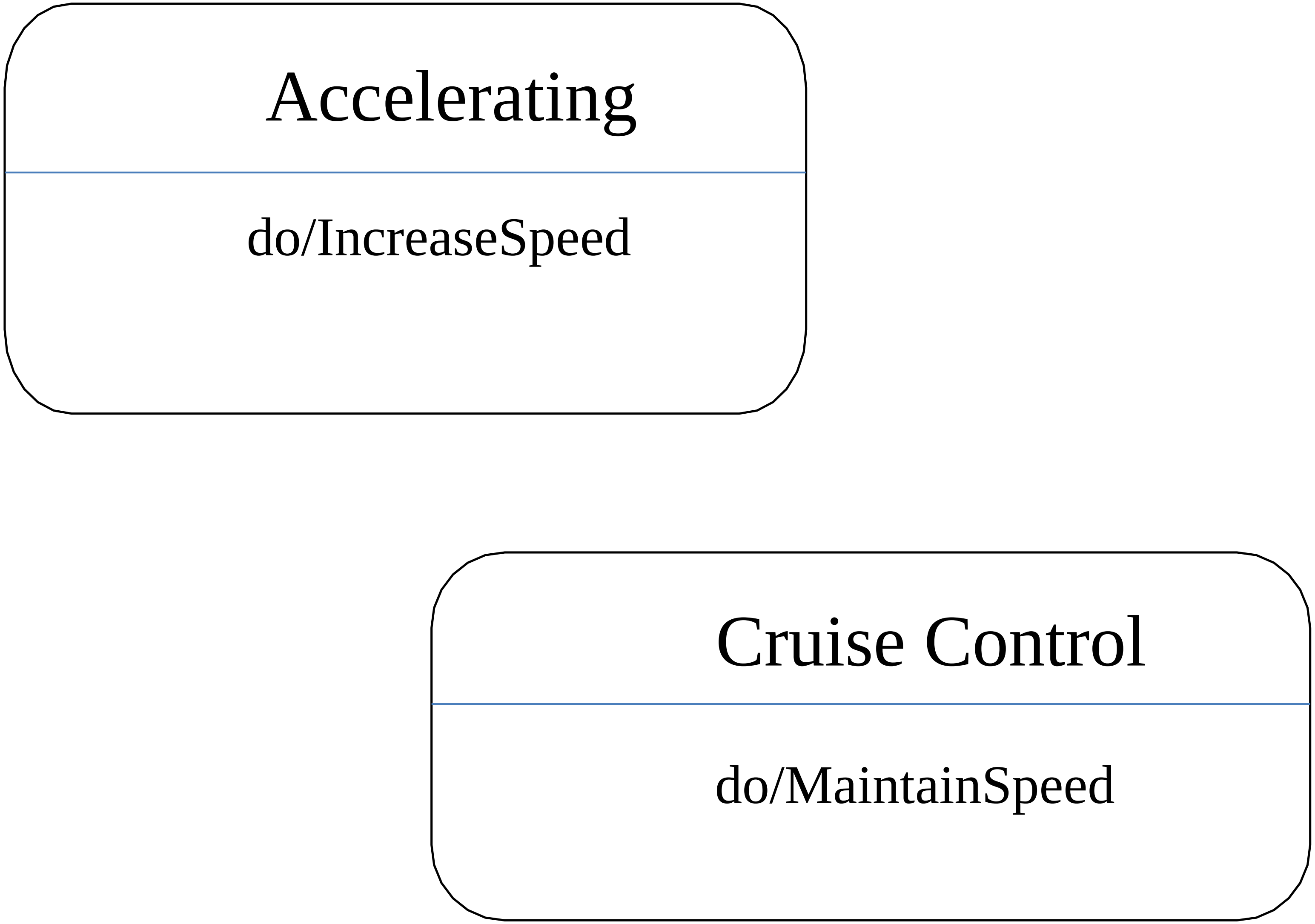
State Machine Diagram

for a Printer


State indicates a state of being of the object.
Transition-name has trigger name, guard, and action-expression.
Off
onButtonPushed [Safety cover closed] / run self-test
On
offButtonPushed
Beginning pseudostate denotes start of state machine diagram.
Transition moves the object from the origin state to the destination state.

State Machine Diagrams

- Activity
  - An Activity is the UML way to specify that some relatively long-term amount of work gets done while an object is in a state
  - The work is continuous and interruptible (it stops when you exit the state)
- Notation
  - Compartmentalize the state
  - Include "do/activity-name" in the lower compartment of every state that has an activity



