CSCI 4050/6050
Software Engineering

Dynamic Modeling

Activity diagrams
&
Statechart Diagrams

Based on the textbook slides by Bruegge and Dutoit

Dr. Eman Saleh eman.saleh@uga.edu

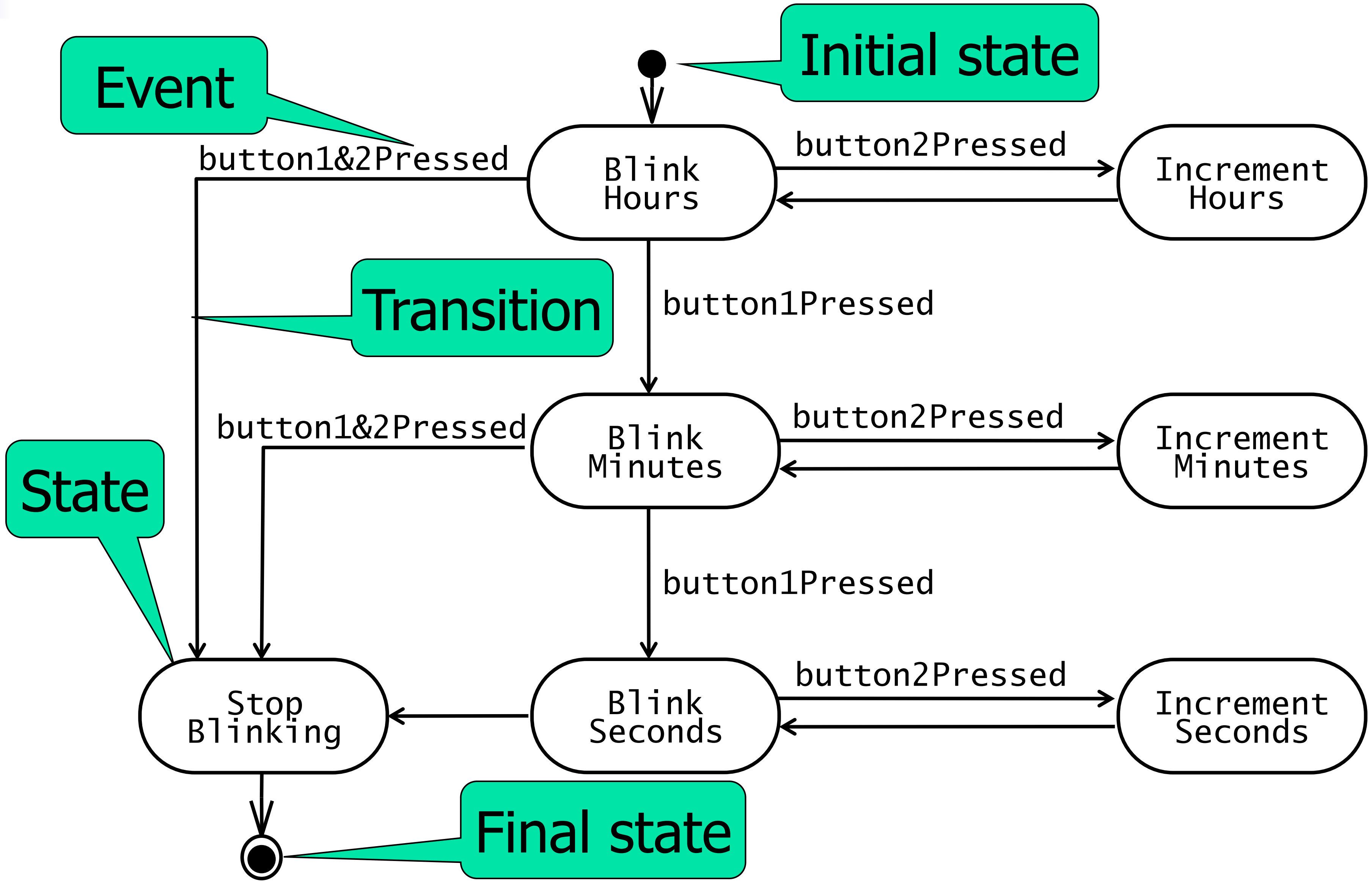
UML State and Activity Diagrams

- Statechart Diagram:
  - A state machine that describes the response of an object of a given class to the receipt of outside messages (or events).
  - Used to model the behavior of a single class across multiple use cases.
  - Similar to Finite State Machines.
- Activity Diagram:
  - A special type of state chart diagram, where all states are action states.
  - Used to model the behavior of multiple classes across multiple use cases.
  - May represent perspectives of multiple actors.
  - Used to model *workflows*.
  - Since UML 2.0, semantics is based on Petri Nets.

Statechart vs Sequence Diagram

- Statechart diagrams help to identify:
  - Changes to an individual object over time
- Sequence diagrams help to identify:
  - The temporal relationship between objects over time
  - Sequence of operations as a response to one ore more events. (Input and output messages)

UML Statechart diagram



State

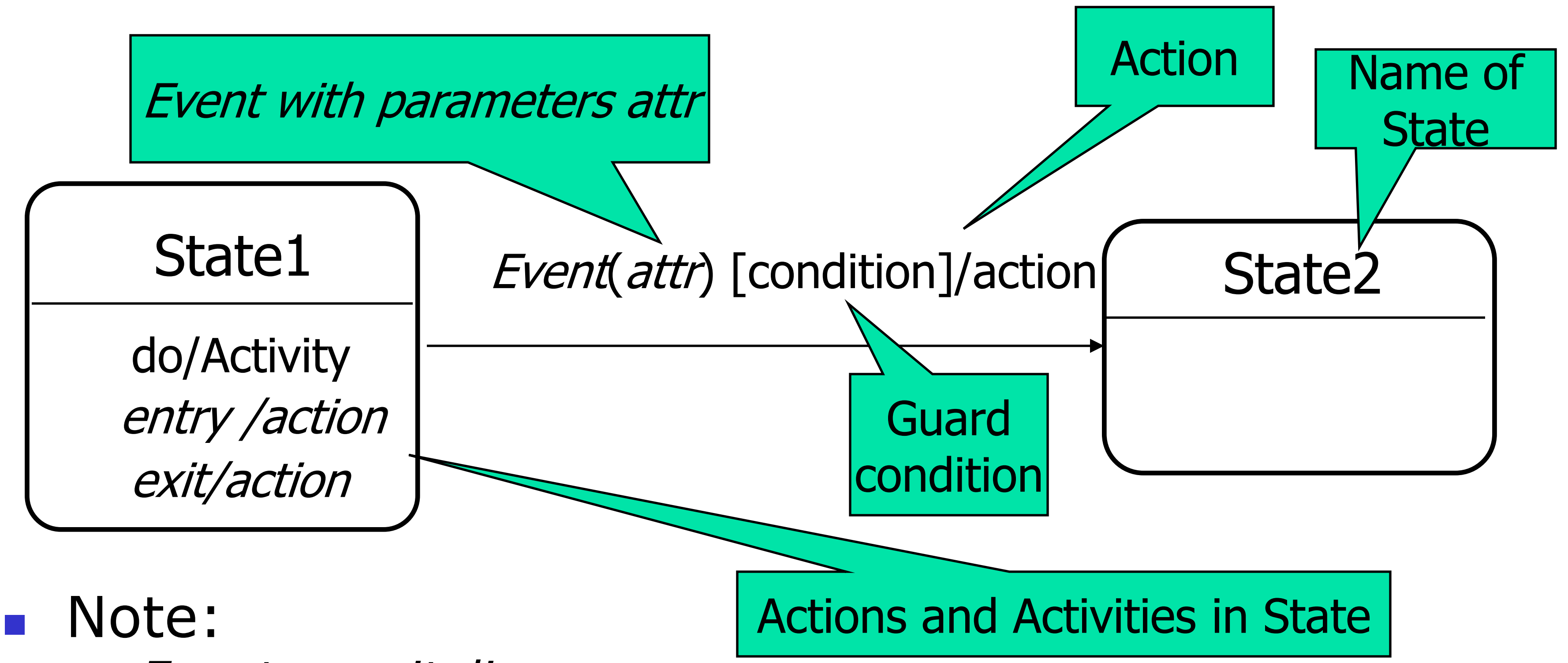
- An abstraction of the attributes of a class
  - State is the aggregation of several attributes a class
- A state is an equivalence class of all those attribute values and links that do need to be distinguished

  Examples:
  - state of an Account (over-drafted, in-credit), or
  - state of a Course Section (e.g., offered, under-full, full, cancelled)
- State has duration

In-State Operations

- We distinguish between two types of operations:
  - Activity: Operation that takes time to complete
    - associated with states
  - Action: Instantaneous operation
    - associated with events
- A statechart diagram relates events and states for one class only, but possibly for multiple use cases.
- An object model with several classes with interesting behavior has *a set* of state diagrams.

