UML Dynamic Diagrams

State Diagrams

Behavioral Diagrams

Used to visualize, specify, construct, document dynamic aspects of system

- use case diagram
- sequence diagram
- collaboration diagram
- ***State machine diagram***
- ***activity diagram***

Are UML Primary Diagrams

- State machine diagrams
  - Describe the dynamic behavior of an individual object as a finite state machine.
- Activity diagrams
  - Model the dynamic behavior of a system, in particular the workflow, i.e. a flowchart.

State Machine and Transition

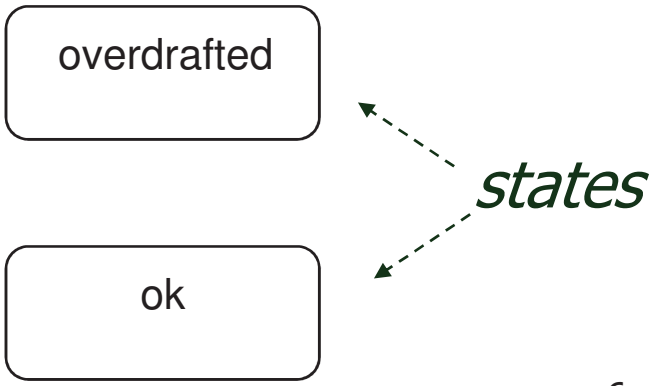
- A state machine is a behavior that specifies the sequences of states that an object goes through in its lifetime, in response to events, and also its responses to those events.
- A transition is a relationship between two states; it indicates that an object in the first state will perform certain actions, then enter the second state when a given event occurs.

State Machine Diagrams

- Shows the behavior of an object from a single class
  - shows all possible states for this object
  - shows how the object's state changes as a result of messages it receives
  - Useful for objects with attributes that have a small number of states or values
  - Also called state diagram, statechart diagram.

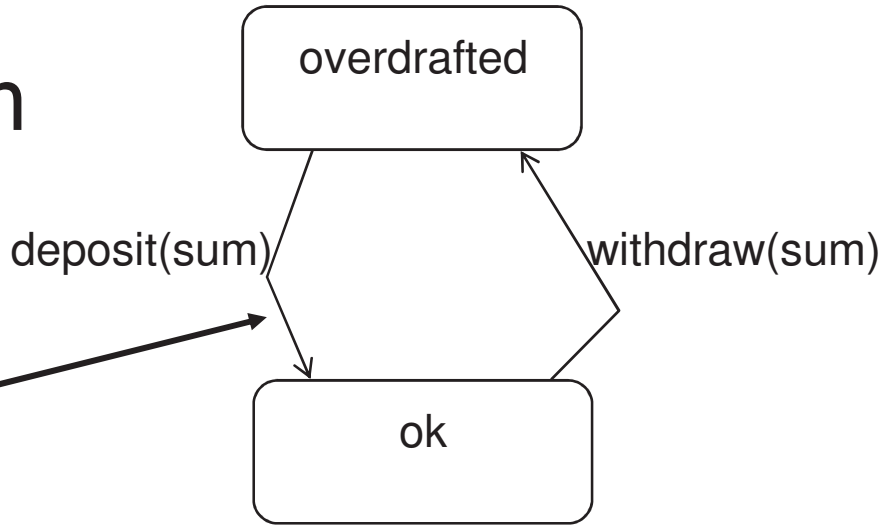
Object States

- State
  = set of values that describe an object (its condition/situation) at a specific moment in time
- State is determined based on the attribute values



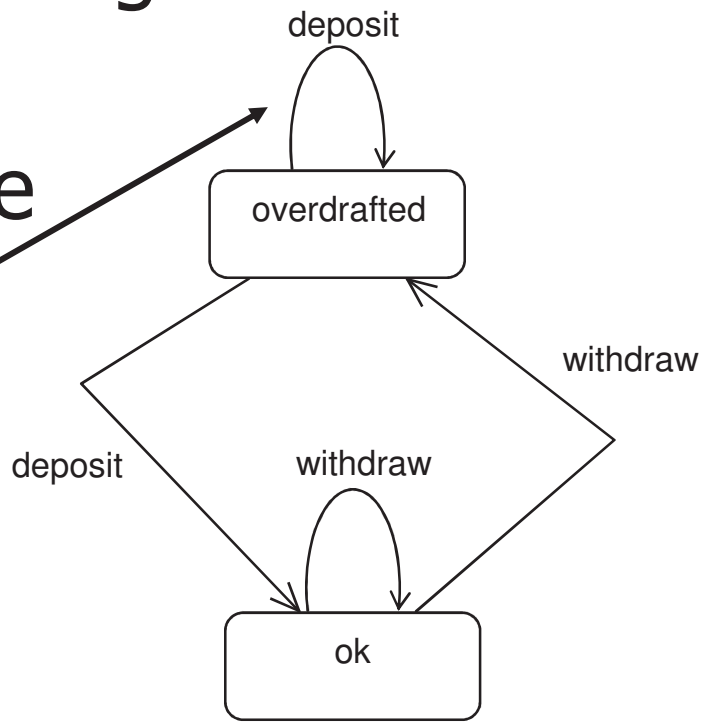
State Changes (1)

