Behavior With UML

State-charts and components

UML Diagrams

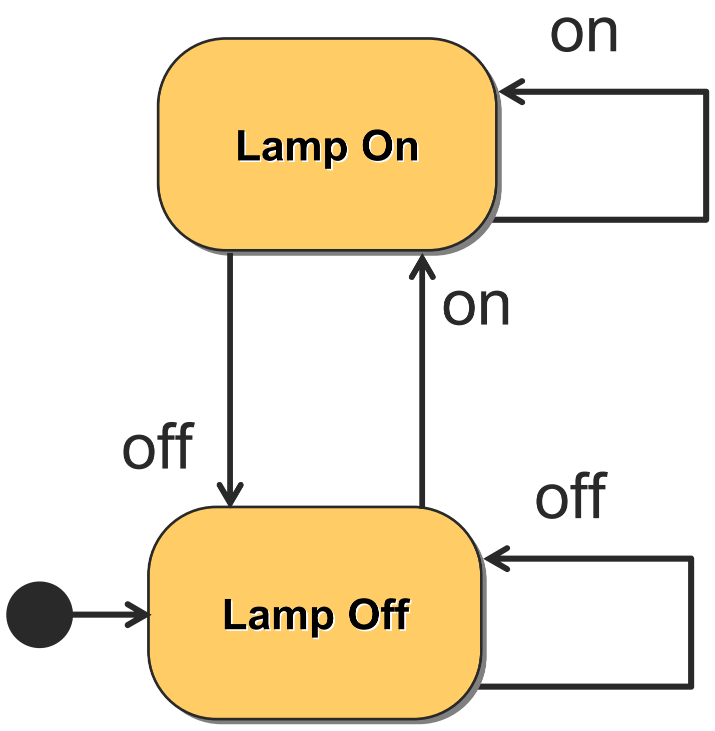
- Requirements capture
  - **Use-case diagrams**
- Structural view (static aspect of model)
  - **Class diagram**
  - Object diagram
- Functional view (interaction among objects
  - **Sequence diagram**
  - Communication diagram
- Behavioral view (object dynamics)
  - Activity diagram
  - **State-chart diagram**
- Deployment view
  - **Composite structure diagram**
  - Deployment diagram

Automaton

- A machine whose behavior is not only the consequence of the current input, but also the history of past inputs
- Characterized by an internal state which represents this past history of inputs