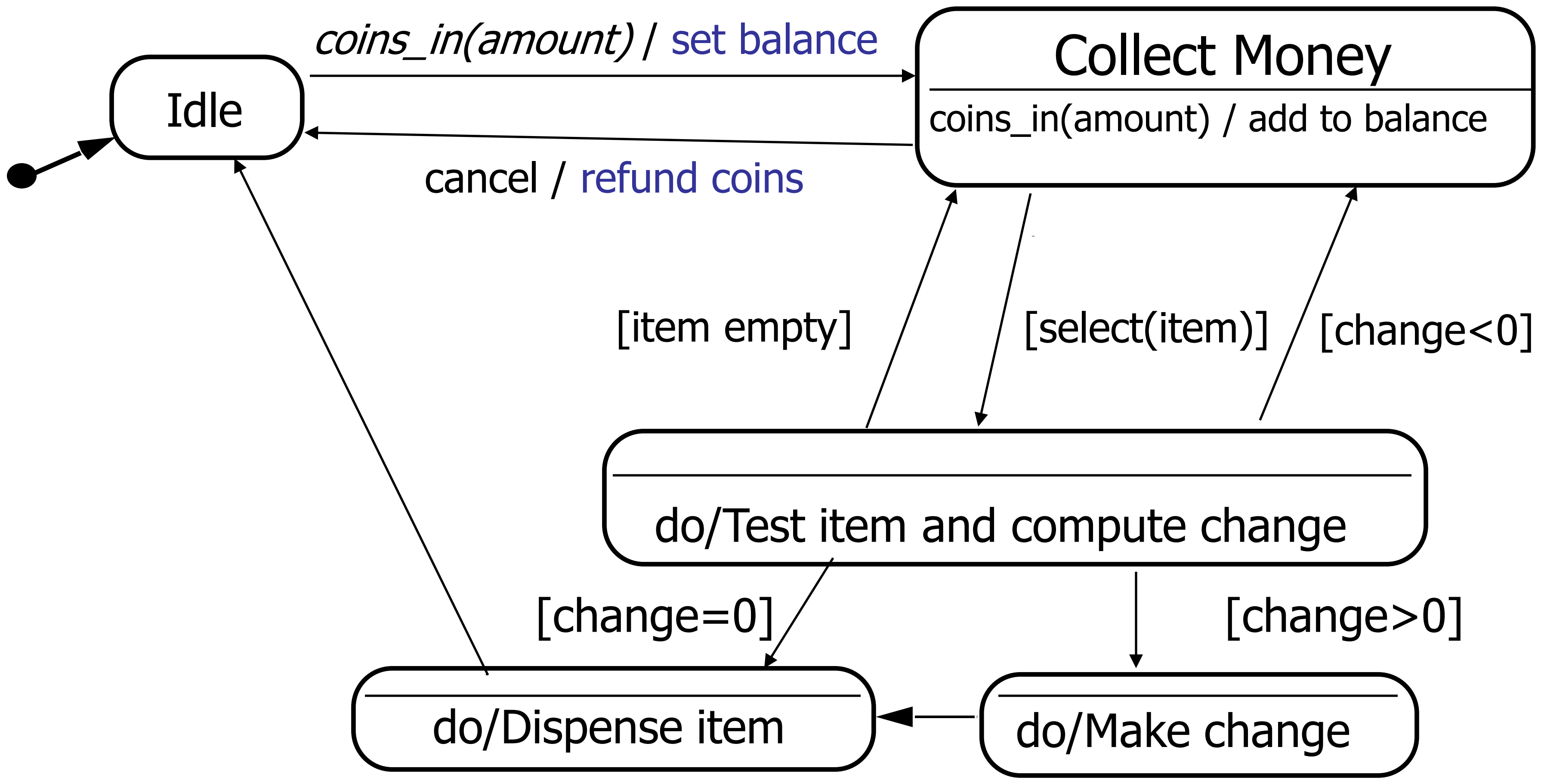
UML Statechart Diagram Notation



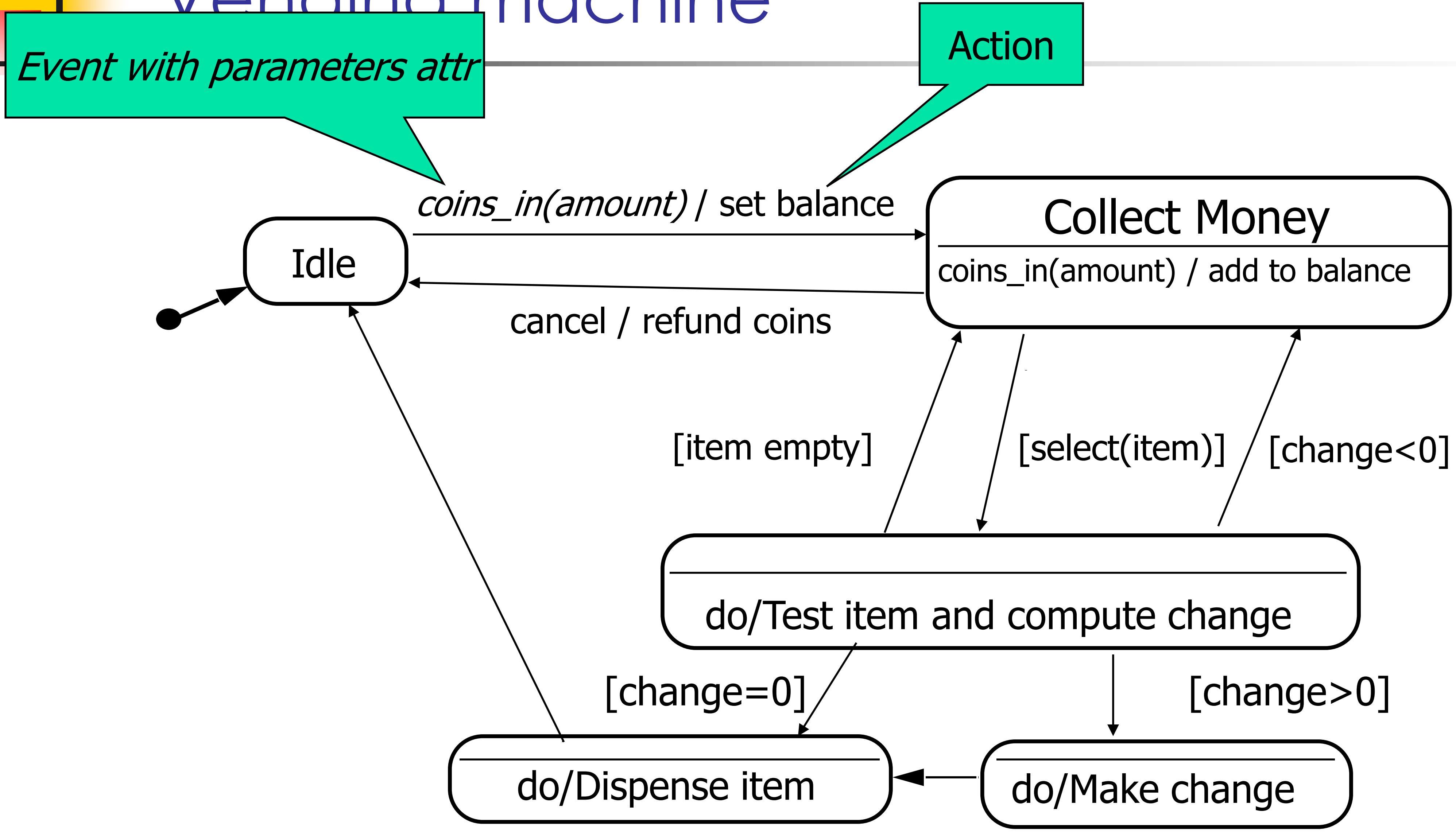
Event with parameters attr
Action
Name of State
State1
do/Activity
entry /action
exit/action
Event(*attr*) [condition]/action
State2
Guard condition
Actions and Activities in State

- Note:
  - *Events are italics*
  - Conditions are enclosed with brackets: []
  - Actions and activities are prefixed with a slash /
- Notation is based on work by Harel
- Added are a few object-oriented modifications.

