Object Oriented Methods with UML

Lecture -5

Introduction to Activity and state Diagram

Prepared By

Dr.A.Bazila Banu

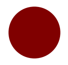
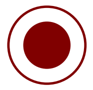
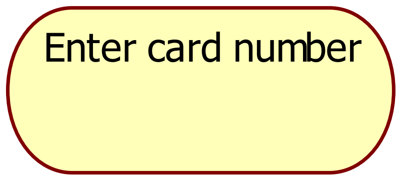
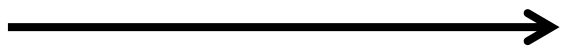
Lecturer/CSSE

- Topics (19/04/2016)
  - Activity Diagram
  - State Diagram

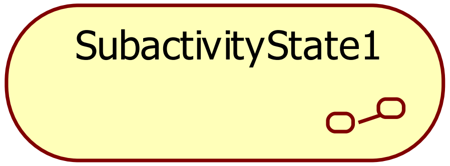
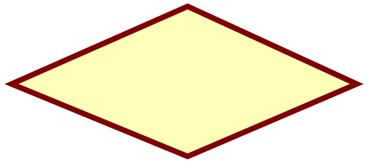
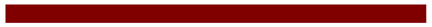
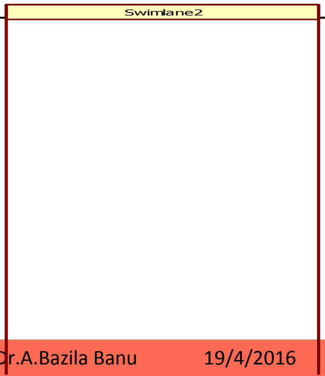
What is an Activity Diagram?

- Activity diagrams represent the dynamic (behavioral) view of a system
- Activity diagrams are typically used for business (transaction) process modeling and modeling the logic captured by a single use-case or usage scenario
- Activity diagram is used to represent the flow across use cases or to represent flow within a particular use case
- UML activity diagrams are the object oriented equivalent of flow chart and data flow diagrams in function-oriented design approach
- Activity diagram contains activities, transitions between activities, decision points, synchronization bars, swim lanes and many more…

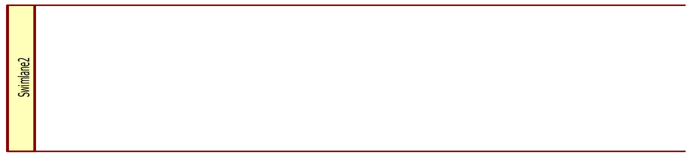
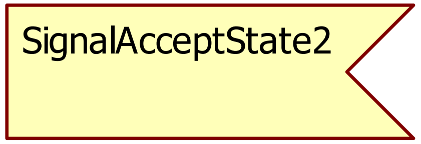
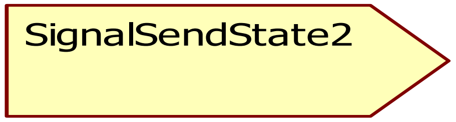
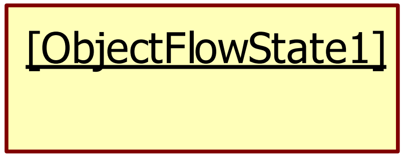
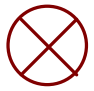
Basic components in Activity Diagram

Component	Notation
Initial node The filled circle is the starting point of the diagram.	
Final node The filled circle with a boarder is the ending point. An activity diagram can have zero or more activity final state.	
Activity The rounded circle represents activities that occur.	
Flow/ edge The arrows in the diagram. No label is necessary	

Basic components in Activity Diagram

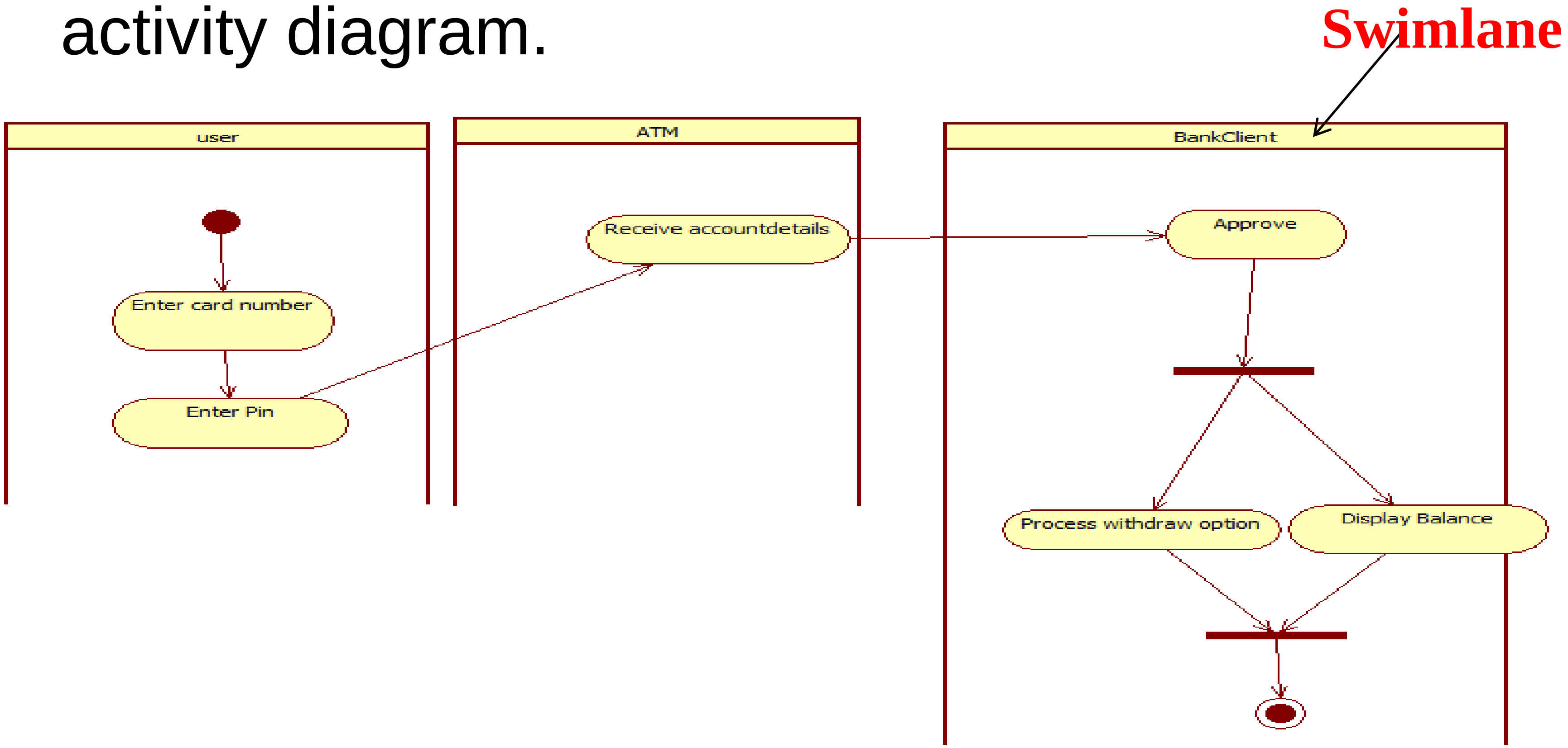
Component	Notation
Sub Activity	
Decision Box	
Synchronization	
Swimlane(Vertical)	