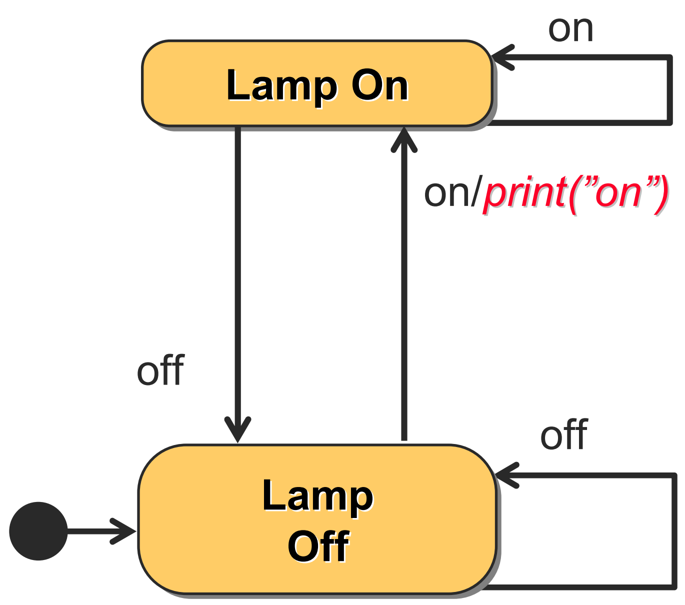
State machine

- A graphical representation of an automaton

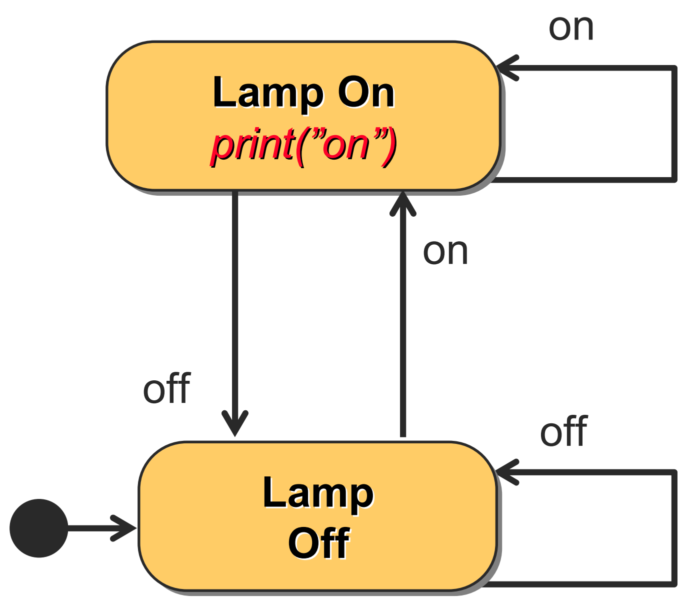


Outputs and Actions

- Outputs can be related to transitions or states



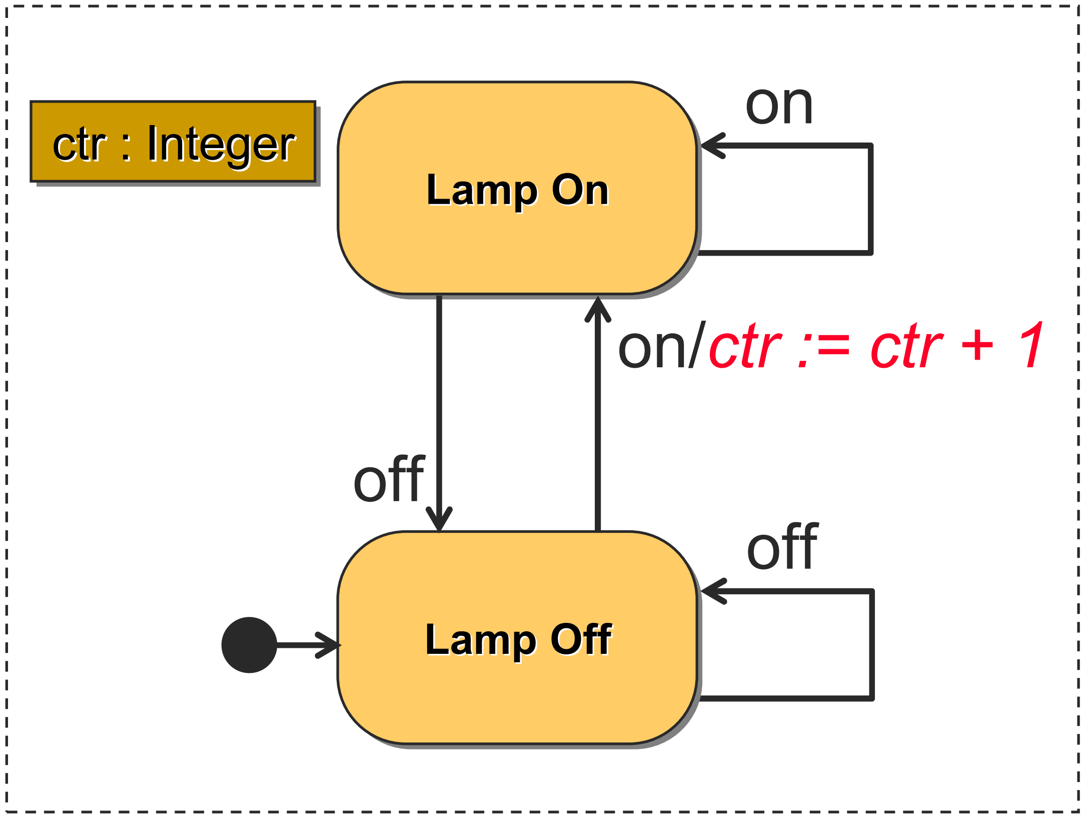
Mealy automaton



Moore automaton

Extended state machines

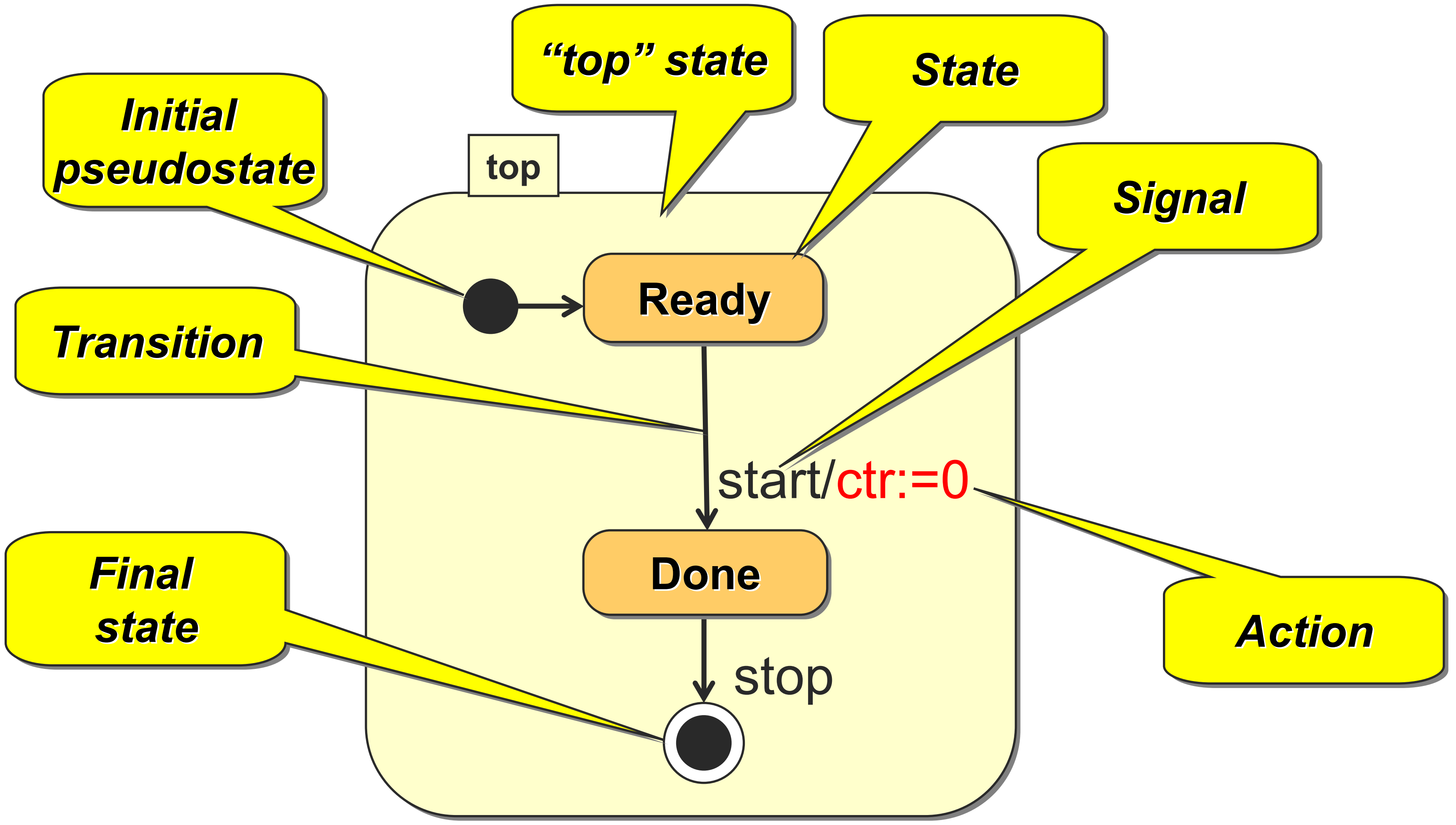
- Addition of variables (“extended state”)



A bit of theory

- An extended (Mealy) state machine is defined by:
  - A set of input signals (input alphabet)
  - A set of output signals (output alphabet)
  - A set of states
  - A set of transitions
    - Triggering signal
    - Action
  - A set of extended state variables
  - An initial state designation
  - A set of final states (if terminating automaton)