Accelerating

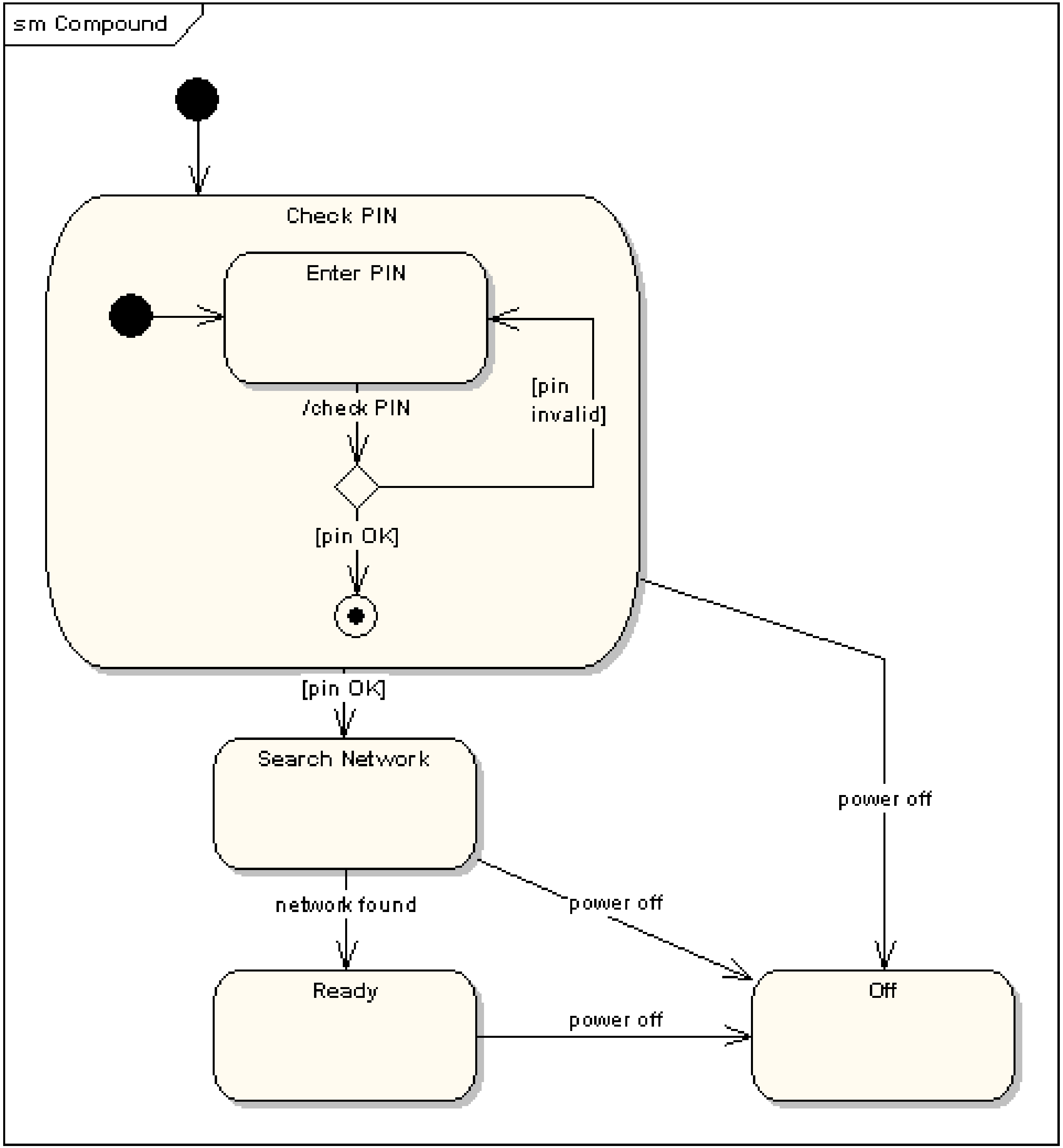
do/IncreaseSpeed

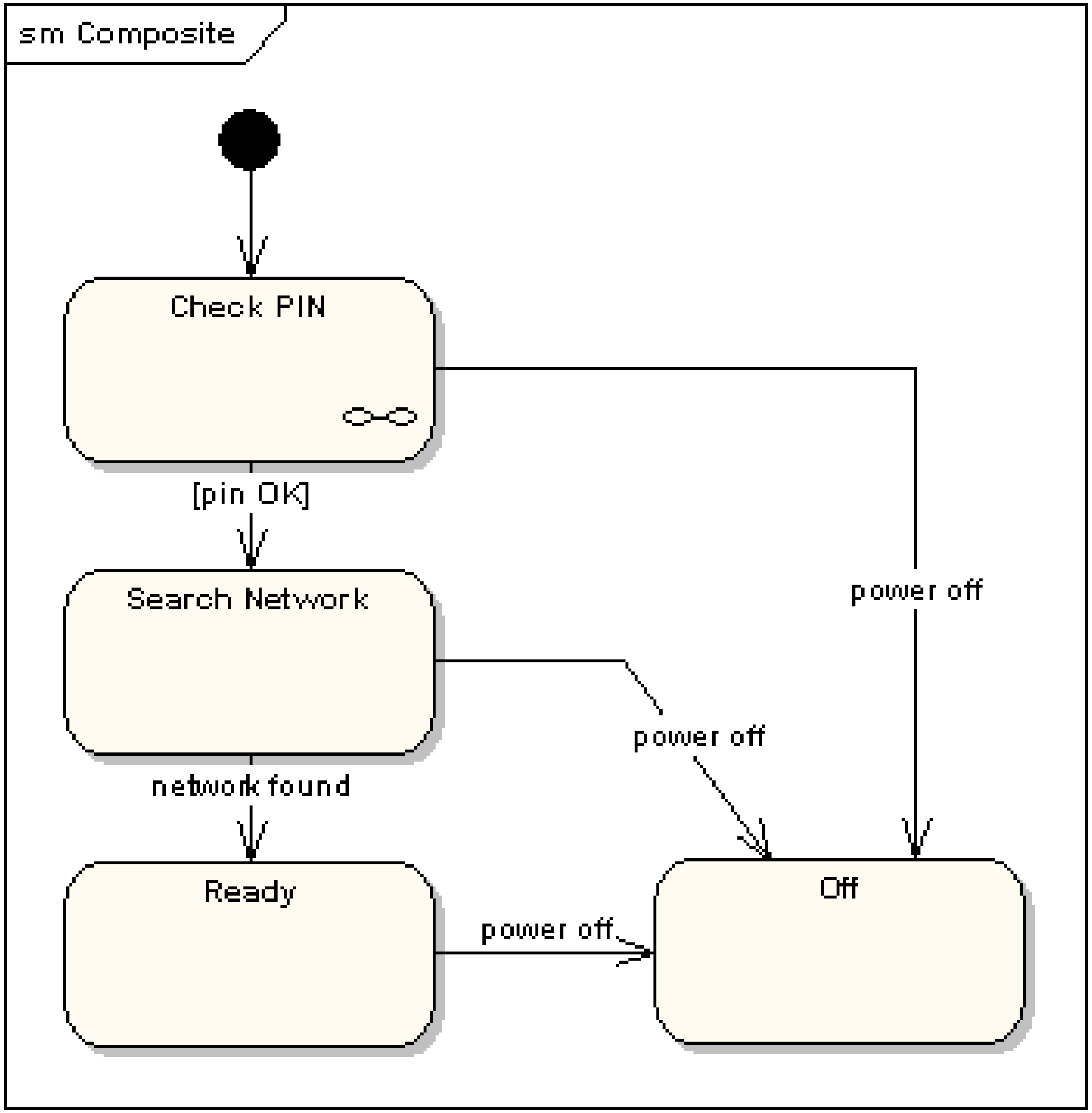
Cruise Control

do/MaintainSpeed

State Machine Diagrams

- ***Compound States*** - A state machine diagram may include sub-state-machine diagrams.


sm Compound
Check PIN
Enter PIN
/check PIN
[pin invalid]
[pin OK]
[pin OK]
Search Network
network found
power off
power off
power off
Ready
Off

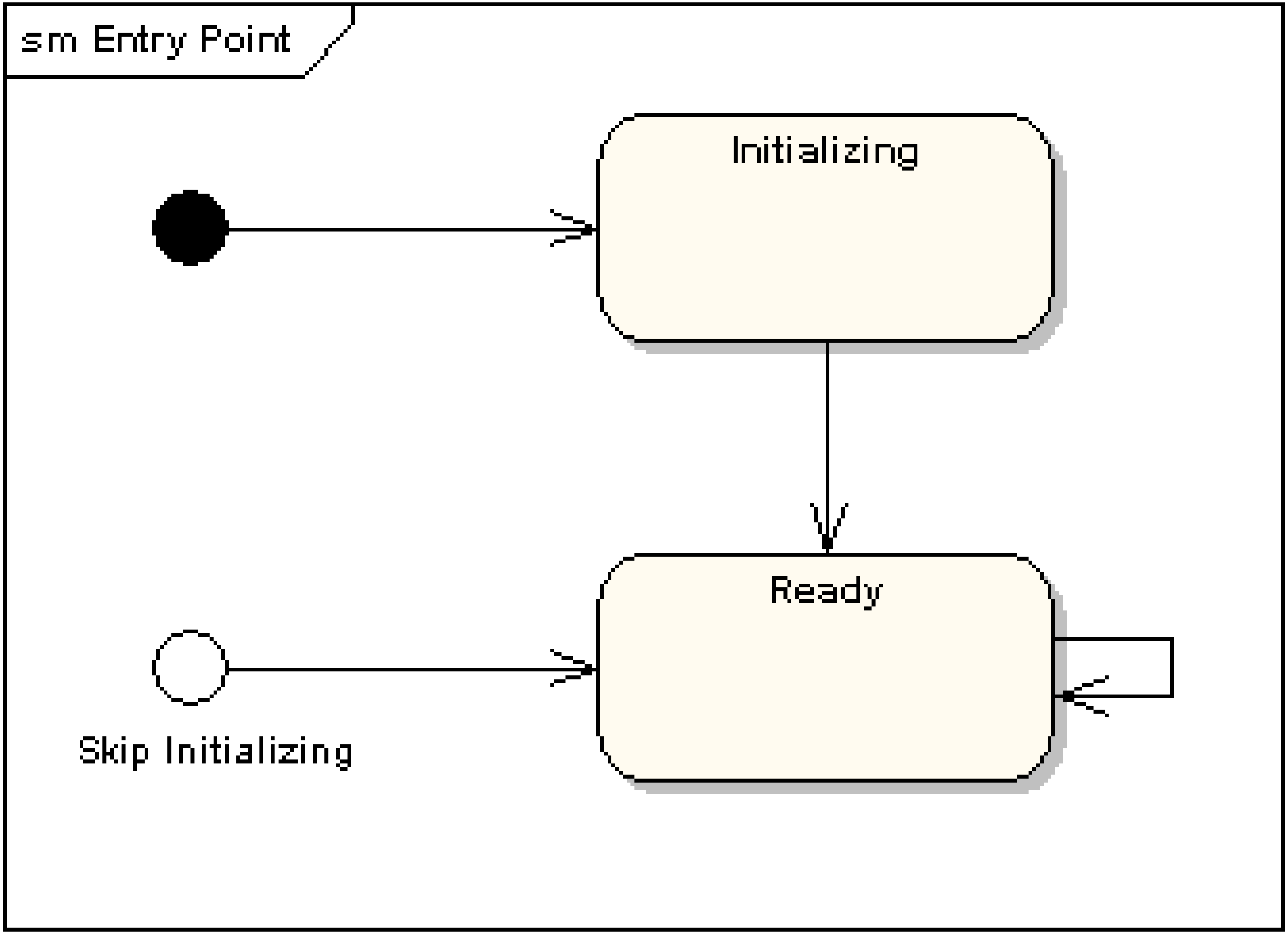


Alternative way to show the same information

- The ∞ symbol indicates that details of the Check PIN sub-machine are shown in a separate diagram.