- States may be changed when an event occurs
- State transition: relationship indicating a state change
  - atomic (i.e. non-interruptible)



State Changes (2)

- Events: Messages or signals received
- Events may or may not change the state
  - Self-transition



States

- A **state** is a condition in which an object can reside during its lifetime while it satisfies some condition, performs an activity, or waits for an event.
- A state may be indicated by values of one or more variables,
- or it may be recognized by the legal operations currently permitted.
- "A state represents the response of the object to input events." (Rumbaugh)
- States depend on the level of abstraction in use:
  - My calendar says "traveling to London" (a single state)
  - The airline says "Atl-Mtl, Mtl-Lon.

Events

- An **event** is a significant occurrence that has a location in time and space.
- Events happen "instantaneously" (states have duration).
- "An event is a one-way transmission of information from one object to another." (Rumbaugh).
- If there is a reply, it is a separate event.

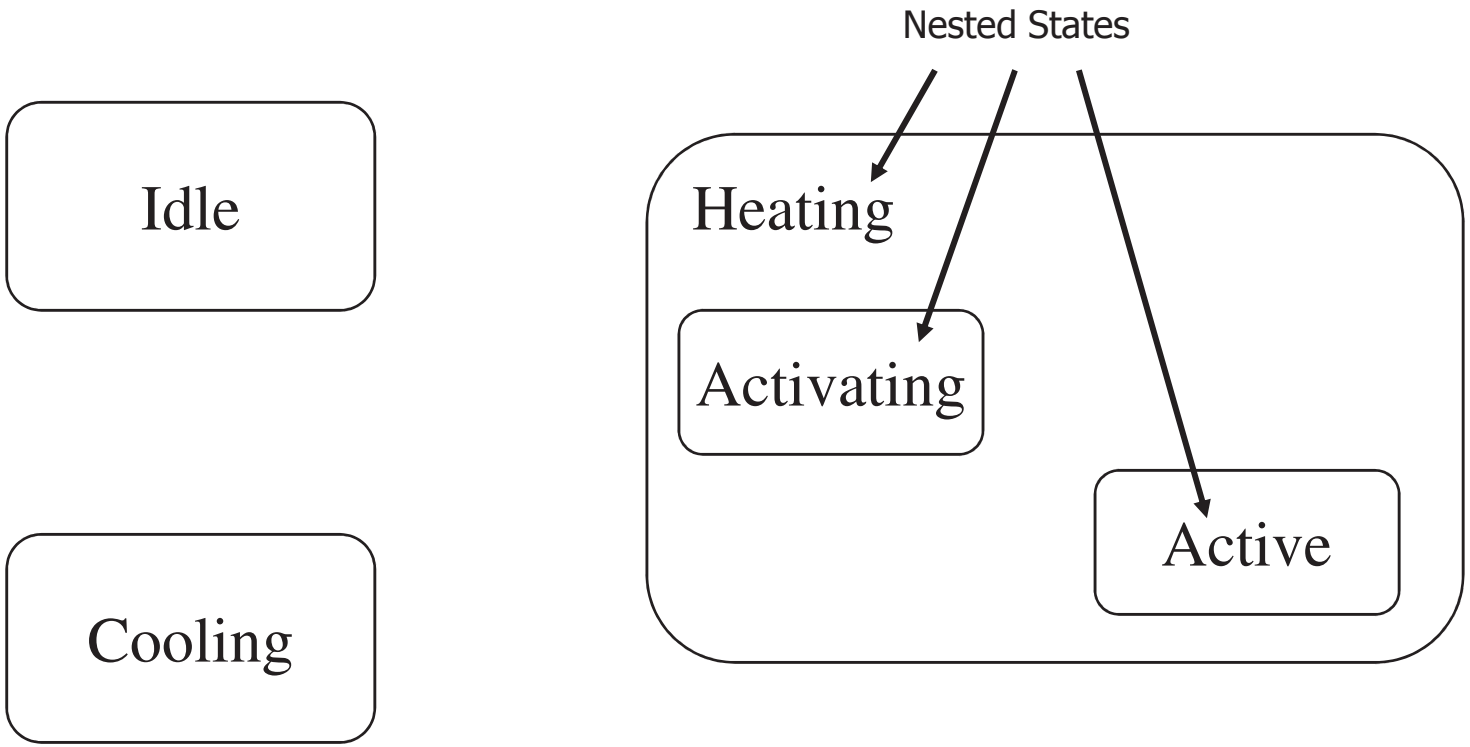