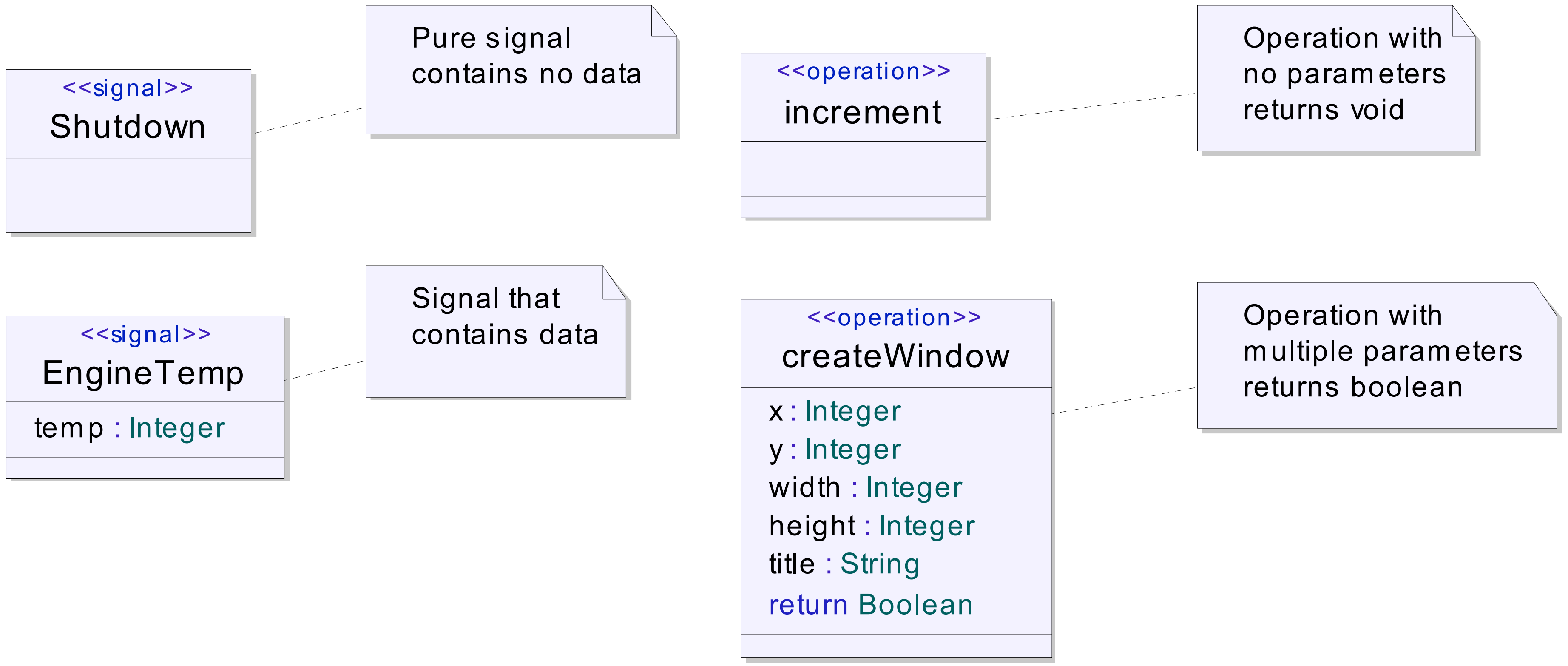
Basic State-chart Diagram



Event-driven Behavior

- Also called *reactive* behavior
- An event is a type of observable behavior
  - Interactions
    - Synchronous (operation call)
    - Asynchronous (signal transmission/reception)
  - Time events
    - Interval expiry
    - Calendar/clock times
  - Change events
    - Change in value of some entity (change events)
- Event Instance: an instance of an event of a certain type
  - Occurs at a particular instance of time, has no duration

Signals and Operations

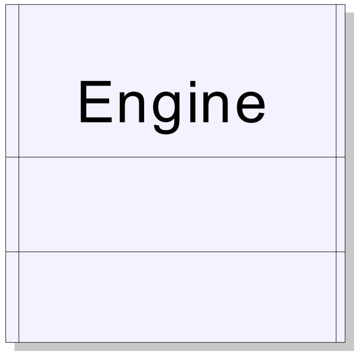


The Behavior of What?