Basic components in Activity Diagram

Component	Notation
Swimlane(Horizontal)	
Signal Accept State	
Signal Send State	
Object Flow	
Flow Final	

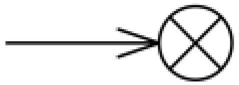
Description about components

- Swimlane
  - It is used for partitioning the children in an activity diagram.



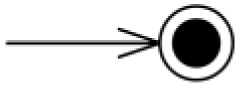
Description about component-Nodes

- ***Flow Final Node***



  - **Flow final node** is a control final node that terminates a **flow**. It destroys all tokens that arrive at it but has no effect on other flows in the activity.

- ***Activity Final Node***

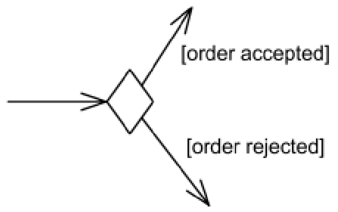


  - **Activity final node** is a control final node that stops all flows in an **activity**. Activity final was introduced in UML 2.0.

-

Description about component-Nodes

- ***Decision Node***

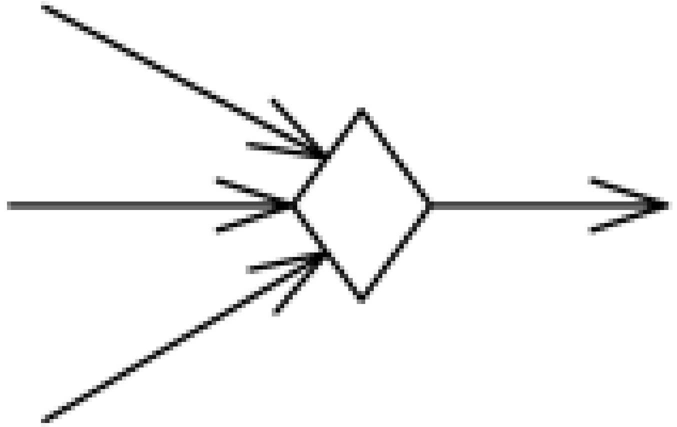


  - **Decision node** is a **control node** that accepts tokens on one or two **incoming edges** and selects one **outgoing edge** from one or more outgoing flows. Decision nodes were introduced in UML to support conditionals in activities.

-

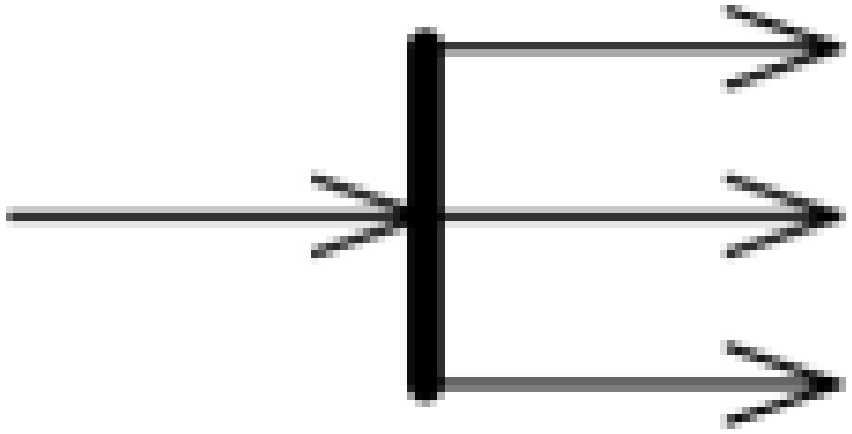
Description about component-Nodes

- ***Merge Node***
  - **Merge node** is a control node that brings together multiple incoming **alternate flows** to accept single outgoing flow. There is no joining of tokens. Merge **should not** be used to synchronize **concurrent flows**.
-



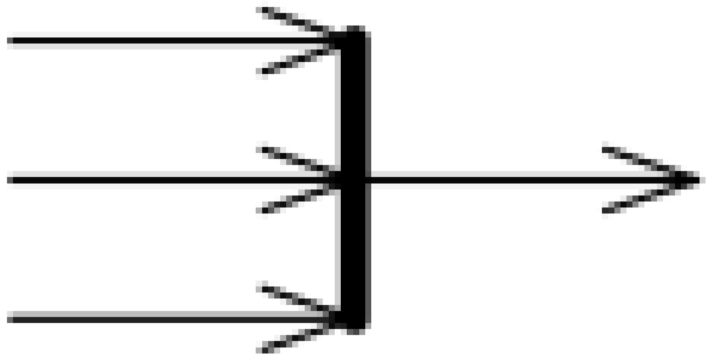
Description about component-Nodes

- ***Fork Node***
  - **Fork node** is a control node that has one incoming edge and multiple outgoing edges and is used to split incoming flow into multiple **concurrent** flows. Fork nodes are introduced to support **parallelism** in **activities**.