Other Stuff in State Diagrams

- Guards (conditions): a logical condition that must be true, *in addition* to the firing of an event.
- "When you arrive to class late (event), if the professor is in a good mood (condition), you'll be allowed in (new state).

Other Stuff (cont.)

- Operations
  - Happen within a state – entry actions or operations (Moore convention), or
  - Can happen at the time of transition and cause the transition – exit actions or operations (Mealy convention).
  - Operations use the same notation as class operations.
  - Entry and exit operations are also distinguished.

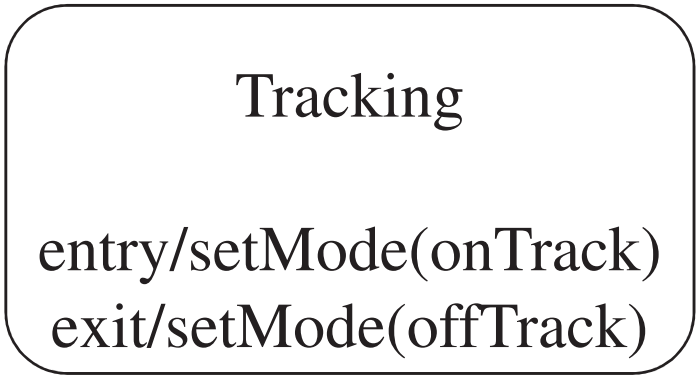
State Notation



Entry and Exit Actions

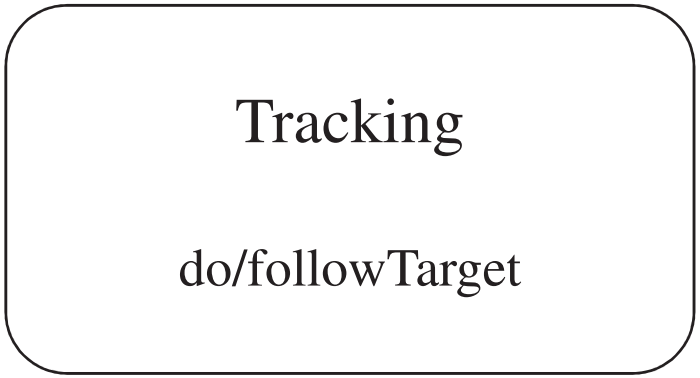
An entry action (or operation) is the first thing that occurs each time an object enters a particular state.

An exit action (or operation) is the last thing that occurs each time an object leaves a particular state.

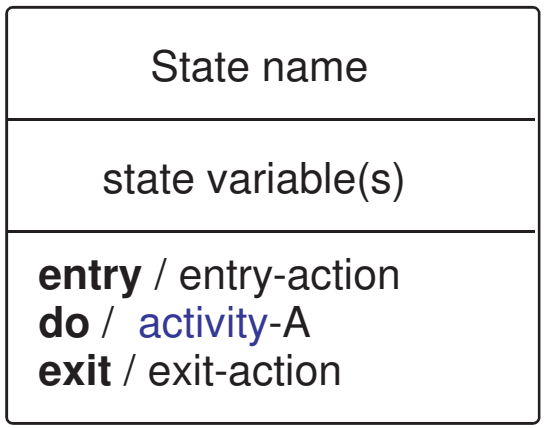


Activities

An activity is an interruptible sequence of actions that an object can perform while it resides in a given state. (Actions are not interruptible.)