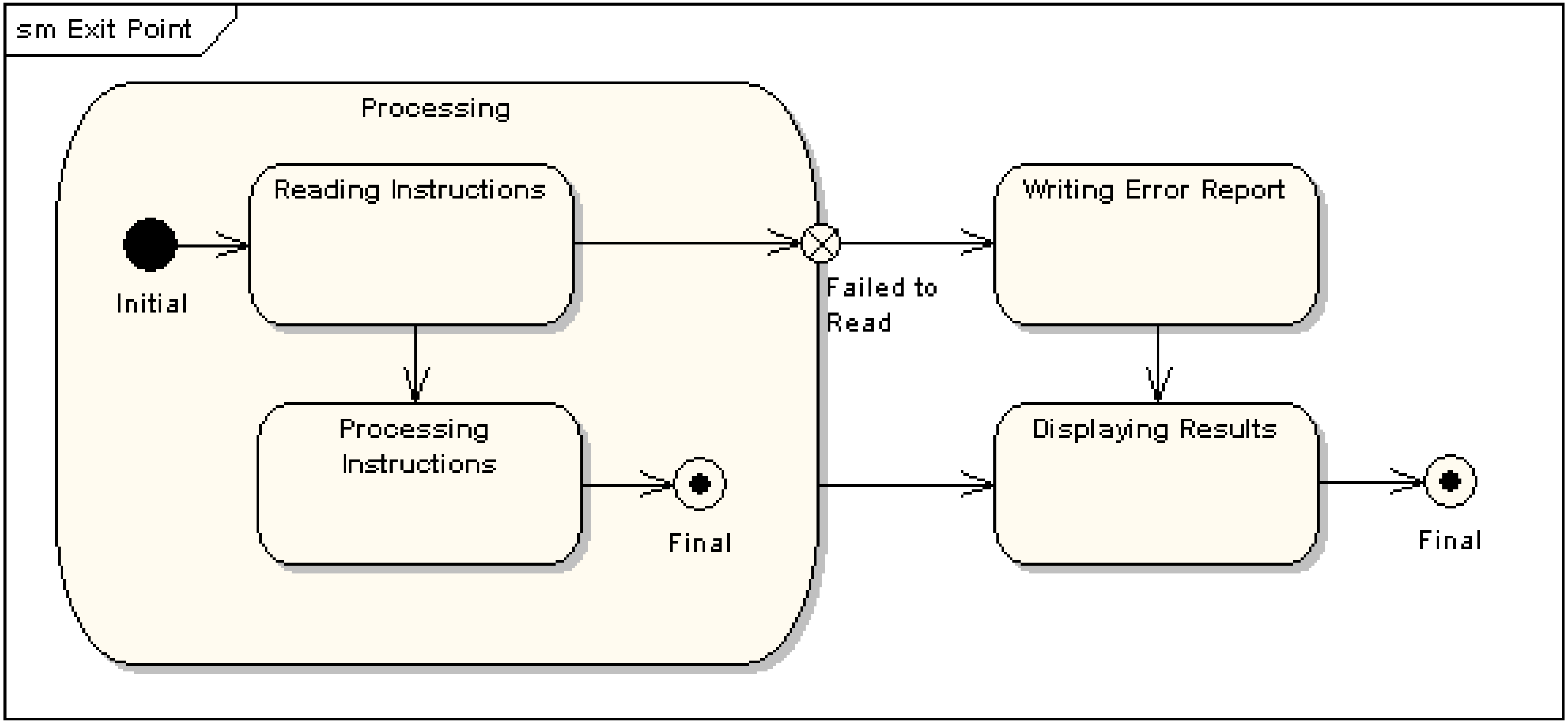
State Machine Diagrams

- ***Entry Point***
  - Sometimes it would be normal to begin in the "Initializing" state,
  - If for some reason it wasn’t necessary to perform the initialization, it would be possible to begin in the "Ready" state by transitioning to the named entry point.