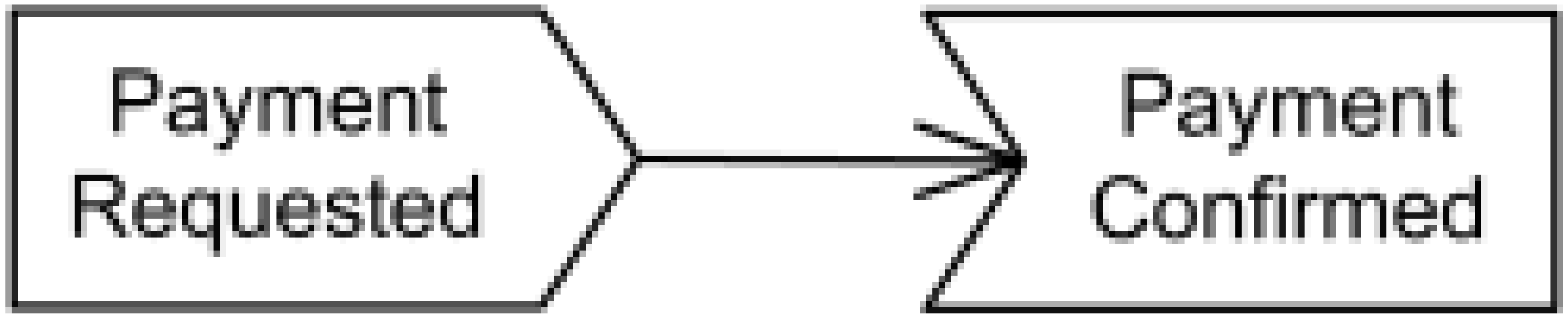


Description about component-Nodes

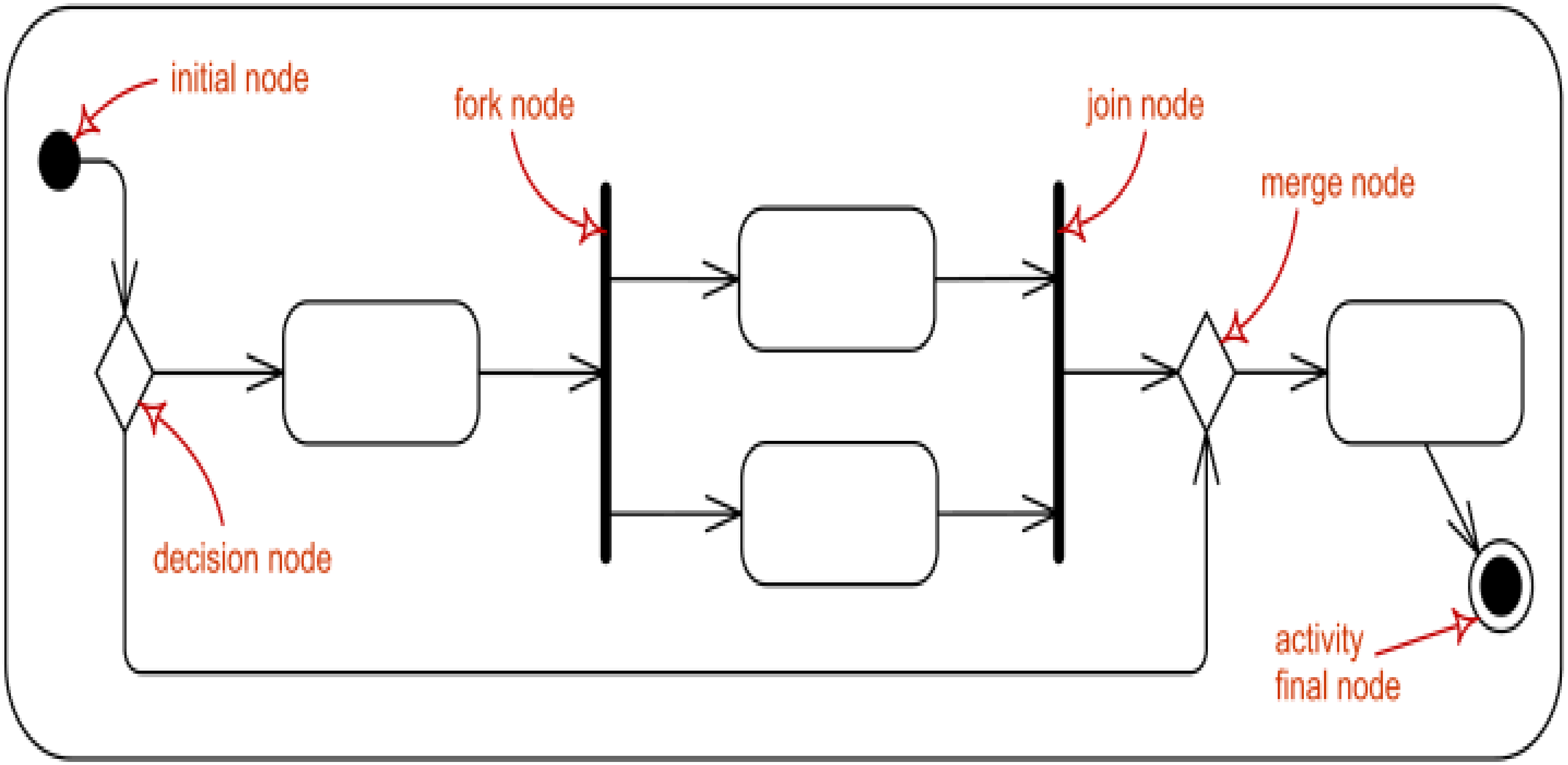
- ***Join Node***
  - **Join node** is a control node that has multiple incoming edges and one outgoing edge and is used to synchronize incoming concurrent flows. Join nodes are introduced to support parallelism in activities.
-