State Diagram Notation: for One State



- Activity: Can take longer and can be interrupted
- Action: Occurs quickly
  - "quickly" = non-interruptible
  - entry: an action that is performed on entry to the state
  - exit: an action performed on exiting the state
- do: an ongoing activity performed while in the state

State Diagram Notation: From One State to Another

- Event: significant occurrence that has a location in time and space
  - triggers the transition
  - signals, calls, passing of time, change in state
- Guard condition:
  - Transition only occurs when guard evaluates to true
  - Guards of transition exiting one state are mutually exclusive
- Action: Processes considered to occur quickly and are not interruptible

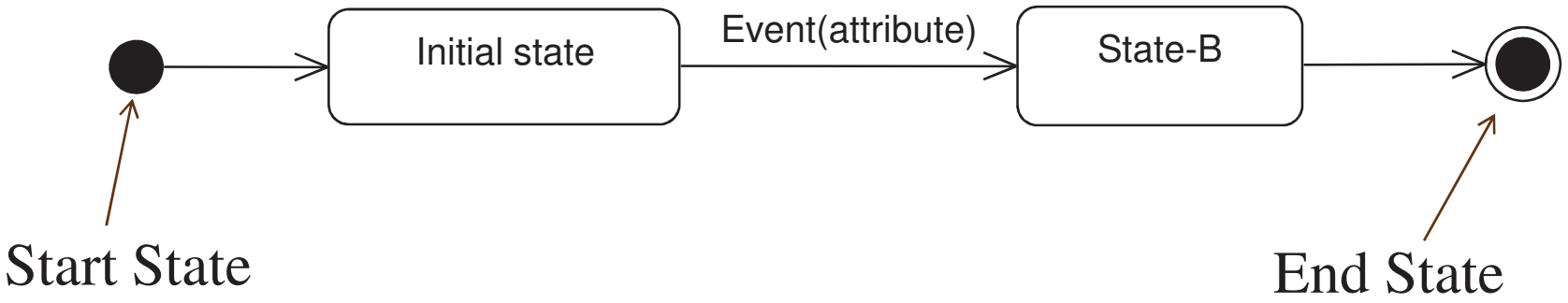
Note: Each part can be omitted!

Initial and Final States

The initial (or start) state is the default starting place for a state machine.

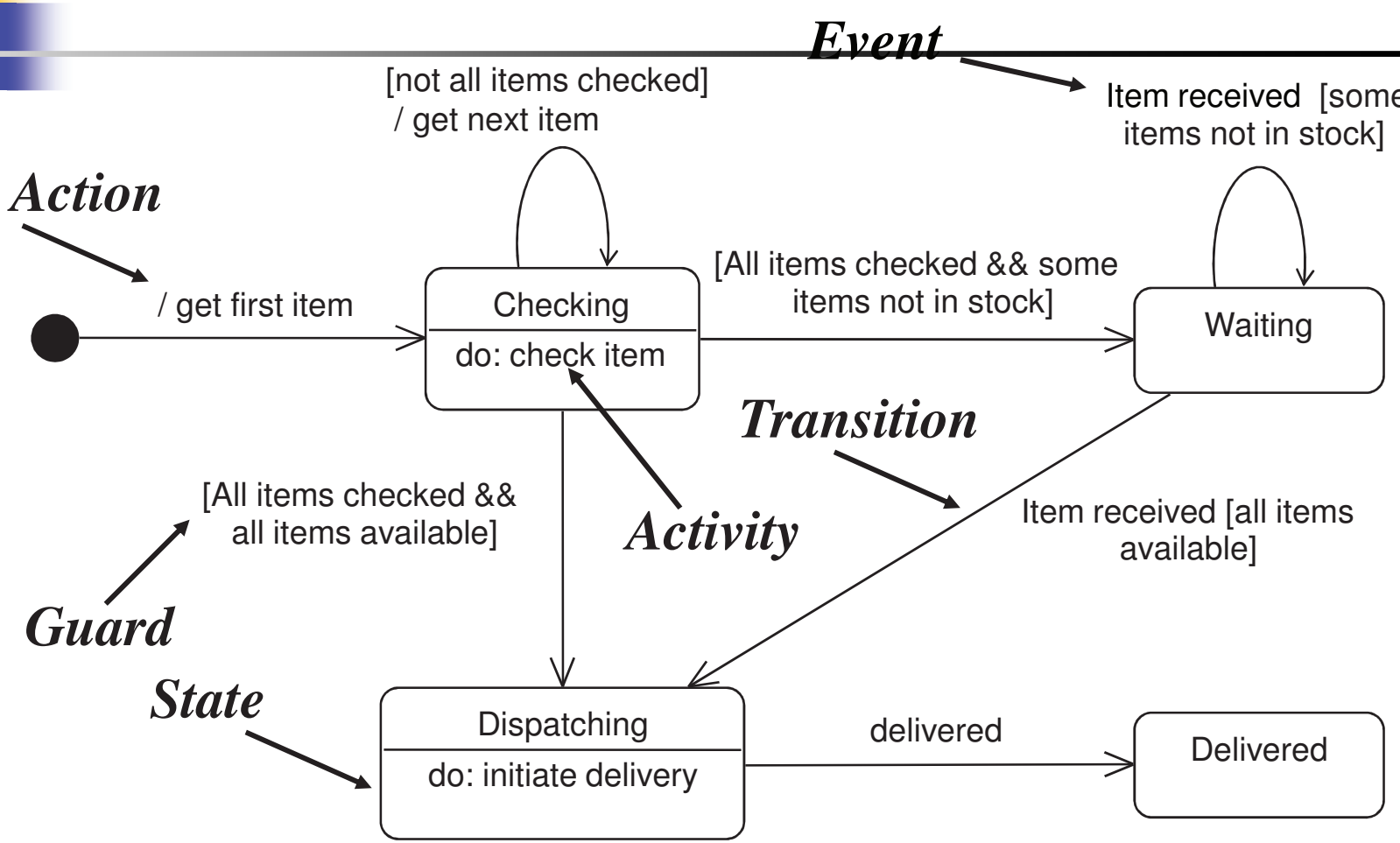
The (optional) final (or end) state indicates the completion of the state machine's execution.