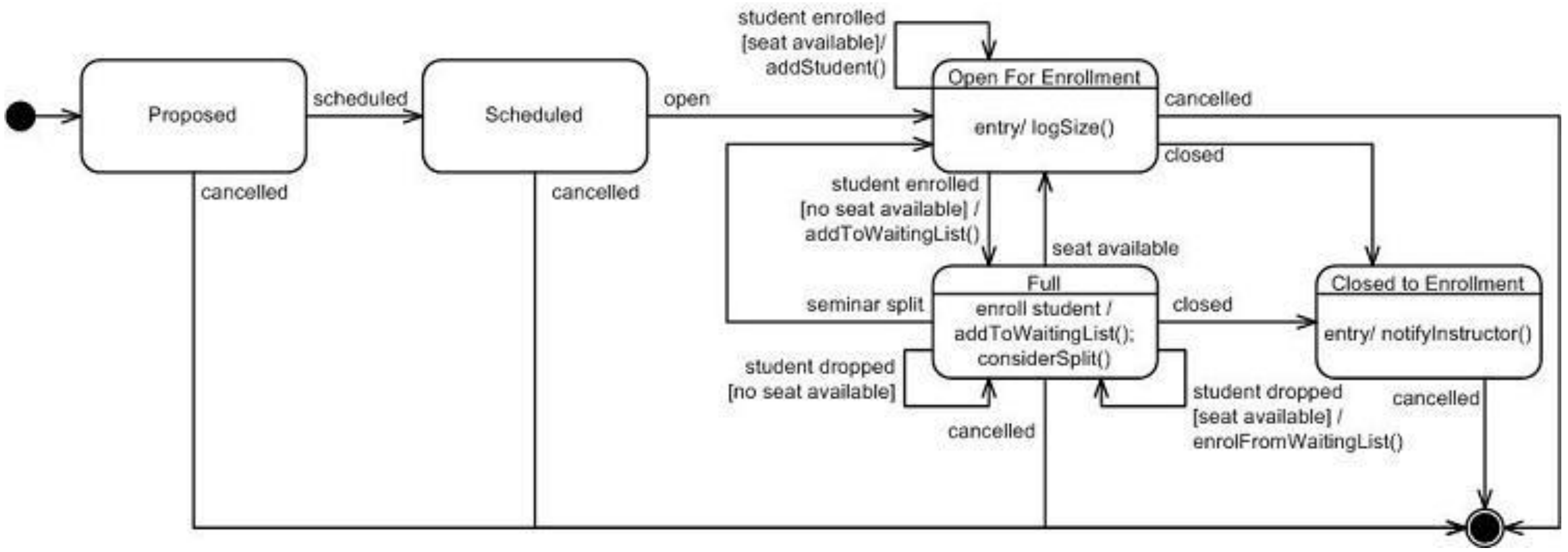
Example of a StateChart Diagram Vending machine



Example of a StateChart Diagram Vending machine



Example of a StateChart Diagram object: seminar



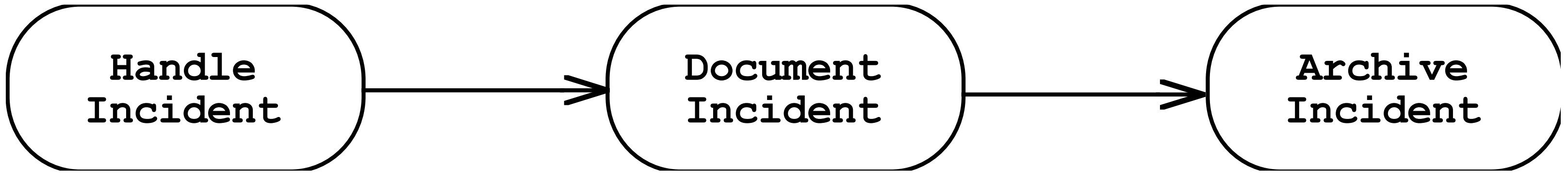
From: http://www.agilemodeling.com

Dynamic Modeling of User Interfaces

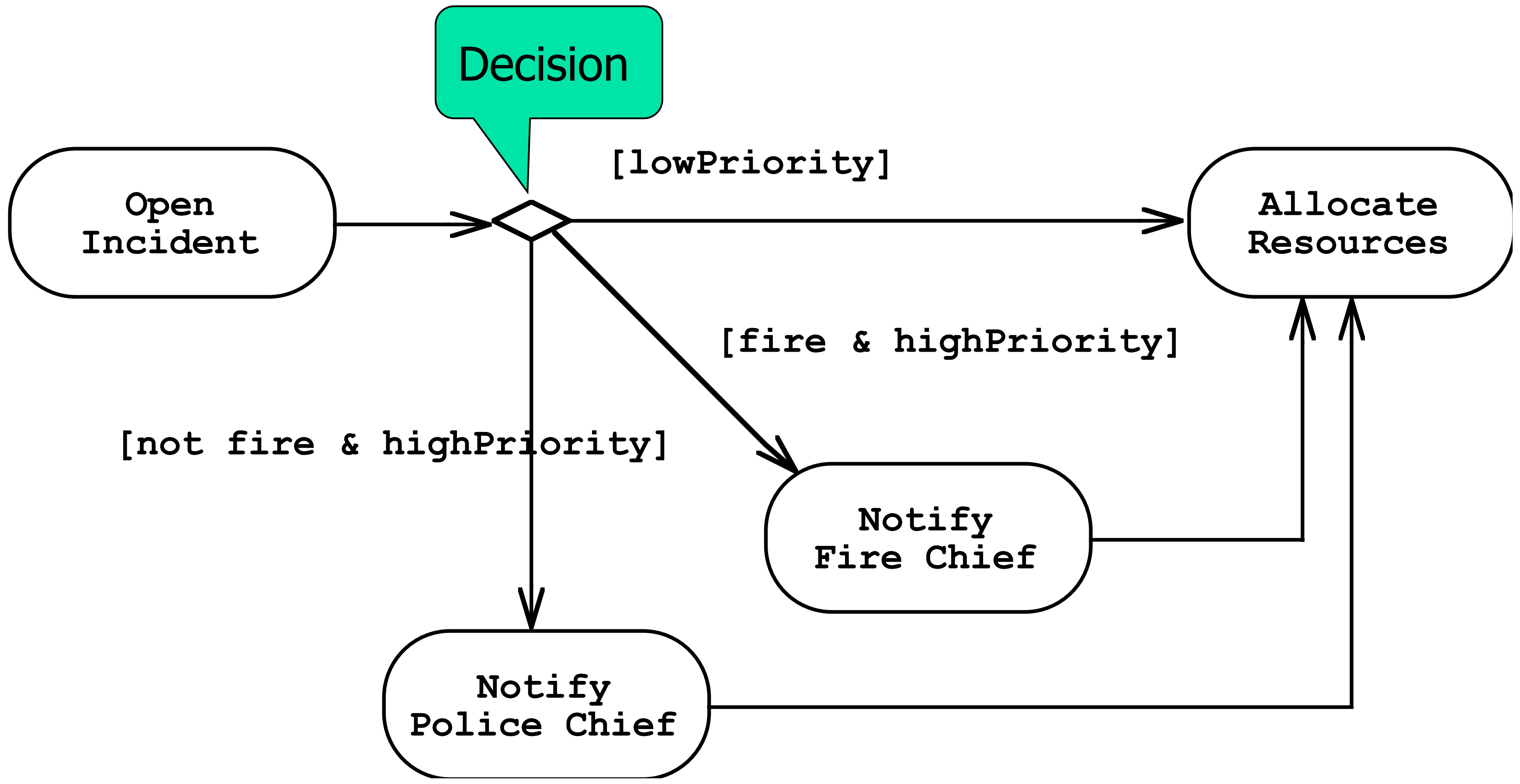
- Statechart diagrams can be used for the design of user interfaces
- States: Name of screens
- Actions or activities are shown as bullets under the screen name

UML Activity Diagrams

- An activity diagram is a special case of a state chart diagram
- The states are activities (“functions”)
- An activity diagram is useful to depict the workflow in a system, i.e. how an overall enterprise functions.
- Activity diagrams model behavior of multiple use cases and multiple classes.