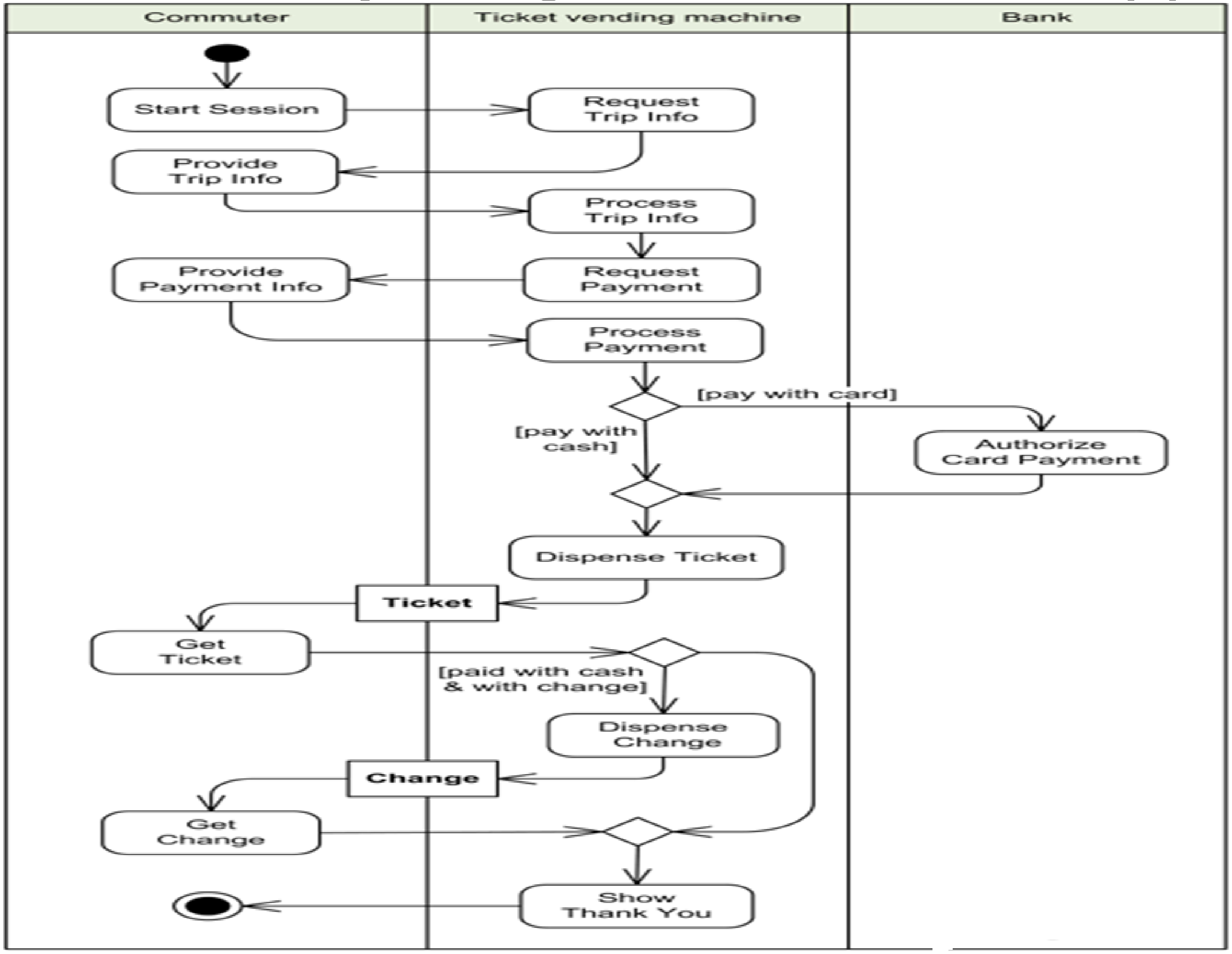
Signal Send/Accept Example



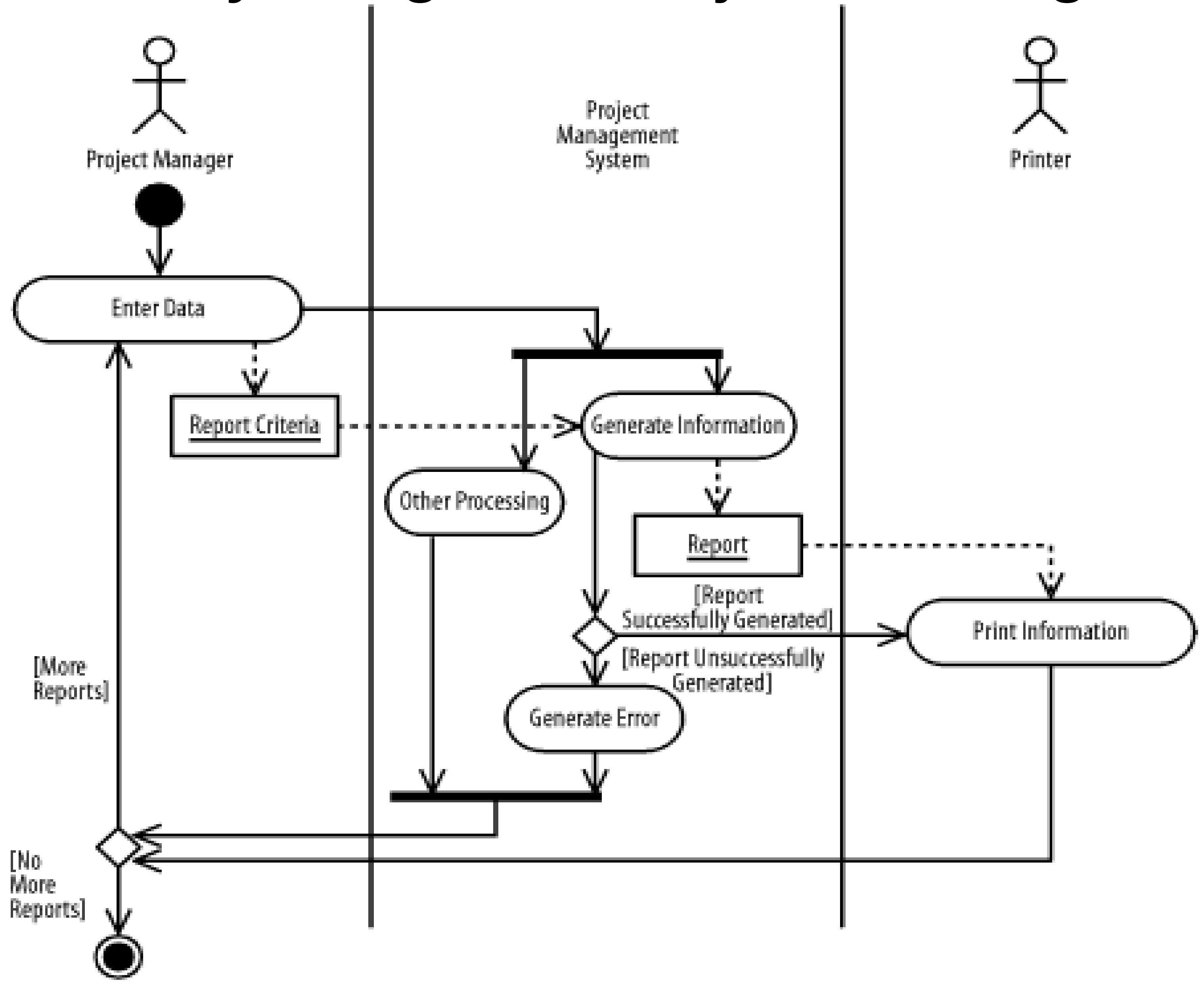
Activity Diagram (Example)



Activity Diagram –on line shopping



Activity Diagram-Project Management



State Diagram

Models the behavior of an individual object .