State Diagram Notation: Special States



- Start state
  - No event triggers allowed
  - may not remain in start state
- End state
  - Top level end state terminates a state machine
  - Is optional – may be omitted on any particular diagram

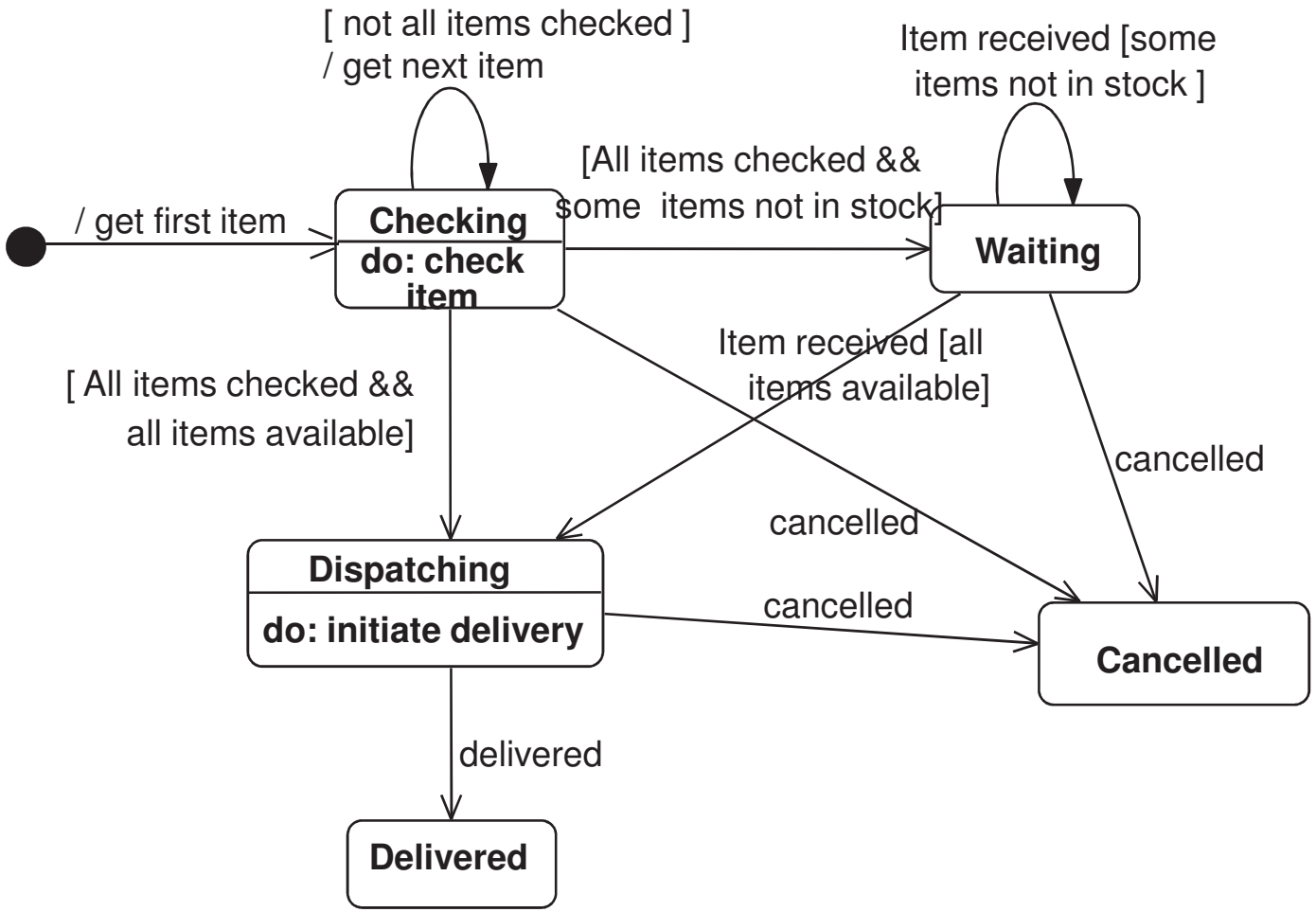