sm Entry Point
Initializing
Ready
Skip Initializing

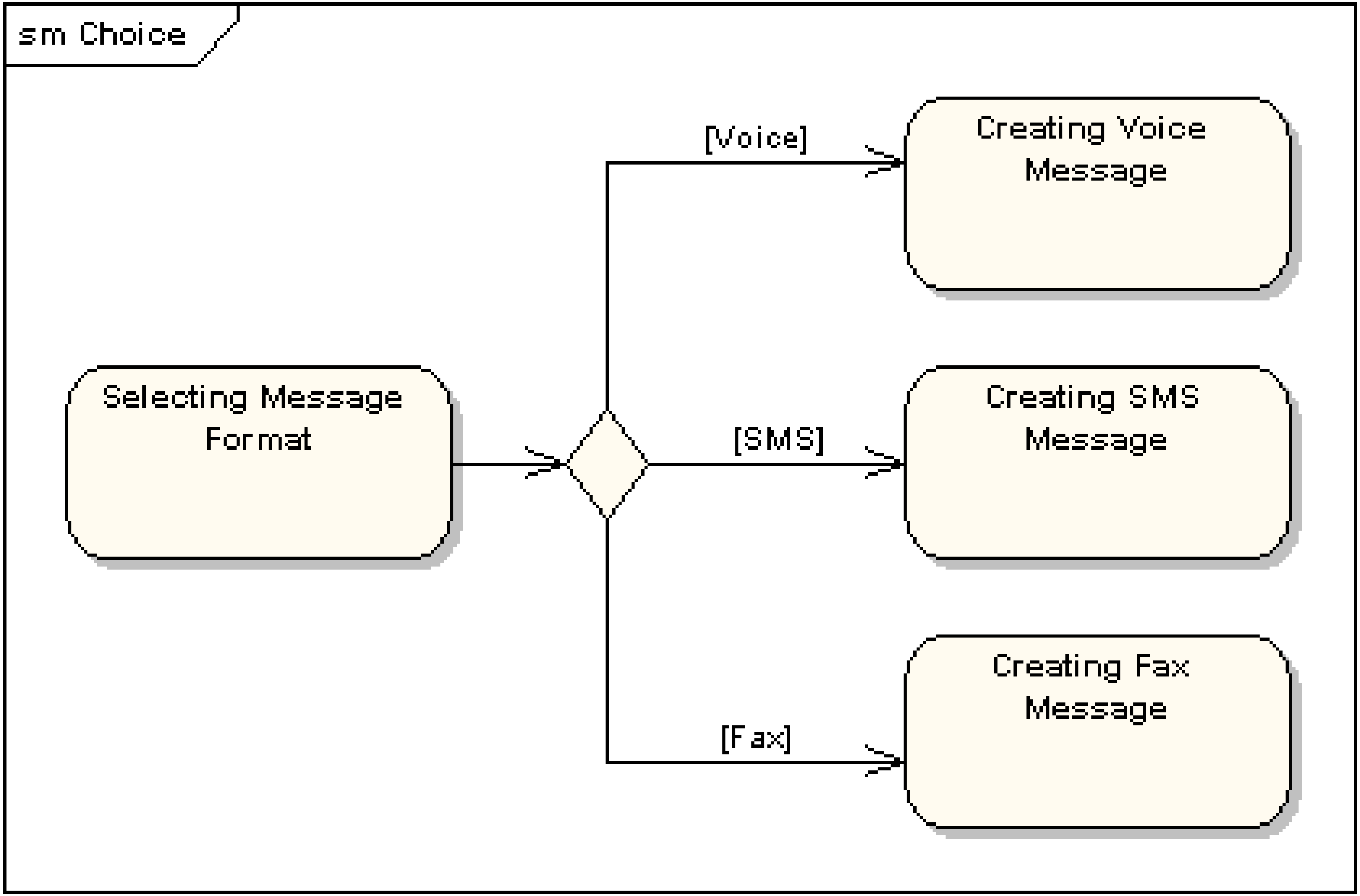
State Machine Diagrams

- ***Exit Point***
  - **I**t is possible to have named alternative exit points.
  - The following diagram gives an example where the state executed after the main processing state depends on which route is used to transition out of the state.


sm Exit Point
Processing
Reading Instructions
Initial
Processing Instructions
Final
Failed to Read
Writing Error Report
Displaying Results
Final

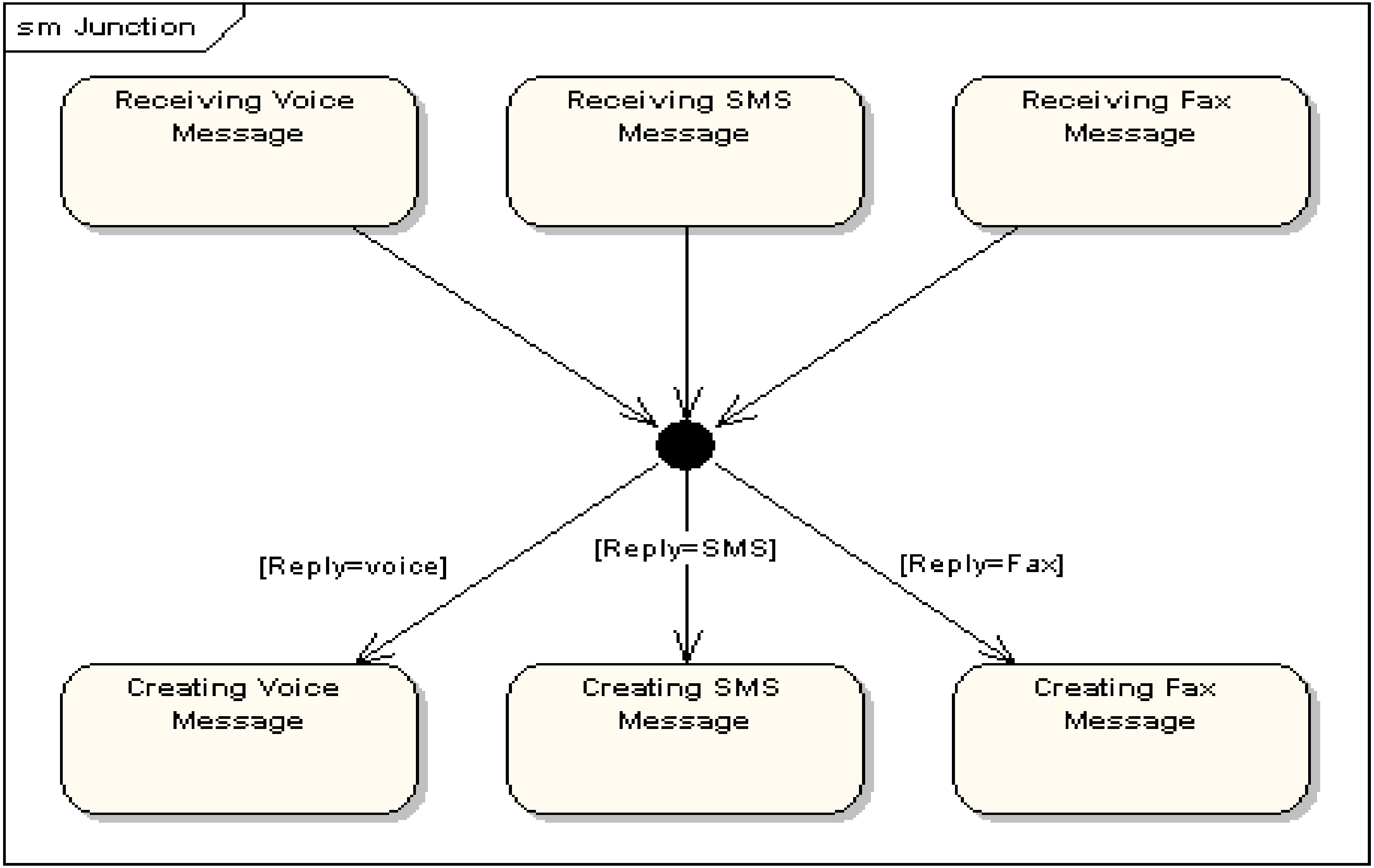
State Machine Diagrams

- ***Choice Pseudo-State***
  - A choice pseudo-state is shown as a diamond with one transition arriving and two or more transitions leaving.
  - Whichever state is arrived at, after the choice pseudo-state, is dependent on the message format selected during execution of the previous state.

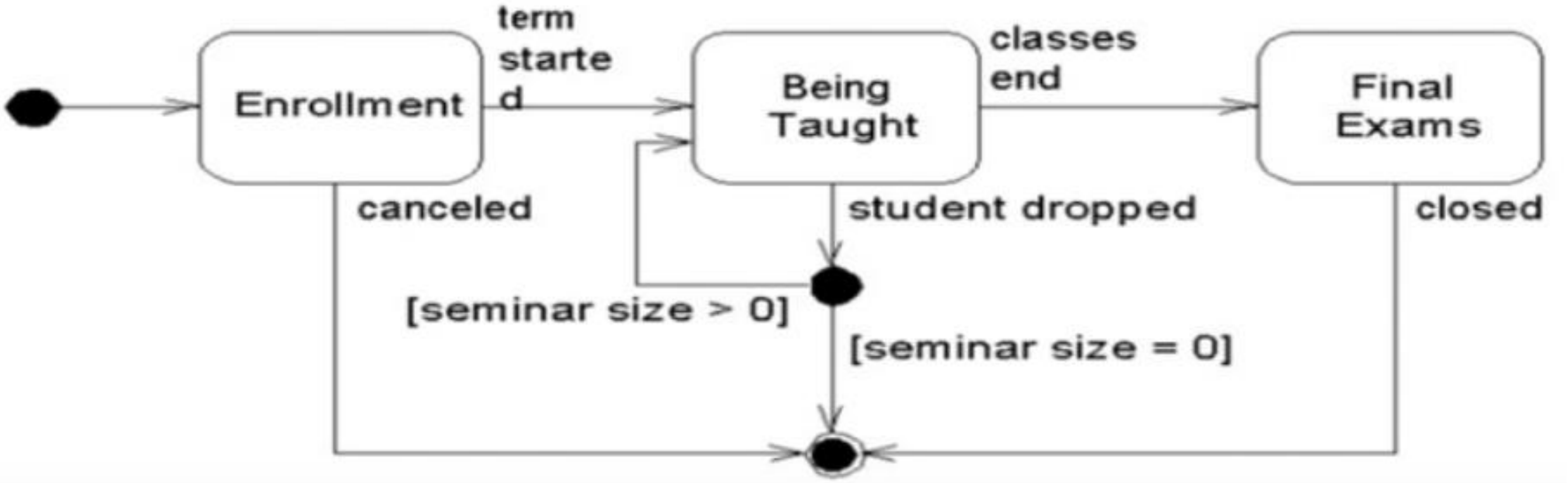

sm Choice
Selecting Message Format
[Voice]
[SMS]
[Fax]
Creating Voice Message
Creating SMS Message
Creating Fax Message

State Machine Diagrams

- ***Junction Pseudo-State***
  - Junction pseudo-states are used to chain together multiple transitions.
  - A single junction can have one or more incoming, and one or more outgoing, transitions; a guard can be applied to each transition.
  - Junctions are semantic-free. (do not carry specific meaning or context)
  - A junction which splits an incoming transition in to multiple outgoing transitions realizes a static conditional branch, as opposed to a choice pseudo-state which realizes a dynamic conditional branch.