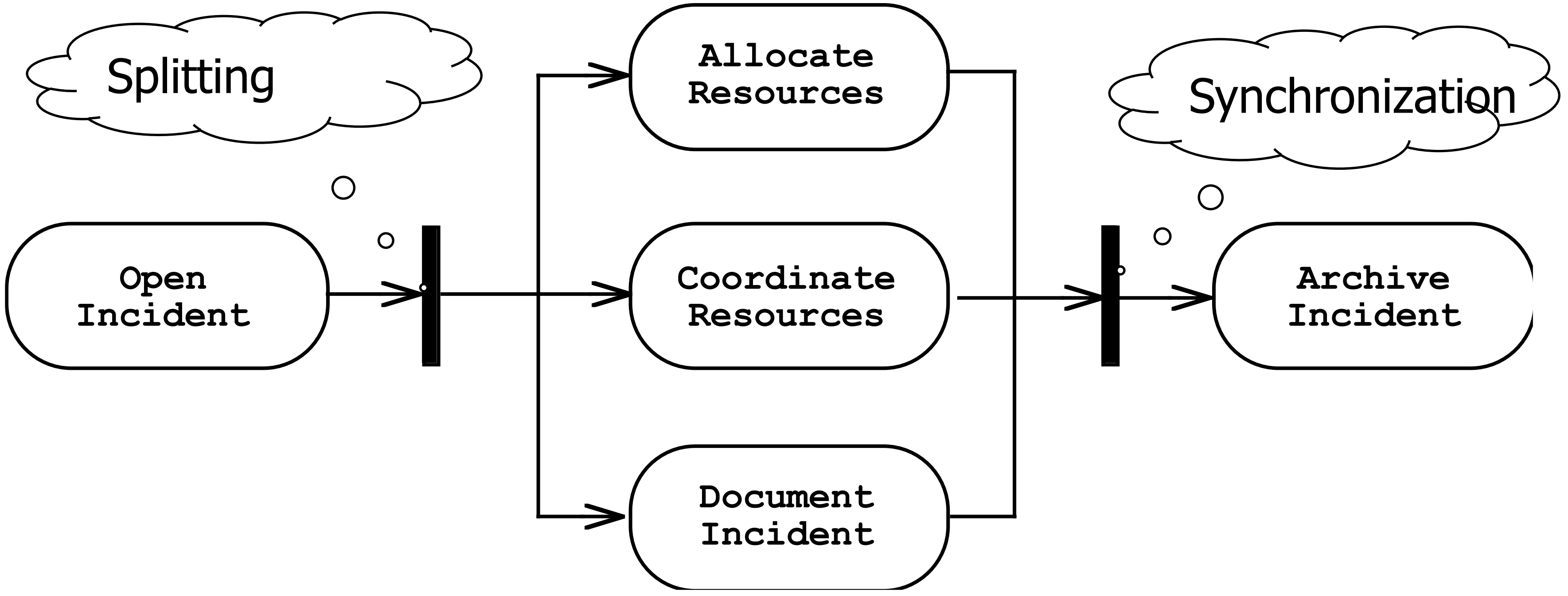


Activity Diagrams allow to model Decisions

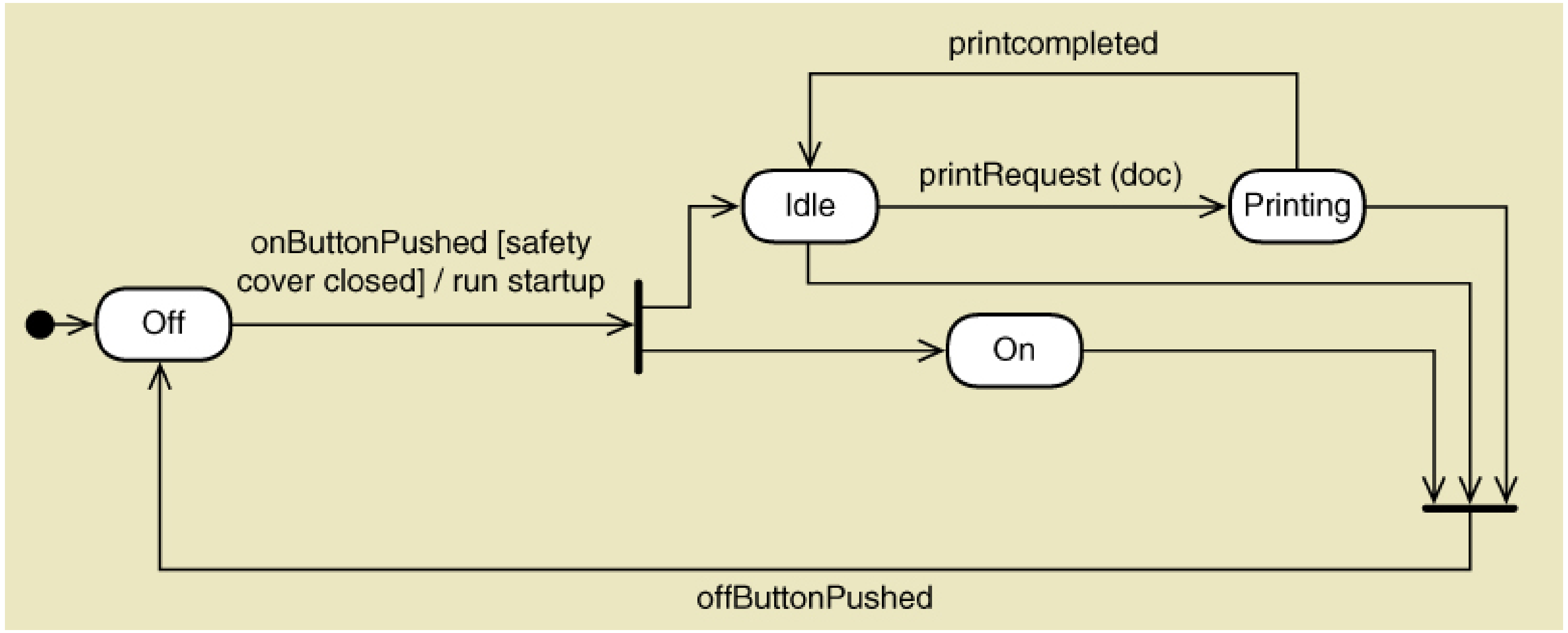


Activity Diagrams can model Concurrency

- Synchronization of multiple activities
- Splitting the flow of control into multiple threads



State Chart: Printer with Concurrent Paths



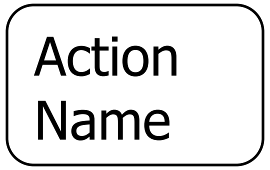
- Concurrent paths often shown by synchronization bars (same as Activity Diagram)
- Multiple exits from a state is an “OR” condition.
- Multiple exits from a synchronization bar is an “AND” condition.

Activity Diagram: Activity Nodes & Edges

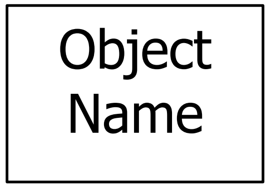
- An activity diagram consists of nodes and edges
- There are **three** types of activity nodes
  - Control nodes
  - Executable nodes
    - Most prominent: **Action**
  - Object nodes
    - E.g. a document
- An **edge** is a directed connection between nodes
  - There are two types of edges
    - Control flow edges
    - Object flow edges