- Events trigger activities which, in turn, trigger actions.
- Actions are atomic.
- Actions may cause the return of a value or the change of state of an object.

Importance of State Diagram

- State chart diagrams are useful when
  - A class has an interesting or complex life cycle, e.g. classes that create or delete instances or associations
  - An instance can update its attributes in a variety of ways as it goes through a life cycle.
  - If two classes are depending on each other, in that one of them can start the other on its life-cycle, or change the order in which it goes from state to state.
  - If you find that the object's current behavior depends on what happened to it before, that is on its past history.

Terms and Concepts

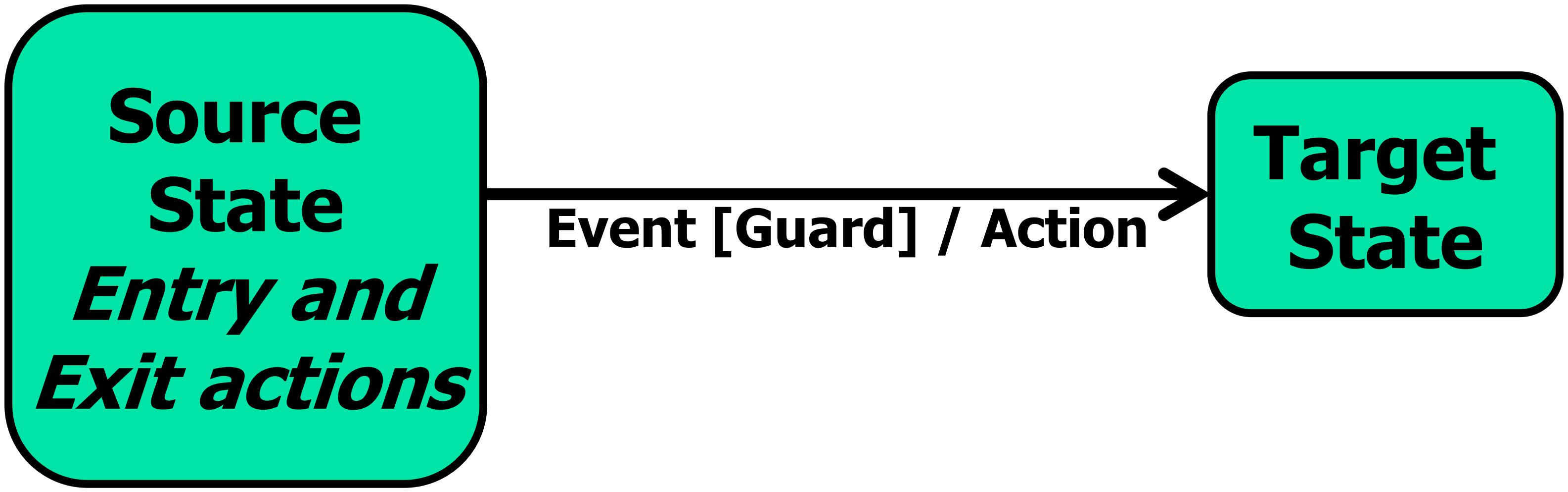
- **State**
  - A state is a condition or situation during the life of an object in which it satisfies some condition, performs some activity, or waits for some event.
  - **A state may include …**
    - Name
    - Entry/exit actions
    - Internal transitions
    - Activities
    - Substates - may sequential or concurrent
    - Deferred events (infrequently used)

Activities and Actions

- Graph whose nodes are states and whose directed arcs are transitions labeled by event names.
- Distinguish between two types of operations:
  - <u>Activity</u>: Operation that takes time to complete
    - associated with states
  - <u>Action</u>: Instantaneous operation
    - associated with events
    - associated with states (reduces drawing complexity): Entry, Exit, Internal Action
- A statechart diagram relates events and states for *one class*
  - An object model with a <u>set</u> of objects has a <u>set</u> of state diagrams