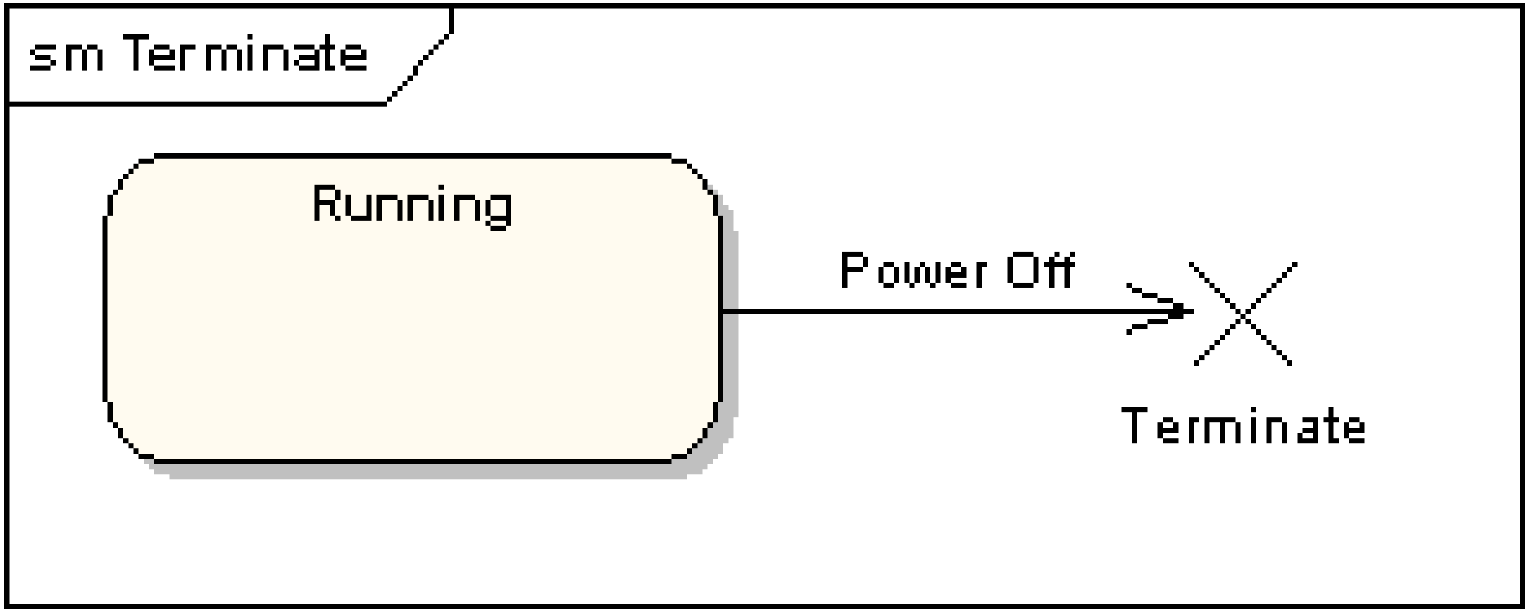

sm Junction
Receiving Voice Message
Receiving SMS Message
Receiving Fax Message
[Reply=voice]
[Reply=SMS]
[Reply=Fax]
Creating Voice Message
Creating SMS Message
Creating Fax Message

Junction Pseudo-State



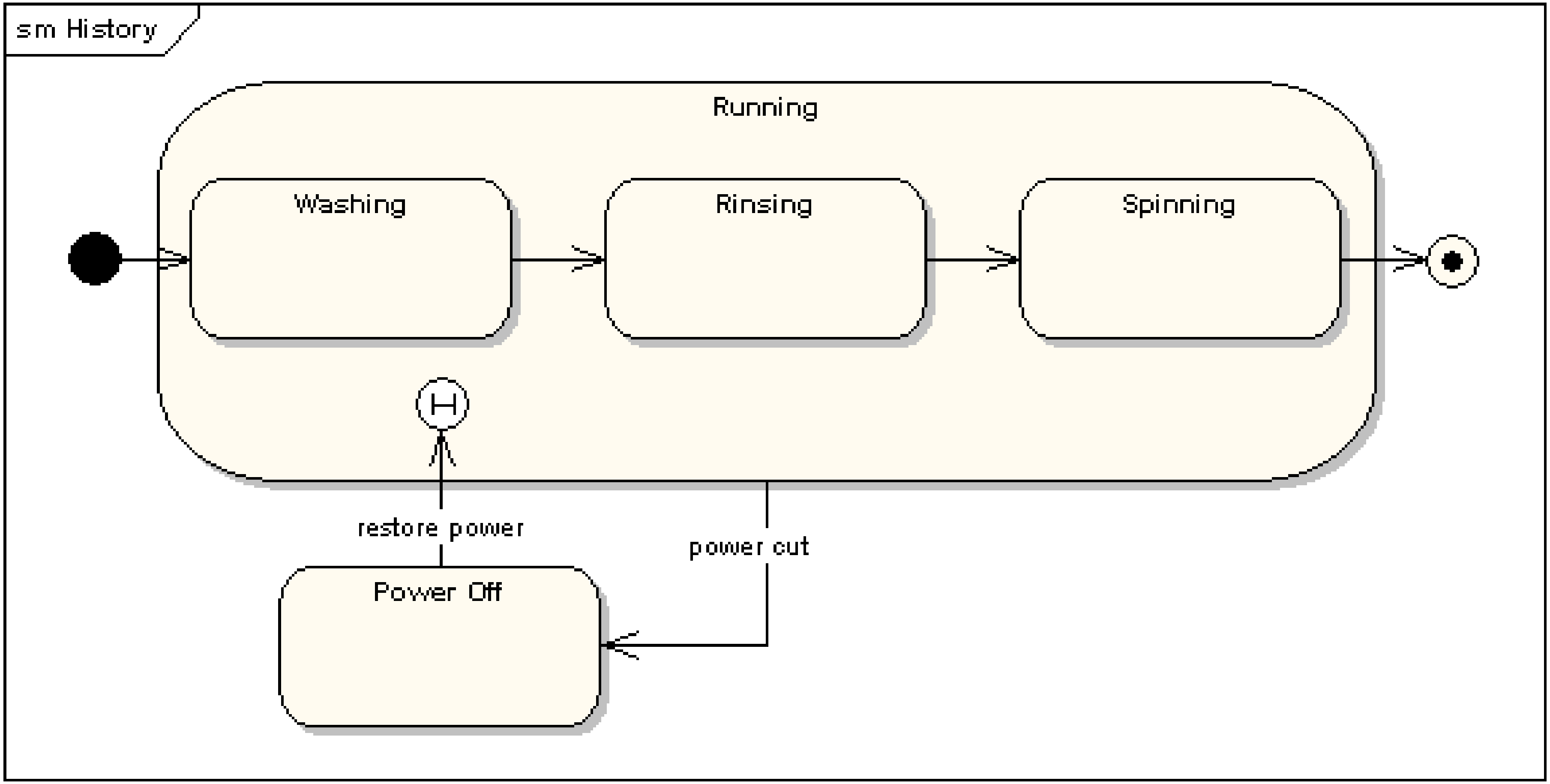
State Machine Diagrams

- ***Terminate Pseudo-State***
  - Entering a terminate pseudo-state indicates that the lifeline of the state machine has ended.
  - A terminate pseudo-state is notated as a cross.