Action Nodes and Object Nodes

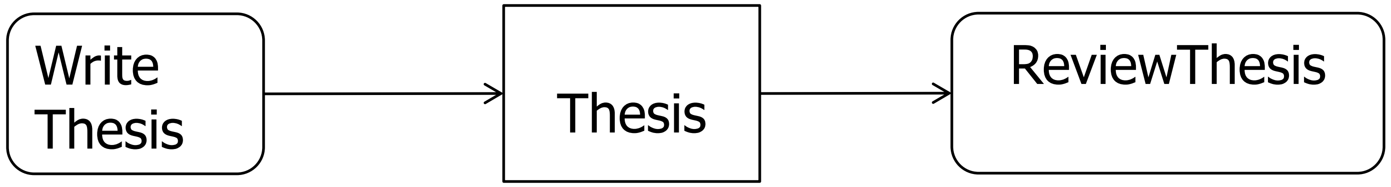
- Action Node



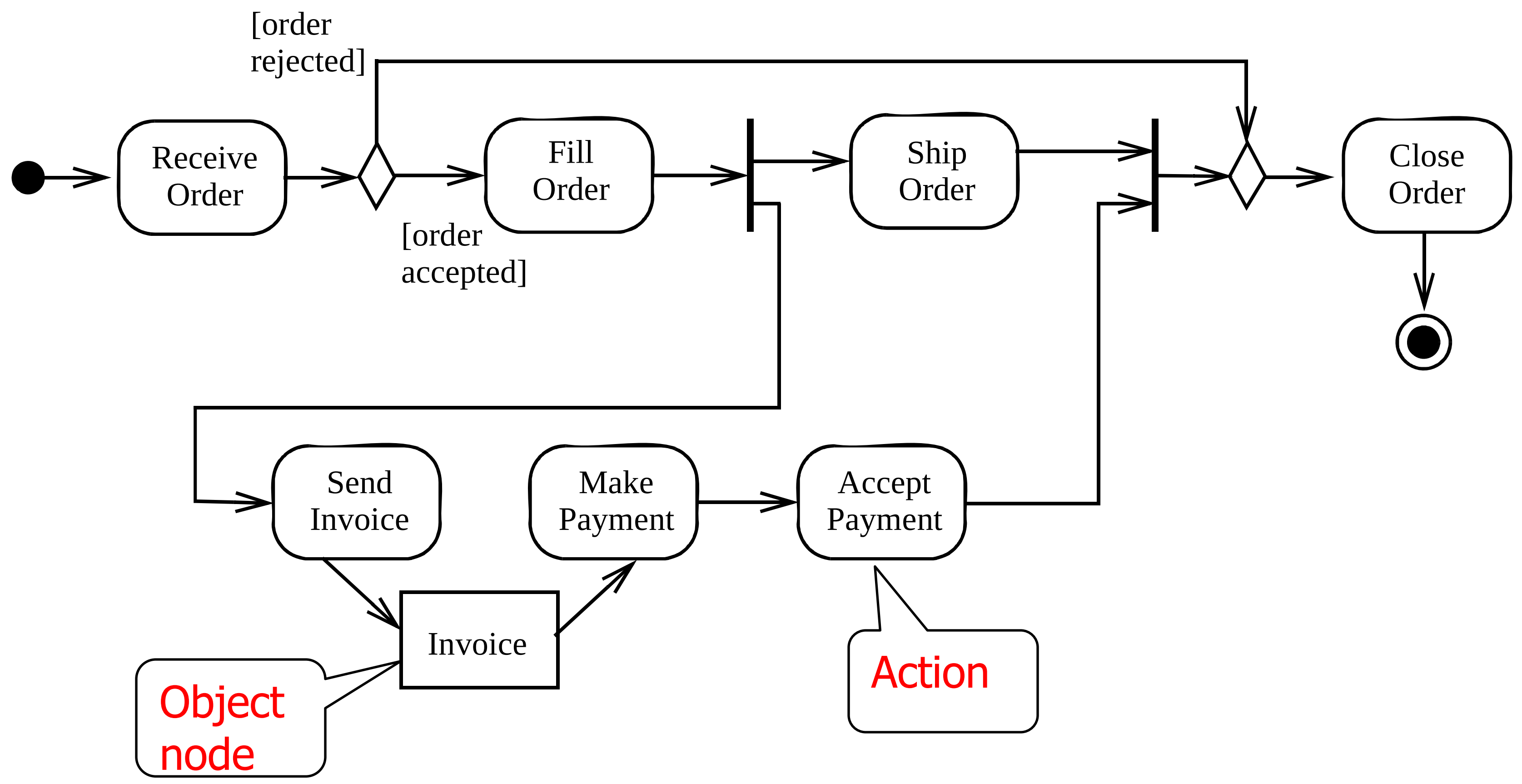
- Object Node



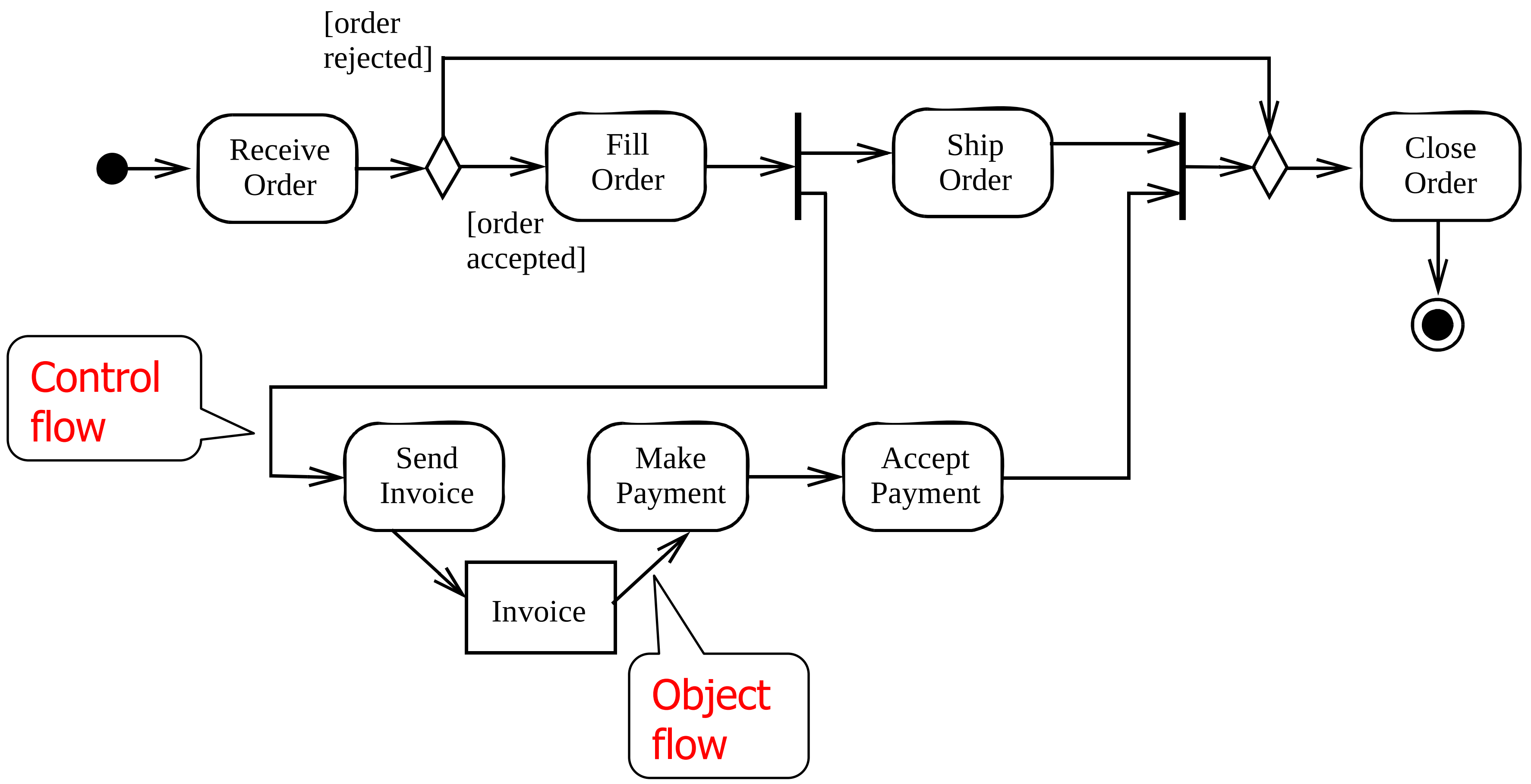
- An action is part of an activity which has local pre- and post conditions
- Historical Remark:
  - In UML 1 an action was the operation on the transition of a state machine.