Example:
State Transitions For an Order



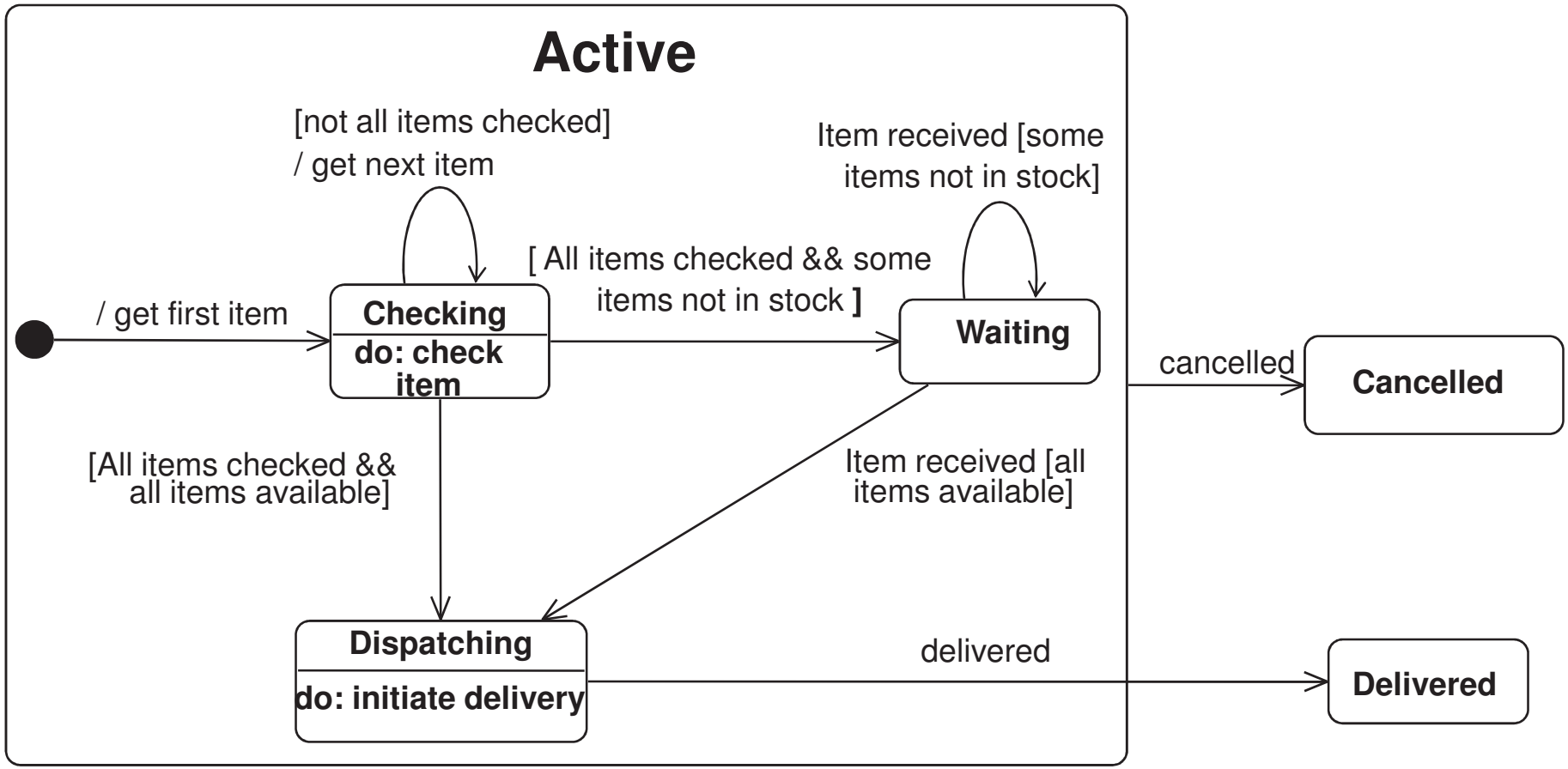
Example Problem: Cancel the Order

- Want to be able to cancel an order at any time
- Solutions
  - Transitions from every state to state "cancelled"