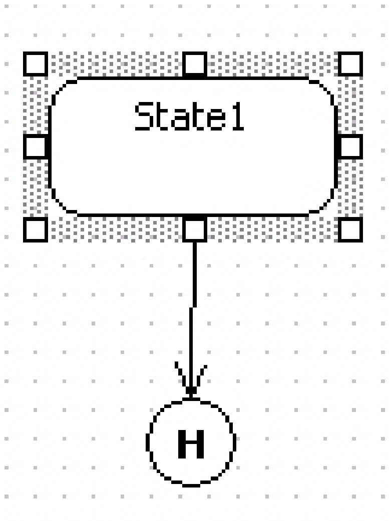
State Machine Diagrams

- ***History States***
  - A history state is used to remember the previous state of a state machine when it was interrupted.
  - The following diagram illustrates the use of history states. The example is a state machine belonging to a washing machine.

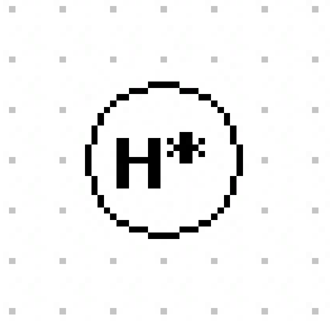

sm History
Running
Washing
Rinsing
Spinning
H
restore power
power cut
Power Off

State Machine Diagrams

- **SHALLOW HISTORY**
  - Shallow history restores the state within the enclosing composite state that was active just before the enclosing state was last exited.
  - Does not restore any substates of the last active state.

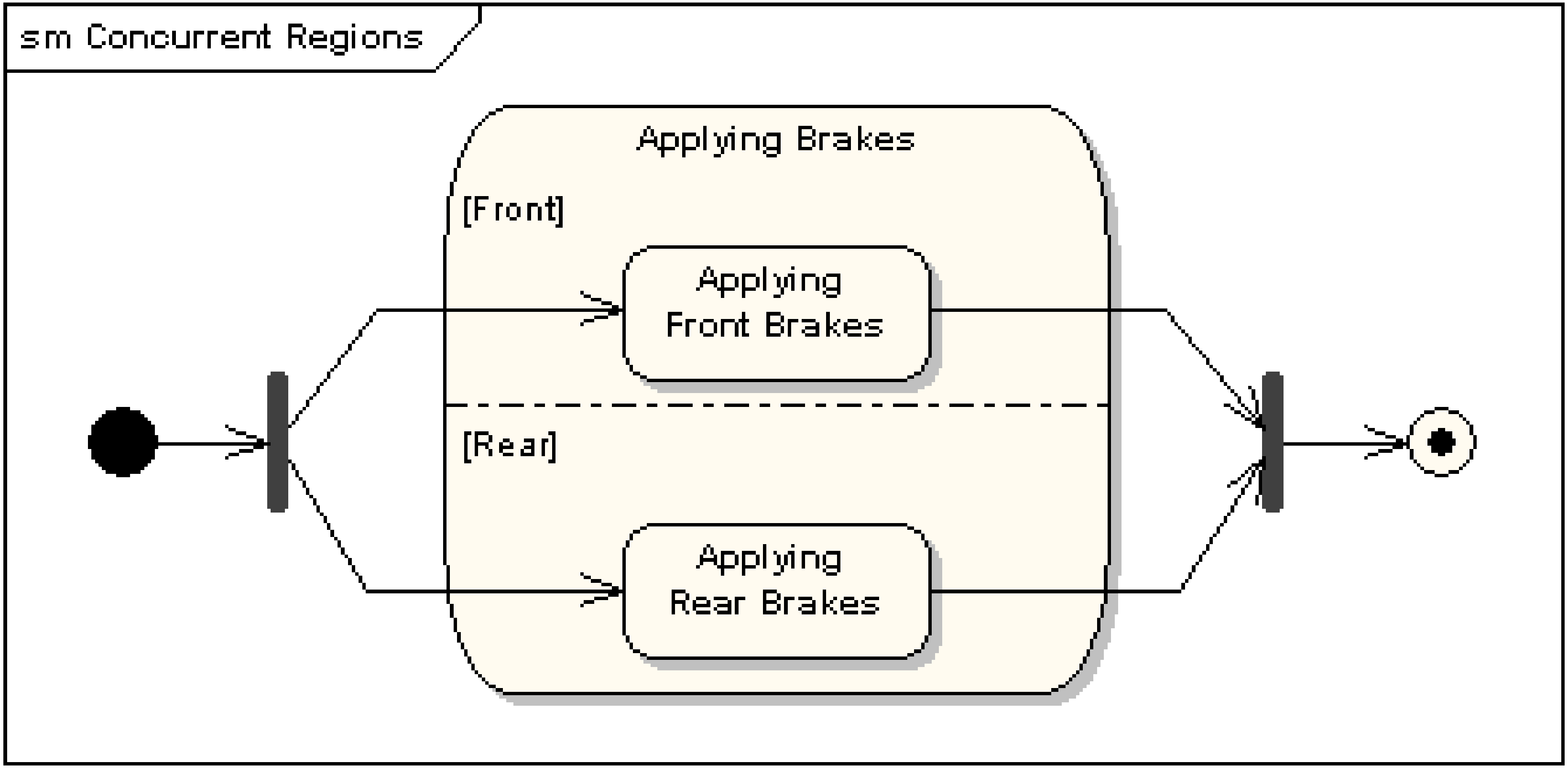


- **DEEP HISTORY**
  - Deep history restores the full state configuration that was active just before the enclosing composite state was last exited.