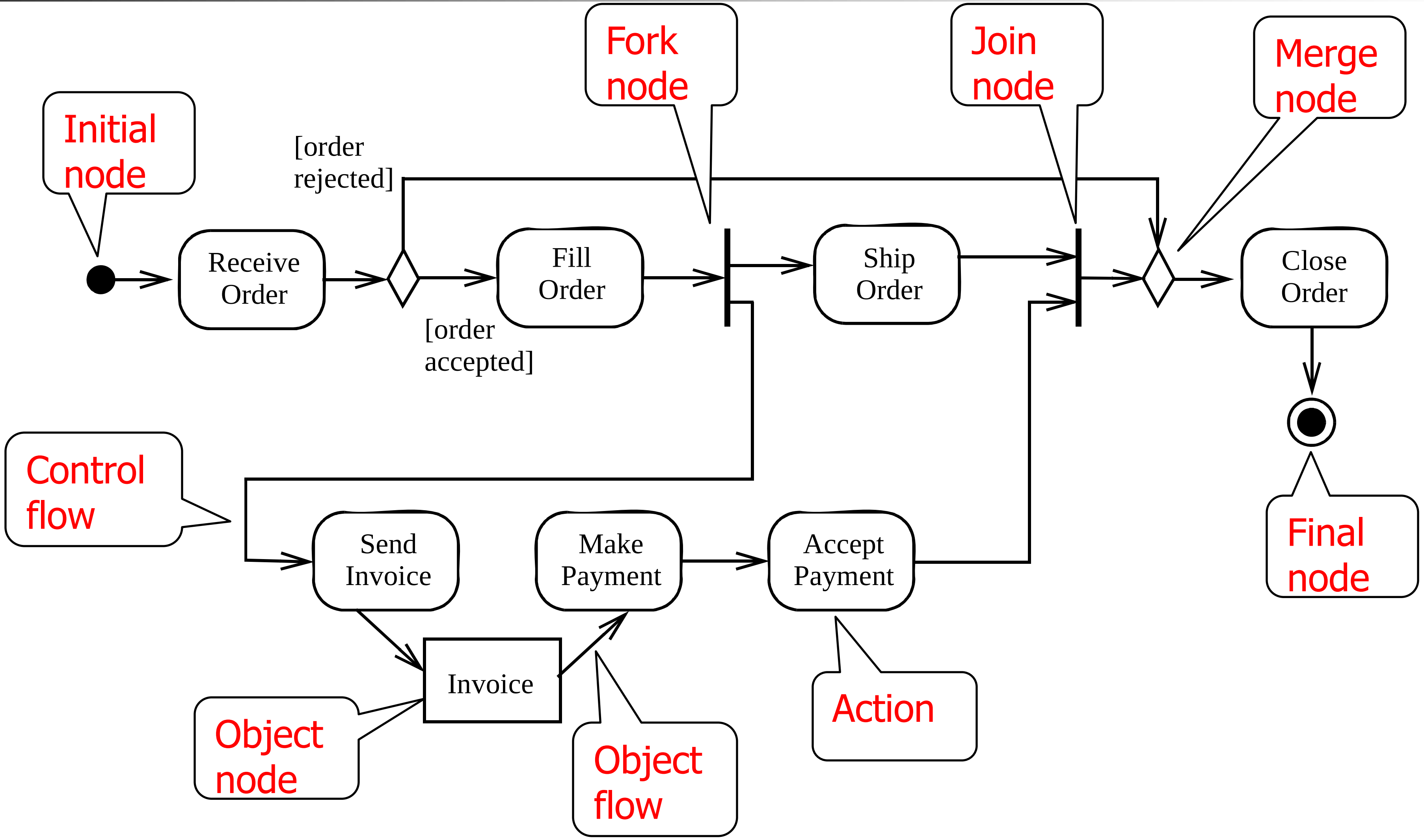
Activity Diagram Example



Activity Diagram Example



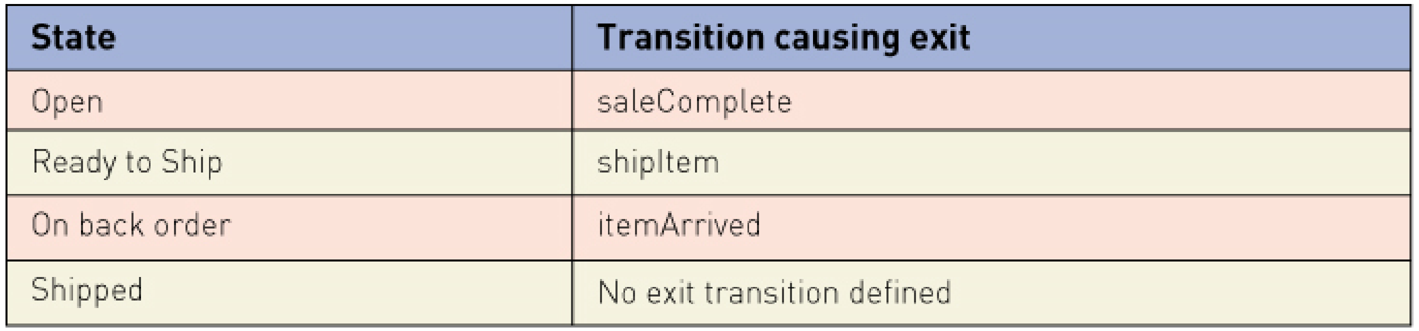
Activity Diagram Example



RMO – Creating a State Machine Diagram

Steps -- SaleItem

1. Choose SaleItem. It has status conditions that need to be tracked
2. List the states and exit transitions

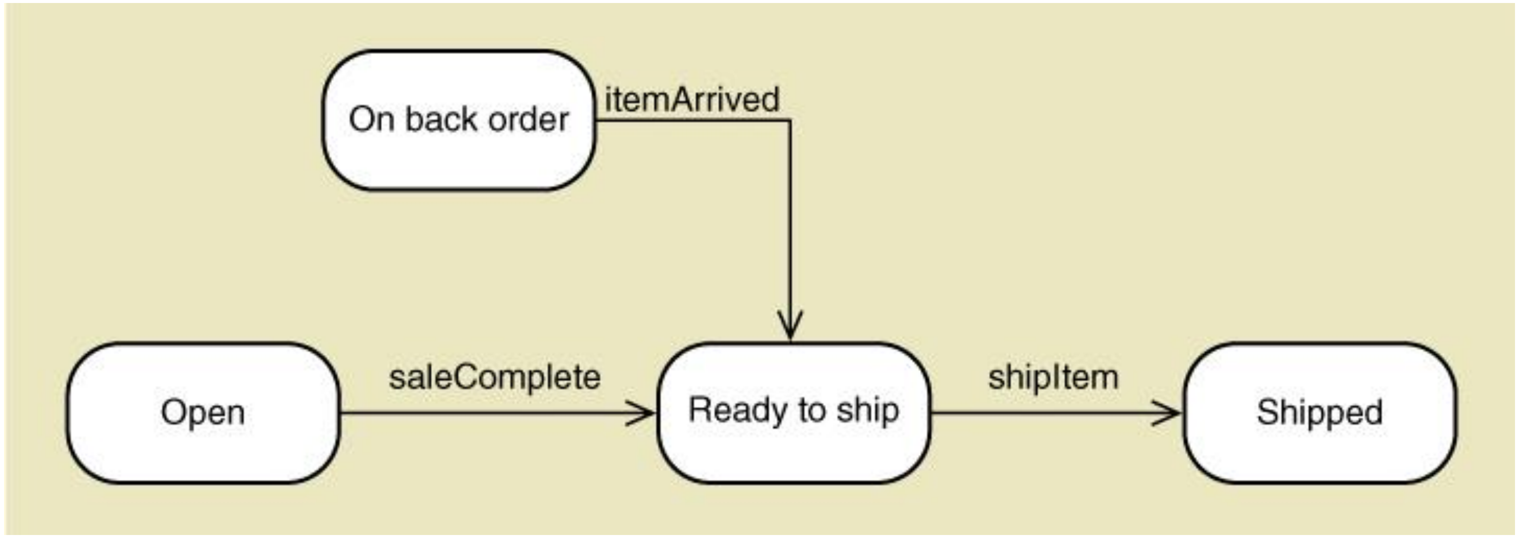


State	Transition causing exit
Open	saleComplete
Ready to Ship	shipItem
On back order	itemArrived
Shipped	No exit transition defined

RMO – Creating a State Machine Diagram

Steps -- SaleItem

3. Build fragments – see figure below
4. Sequence in correct order – see figure below
5. Look for concurrent paths – none