State of an Object

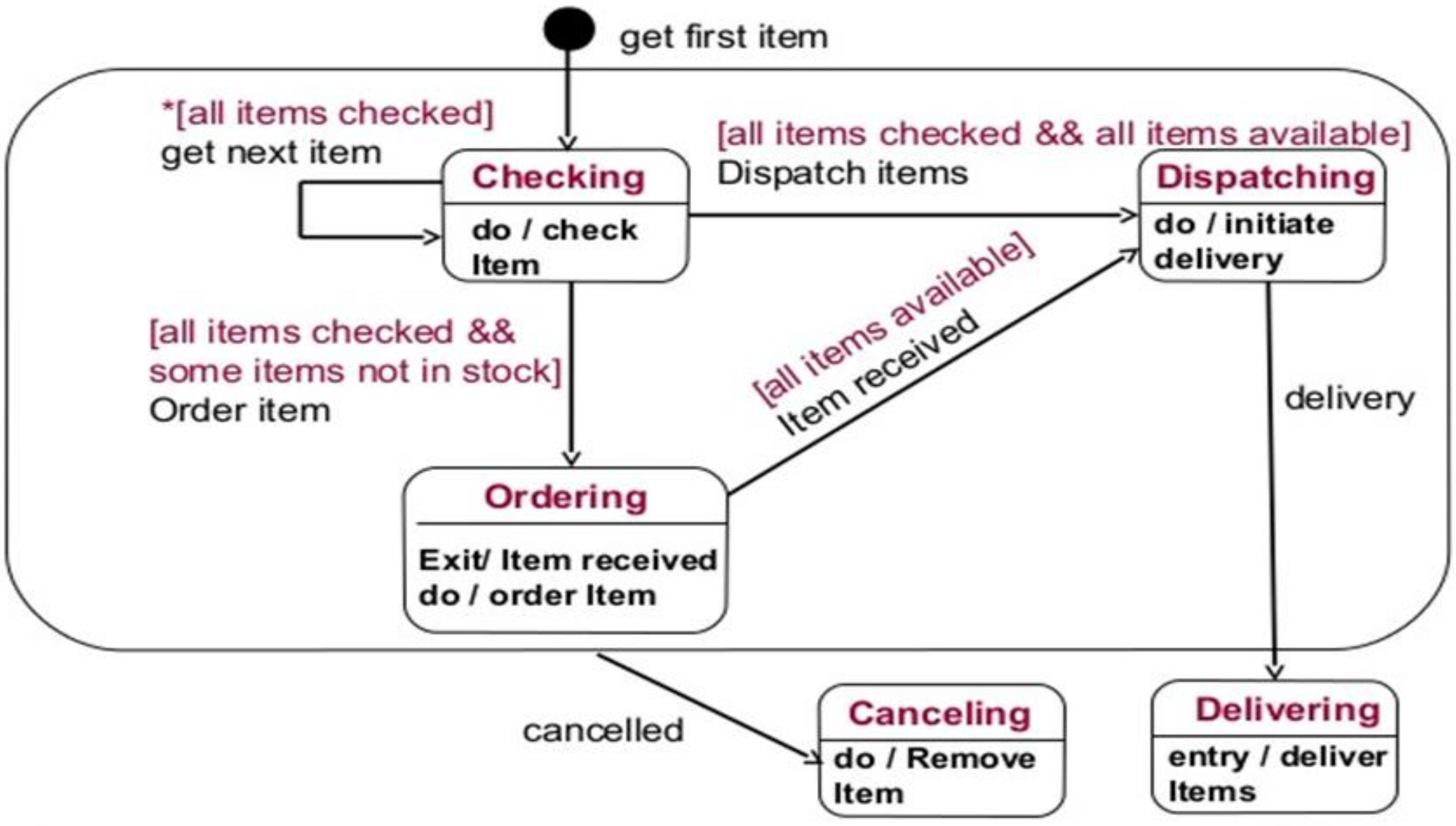
- The *state* of an object is defined by the set of values currently held by its attributes.
- At any moment in time, an object exists in a certain manner or condition, which we say is a *state*.



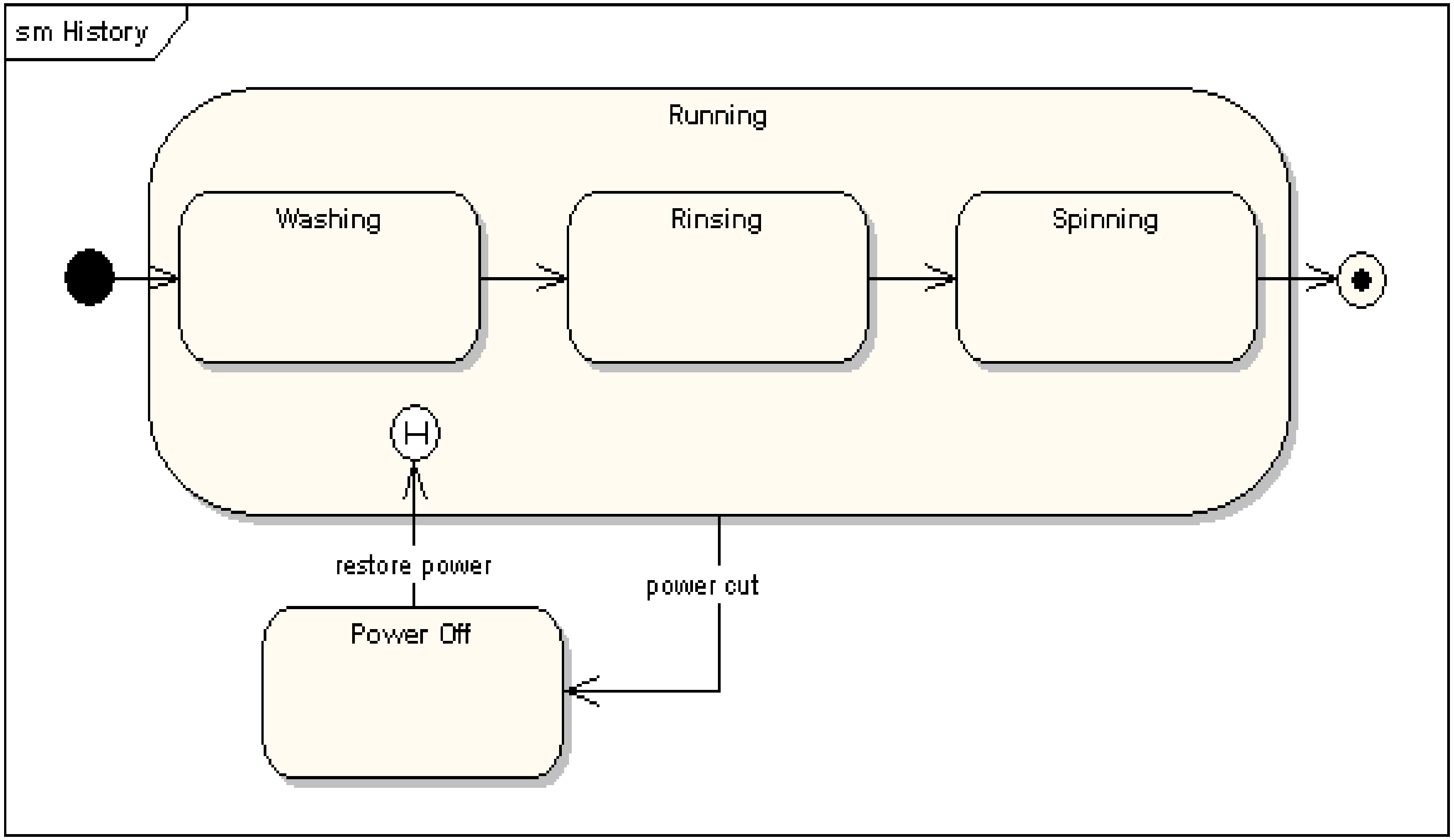
Notations of State Diagram

Component	Notation
Deep History: A history state is used to remember the previous state of a state machine when it was interrupted.	H*
Shallow History: It represents the most recent active sub state of its containing state	H
Junction: junction vertices are semantic-free vertices that are used to chain together multiple transitions.	●
Choice Point It allows splitting of transitions into multiple outgoing paths such that the decision on which path to take may be a function of the results	○