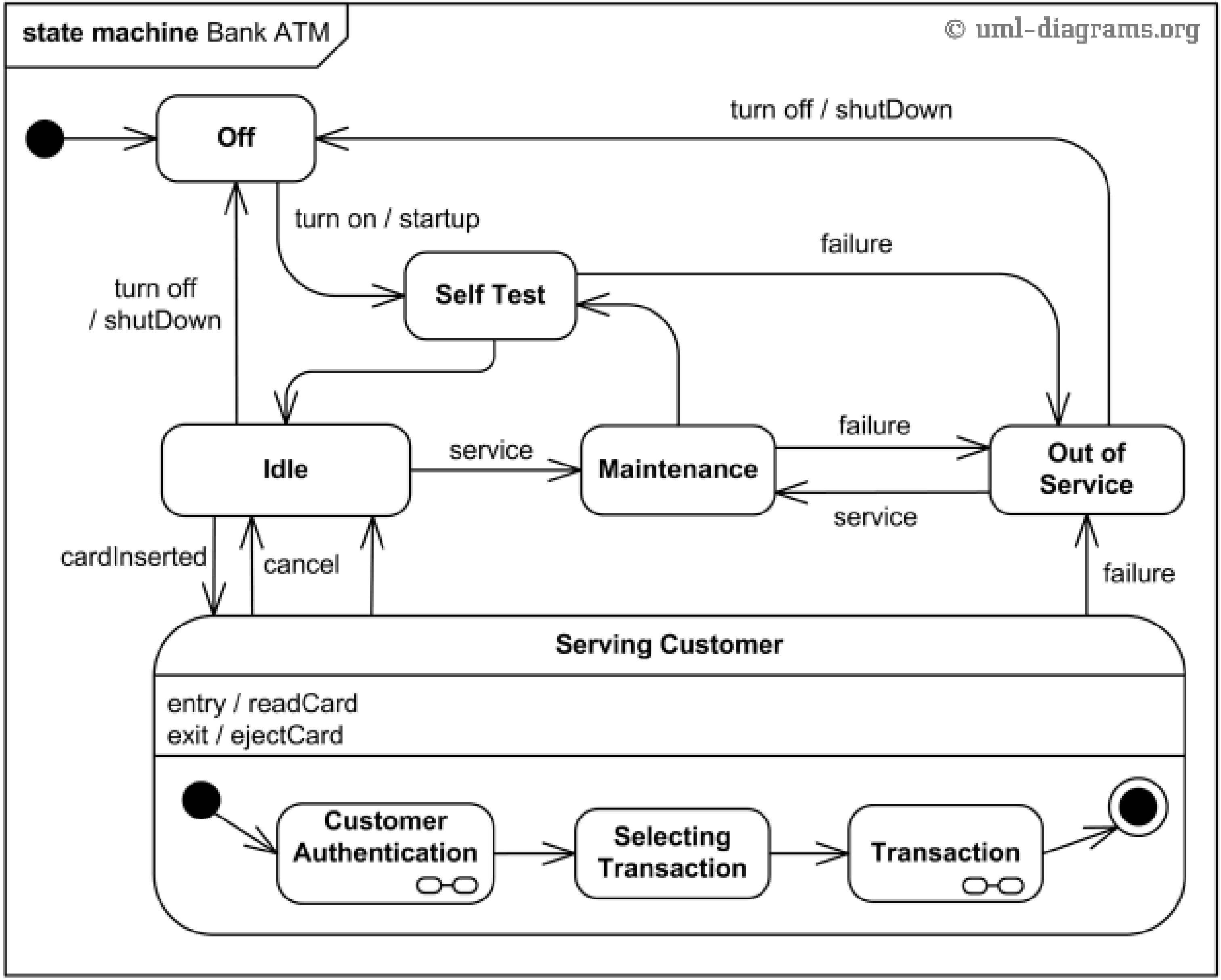


State Machine Diagrams

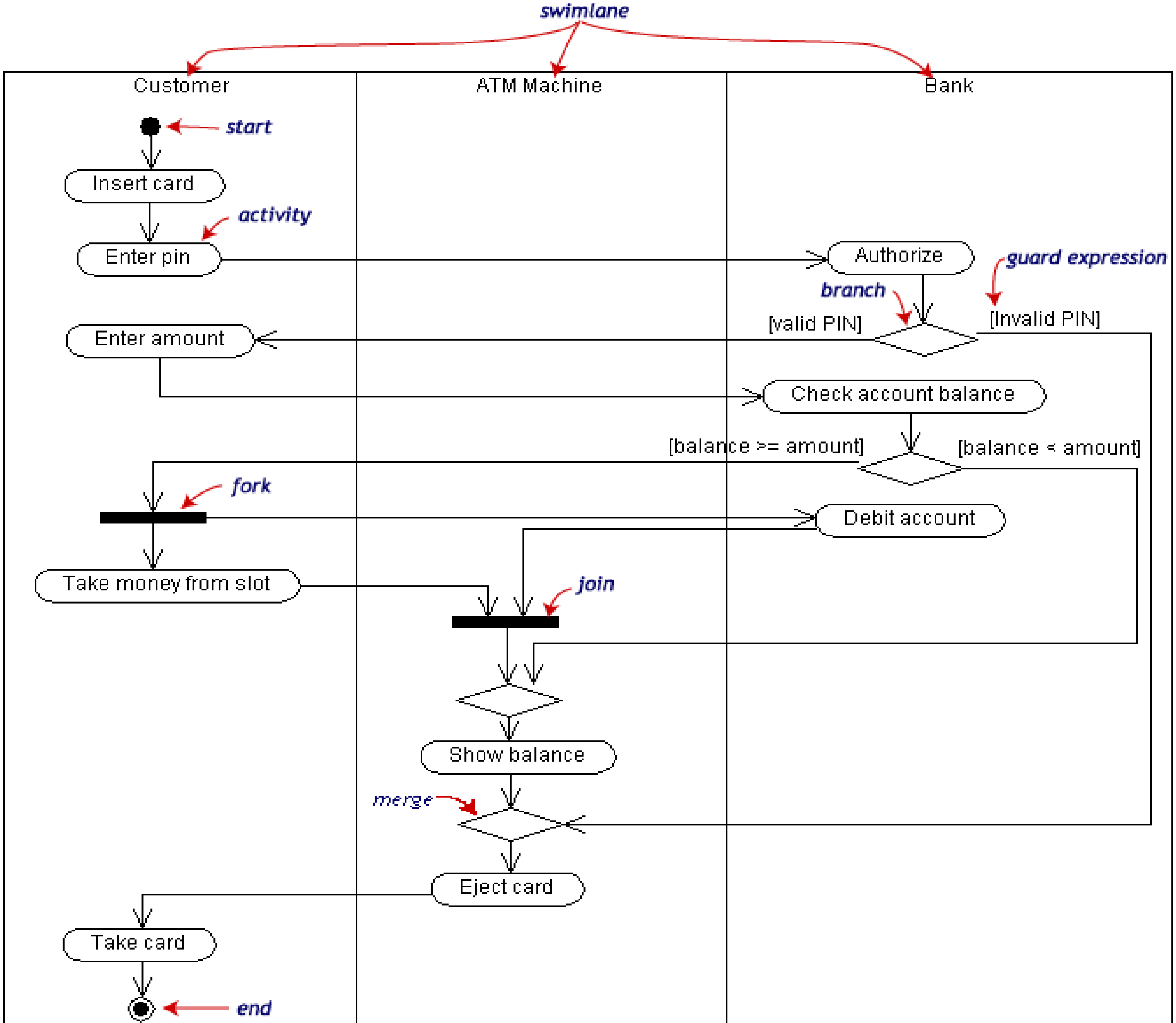
- ***Concurrent Regions***
  - A state may be divided into regions containing sub-states that exist and execute concurrently.
  - Within the state "Applying Brakes", the front and rear brakes will be operating simultaneously and independently.
  - Use fork and join pseudo-states, rather than choice and merge pseudo-states.
  - These symbols are used to synchronize the concurrent threads.


sm Concurrent Regions
Applying Brakes
[Front]
Applying
Front Brakes
[Rear]
Applying
Rear Brakes