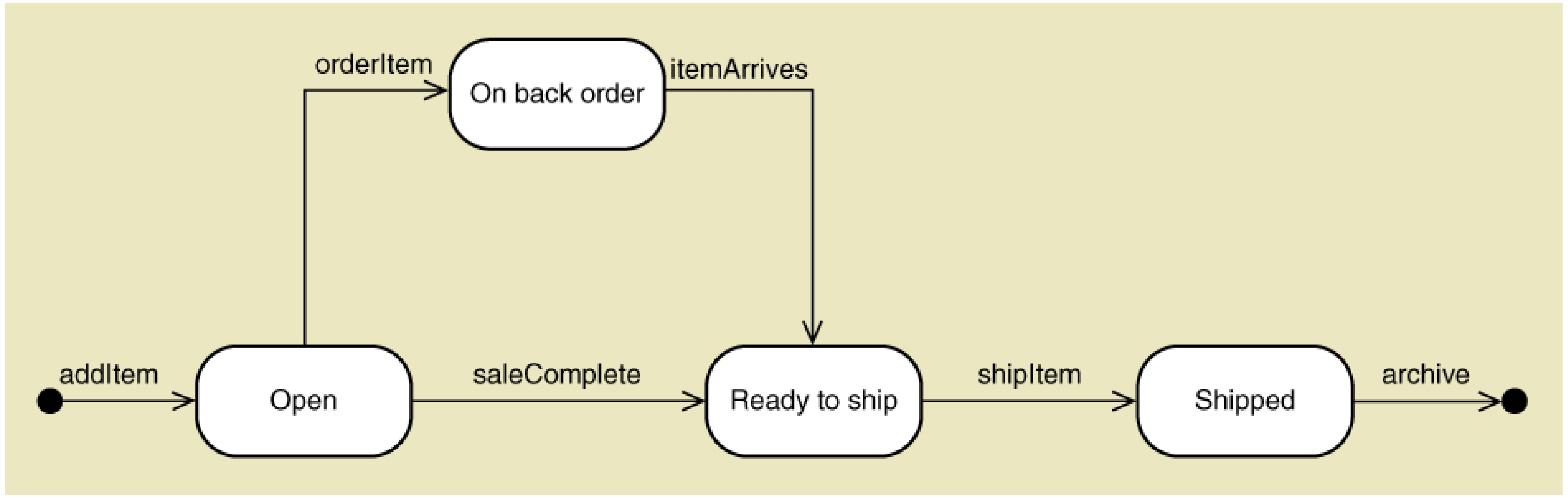


RMO – Creating a State Machine Diagram

Steps -- SaleItem

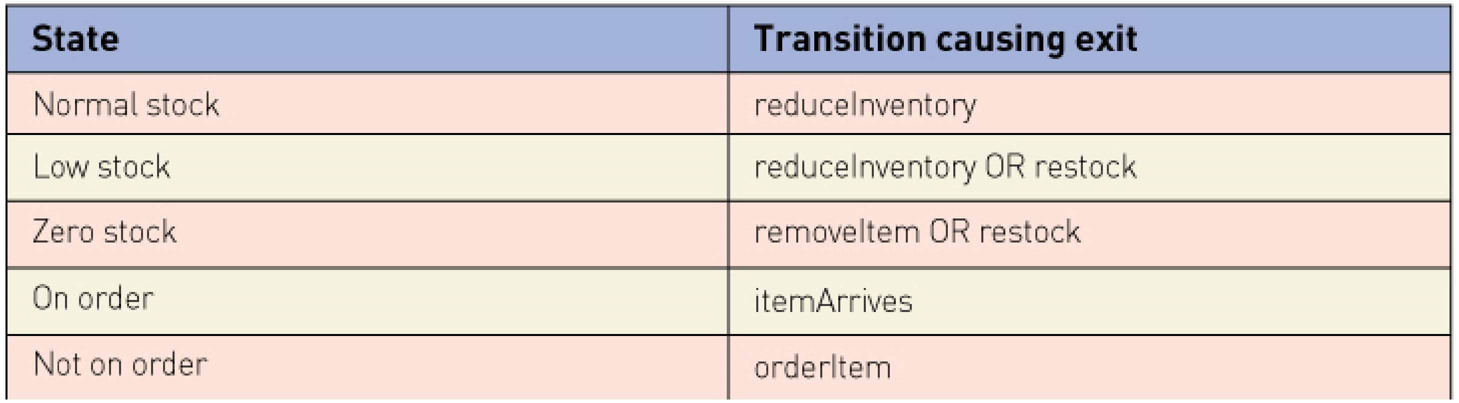
6. Add other required transitions
7. Expand with guard, action-expressions etc.
8. Review and test

Below is the final State Machine Diagram



RMO – Creating a State Machine Diagram
Steps -- InventoryItem

1. Choose InventoryItem. It has status conditions that need to be tracked
2. List the states and exit transitions



State	Transition causing exit
Normal stock	reduceInventory
Low stock	reduceInventory OR restock
Zero stock	removeItem OR restock
On order	itemArrives
Not on order	orderItem

Diagram from Systems Analysis and Design in a Changing World, 7th Edition – Chapter 4

RMO – Creating a State Machine Diagram

Steps -- InventoryItem

3. Build fragments – see figure below
4. Sequence in correct order – see figure below
5. Look for concurrent paths – see figure below

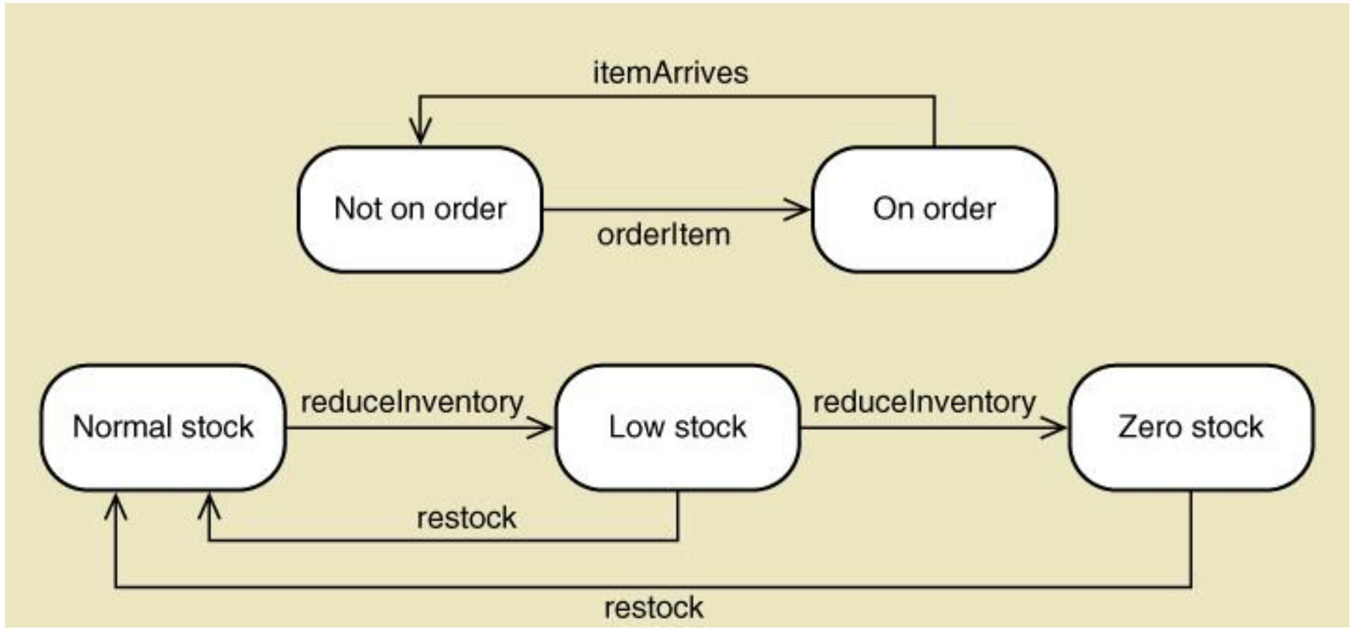


Diagram from Systems Analysis and Design in a Changing World, 7th Edition – Chapter 4