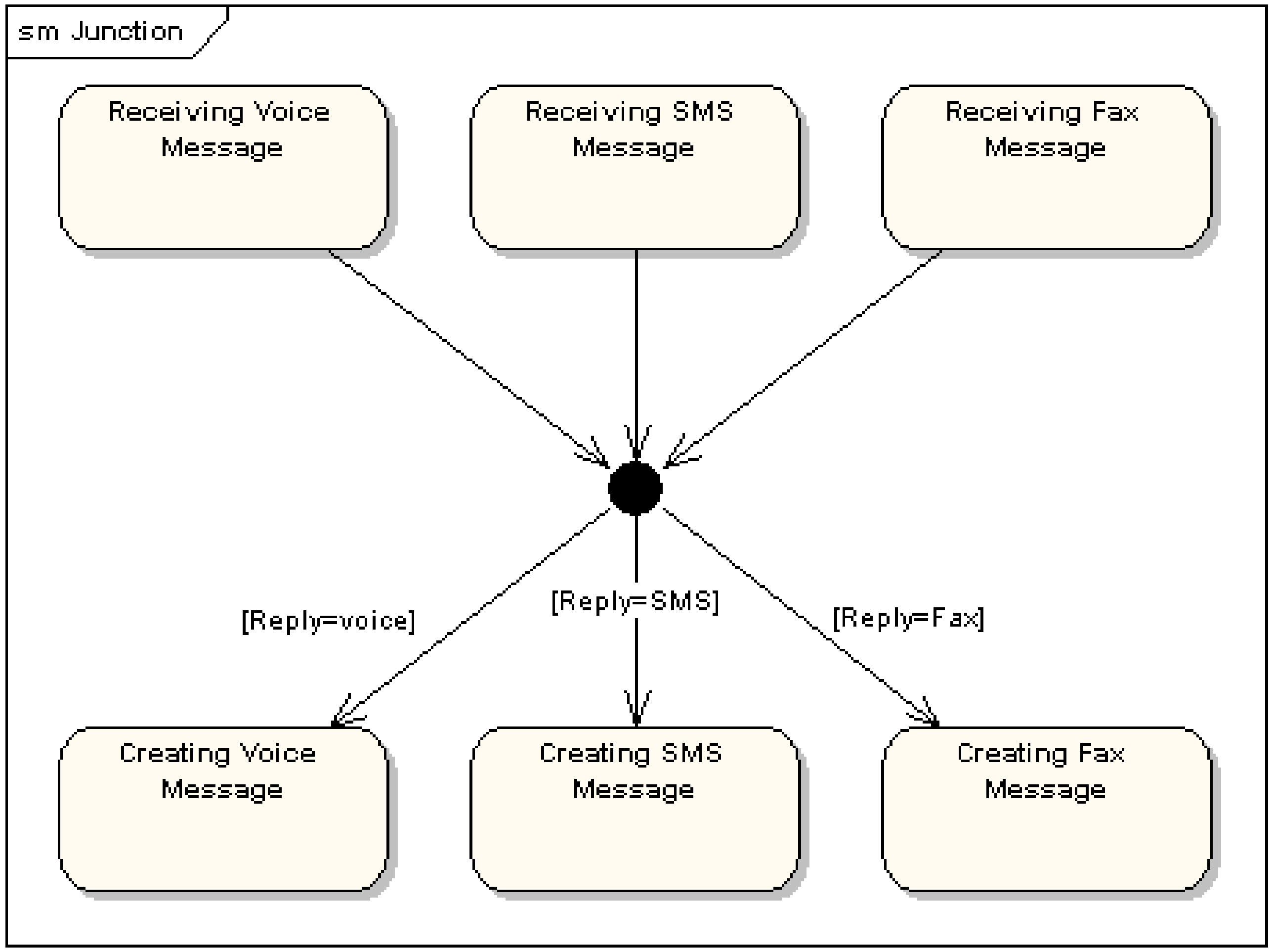
State Diagram-Example



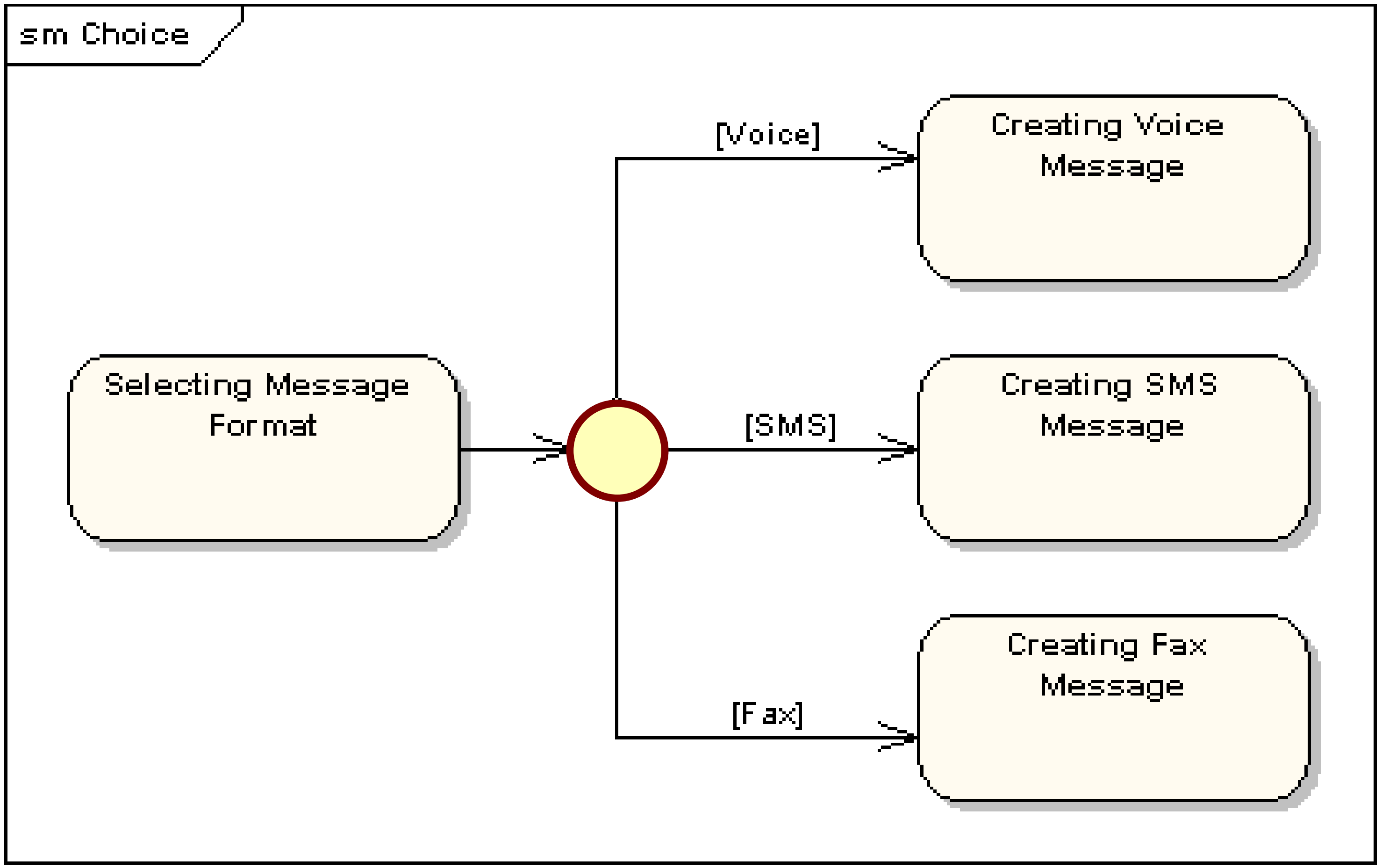
State Diagram(with Deep History) - Example



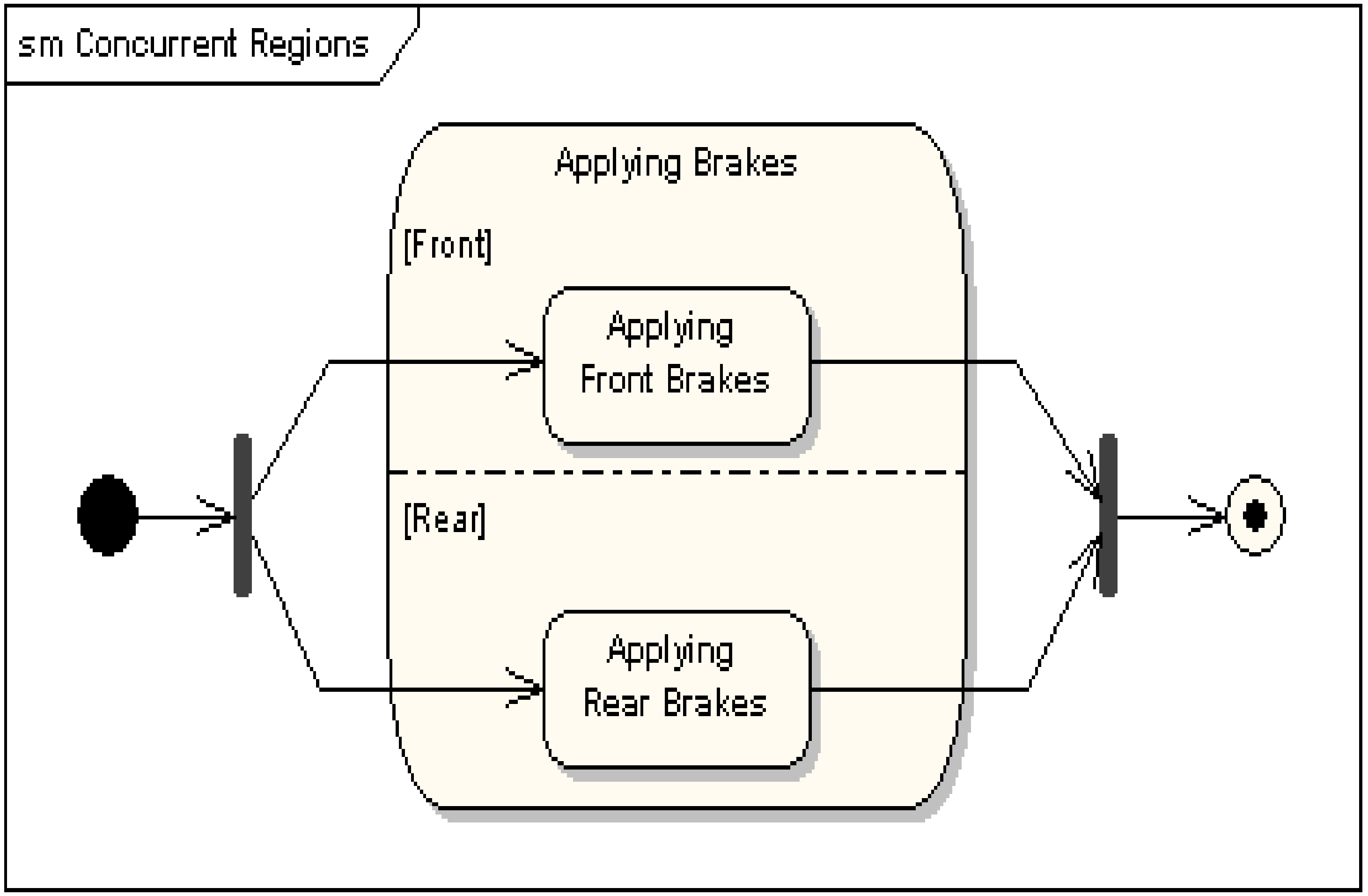
State Diagram (with Junction Point)



State Diagram (with Choice Point)



State Diagram(with Fork and Join)



References

- IBM

1)https://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep03/f_umlbasics_db.pdf

- Microsoft

2)https://msdn.microsoft.com/en-us/library/dd409465.aspx