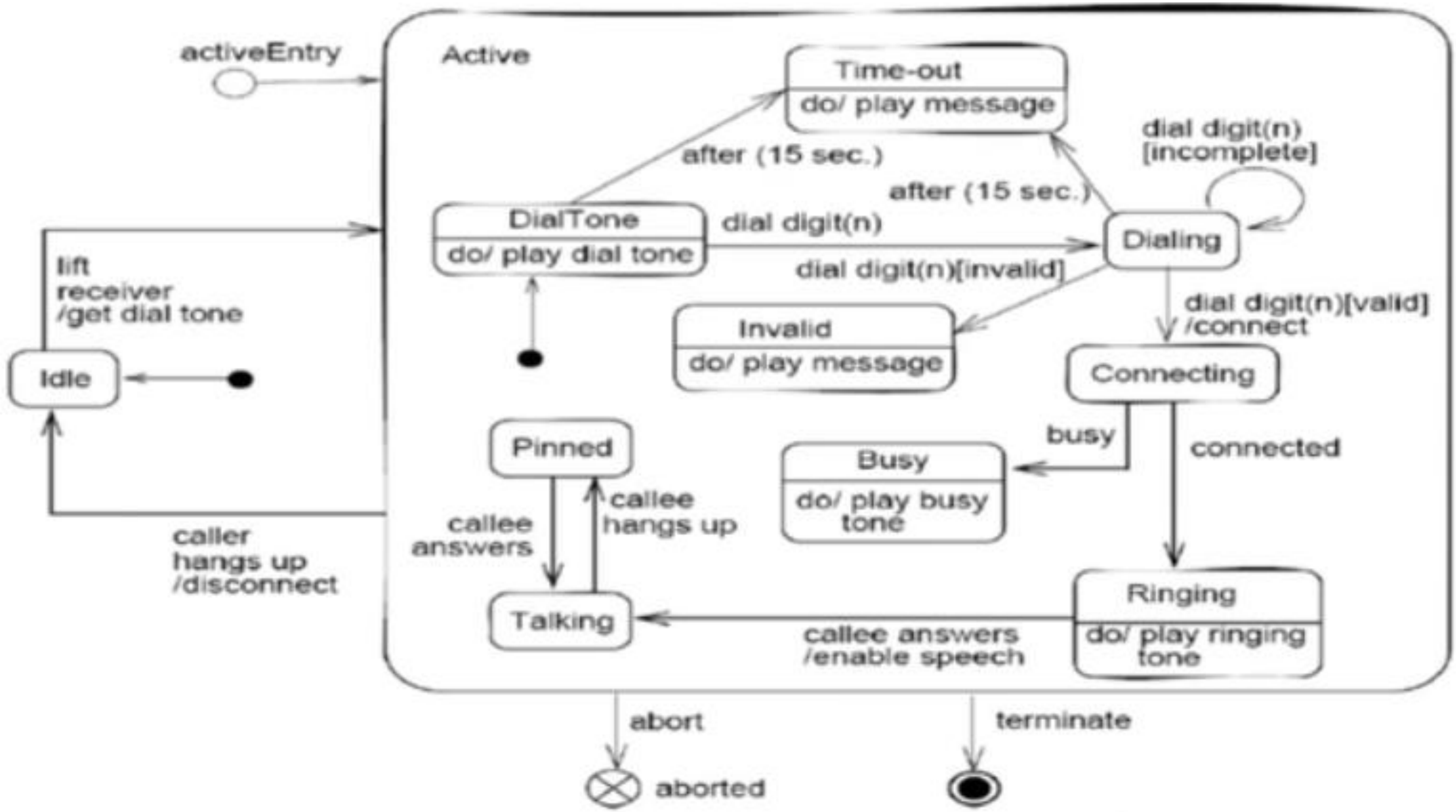
State chart for Bank ATM



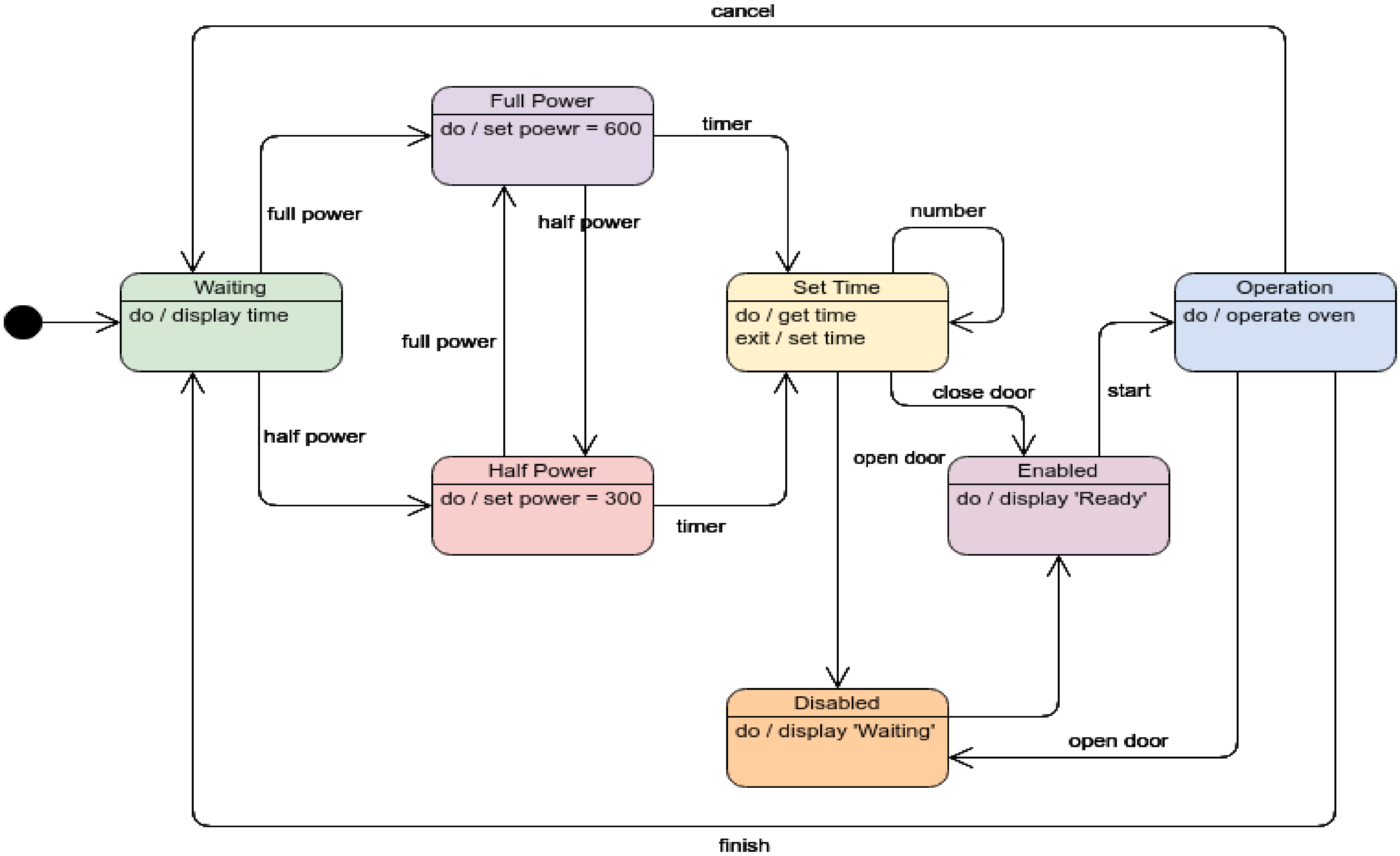
Activity diagram for Bank ATM



State chart for a Phone Line object



State Machine Diagram Example: Oven



Question 1

- Draw a UML State Machine diagram to visualise the states of a torch that is operated with an on-off switch.
- When the torch is switched on, it shines yellow.
- It can only be switched off if the switch is turned off while the torch is shining red.

Question 1

- The torch will start shining red when it is switched on while it was shining yellow.
- If the torch is switched off when shining yellow, it start shining white.
- If it is turned off while shining white, it starts shining yellow.

Question 2

- Create a state diagram to model the operation of a simple cell phone.
- The cell phone has an on/off switch. It has a numeric keypad that produces a keypad press event with a digit as its argument.
- The phone has a three-way switch that is set to ringing, vibrating, or both; it determines the action of the phone when a call comes in.
- The phone also has an action button that
  - (a) initiates a call when seven digits have been entered,

Question 2

  - (b) answers a call when the phone is ringing or vibrating, and
  - (c) terminates a call (hangs up) if a call is in progress.
- If the action button is pressed when fewer than seven digits have been entered and the phone is not ringing, the digits are erased (this is how dialing mistakes are corrected).
- Finally, the phone has a display that shows the digits that have been pressed so far, if any.
- Use at least one composite state in your model