RMO – Creating a State Machine Diagram

Steps -- InventoryItem

6. Add other required transitions
7. Expand with guard, action-expressions etc.
8. Review and test

Below is the final State Machine Diagram

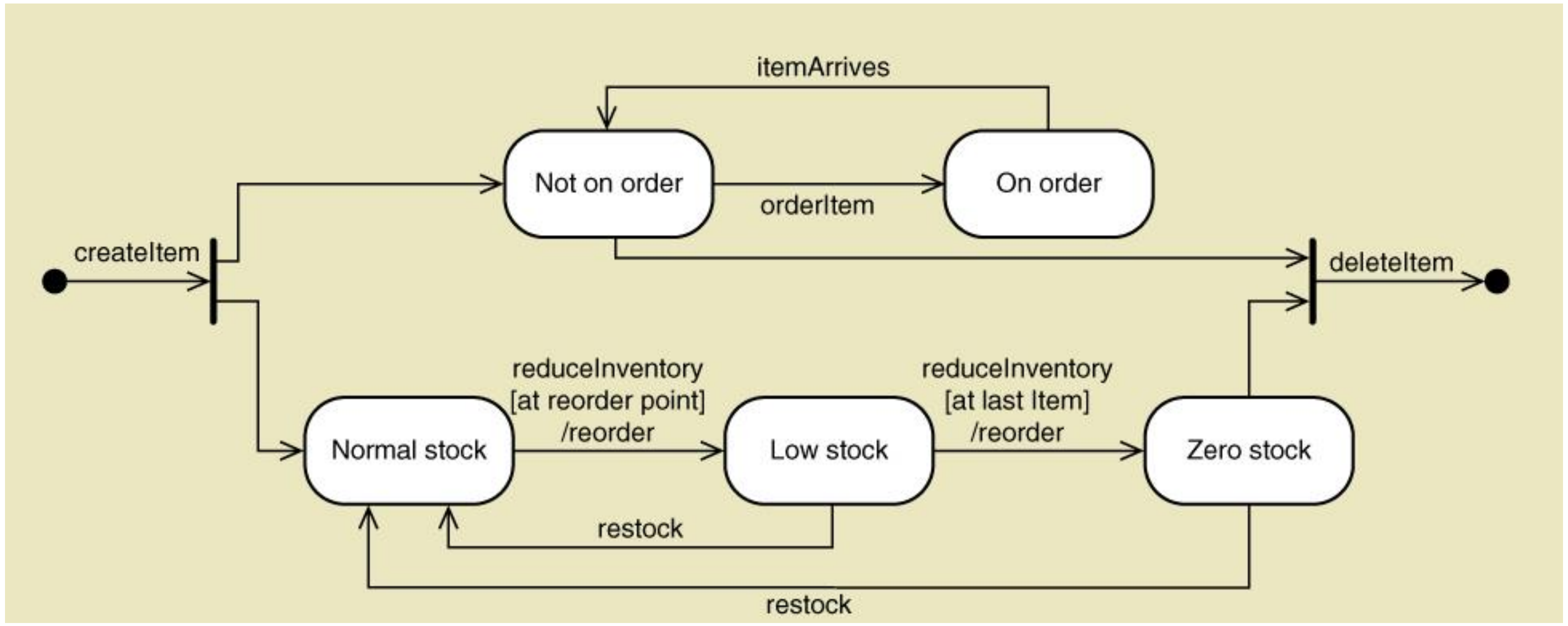
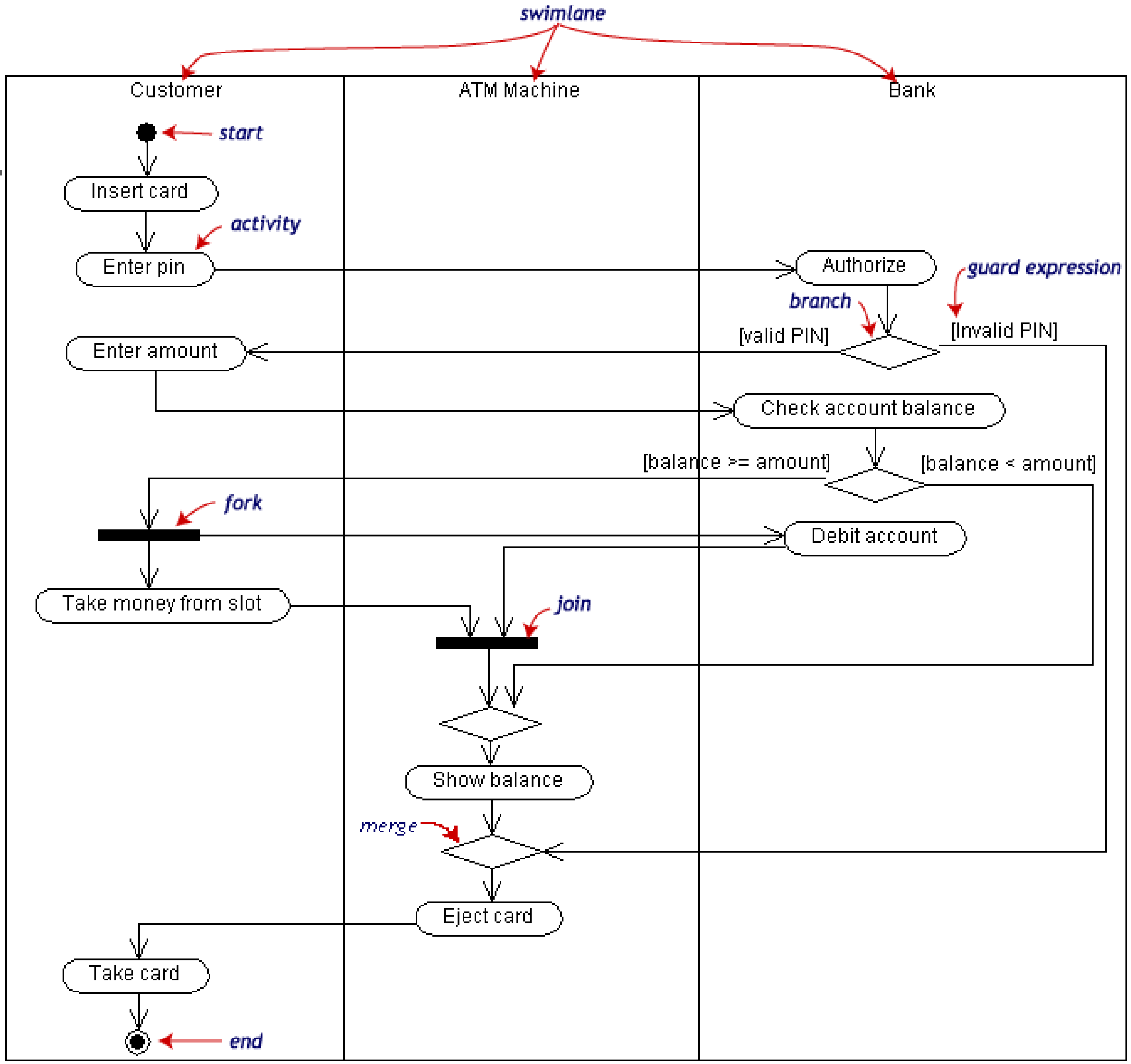


Diagram from Systems Analysis and Design in a Changing World, 7th Edition – Chapter 4



From: http://edn.embarcadero.com

Requirements Analysis Document Template

1. Introduction
2. Current system
3. Proposed system
    - 3.1 Overview
    - 3.2 Functional requirements
    - 3.3 Nonfunctional requirements
    - 3.4 Constraints ("Pseudo requirements")
    - ➡ 3.5 System models
        - 3.5.1 Scenarios
        - 3.5.2 Use case model
        - 3.5.3 Object model
            - 3.5.3.1 Data dictionary
            - 3.5.3.2 Class diagrams
        - 3.5.4 Dynamic models
        - 3.5.5 User interface
4. Glossary

Section 3.5 System Model

3.5.1 Scenarios

- As-is scenarios, visionary scenarios

3.5.2 Use case model

- Actors and use cases

➡ 3.5.3 Object model

- Data dictionary
- Class diagrams (classes, associations, attributes and operations), including entity, boundary, and control classes, likely placed in different packages

➡ 3.5.4 Dynamic model

- Sequence diagrams for collaborating objects (use cases)
- State diagrams for classes with significant dynamic behavior

3.5.5 User Interface

- Navigational Paths, Screen mockups

Section 3.5 System Model

3.5.1 Scenarios

- As-is scenarios, visionary scenarios

3.5.2 Use case model

- Actors and use cases

3.5.3 Object model

- Data dictionary
- Class diagrams (classes, associations, attributes and operations), including entity, boundary, and control classes, likely placed in different packages

Remember about boundary and control classes!

3.5.4 Dynamic model

- Sequence diagrams for collaborating objects (use cases)
- Statechart diagrams for classes with significant dynamic behavior

Include one statechart diagram for the term Project

3.5.5 User Interface

- Navigational Paths, Screen mock