  - Superstate and single transition

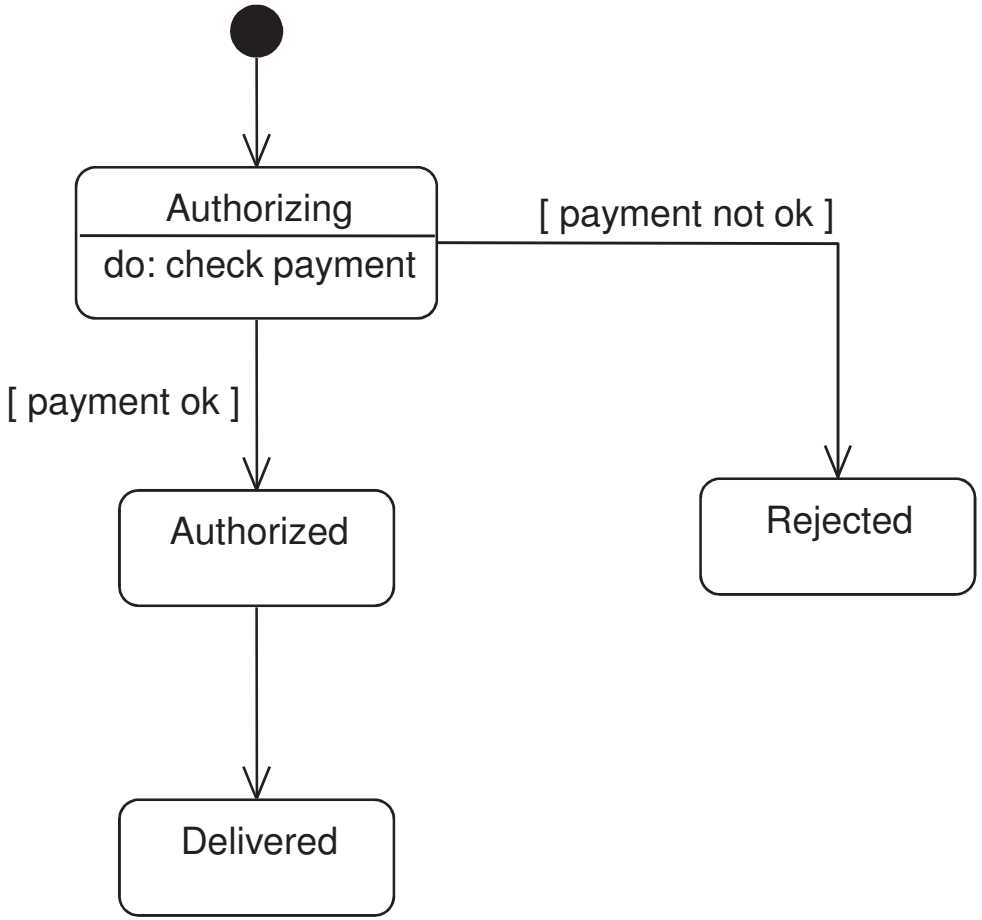
Example: Transitions to "cancelled"



Example: Superstate / Substates



Example: Payment Authorization



2 parallel processes:
- authorization
- order handling

Example: Concurrent State Diagram