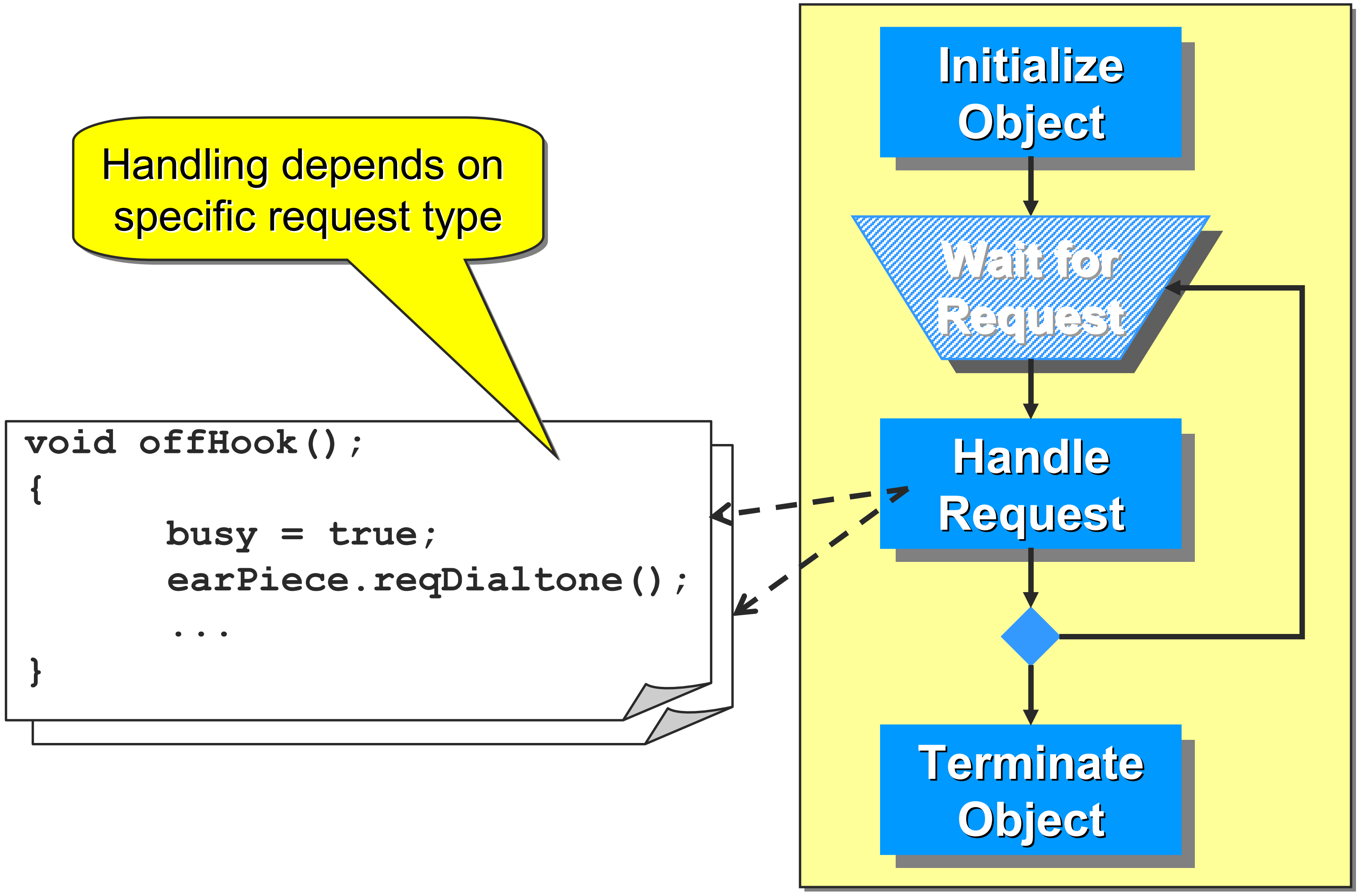
- In principle, anything that manifests event-driven behavior
  - There is no support currently in UML for modeling continuous behavior
- In practice:
  - The behavior of individual objects
  - Object interactions
- The dynamic semantics of UML state machines are currently mainly specified for the case of active objects

Active Classes

- An active class in UML is one that
  - Starts execution of its behavior as soon as an object of it is created
  - Does not cease until either
    - The behavior defined for it is completed
    - It is terminated by another object
  - So it is also referred to as having its own *thread of control*
- The points at which an object of an active class responds to communication is determined solely by its behavior and not by the invoking object
- Presentation
  - An active class is shown by a class box with additional vertical bars on the sides

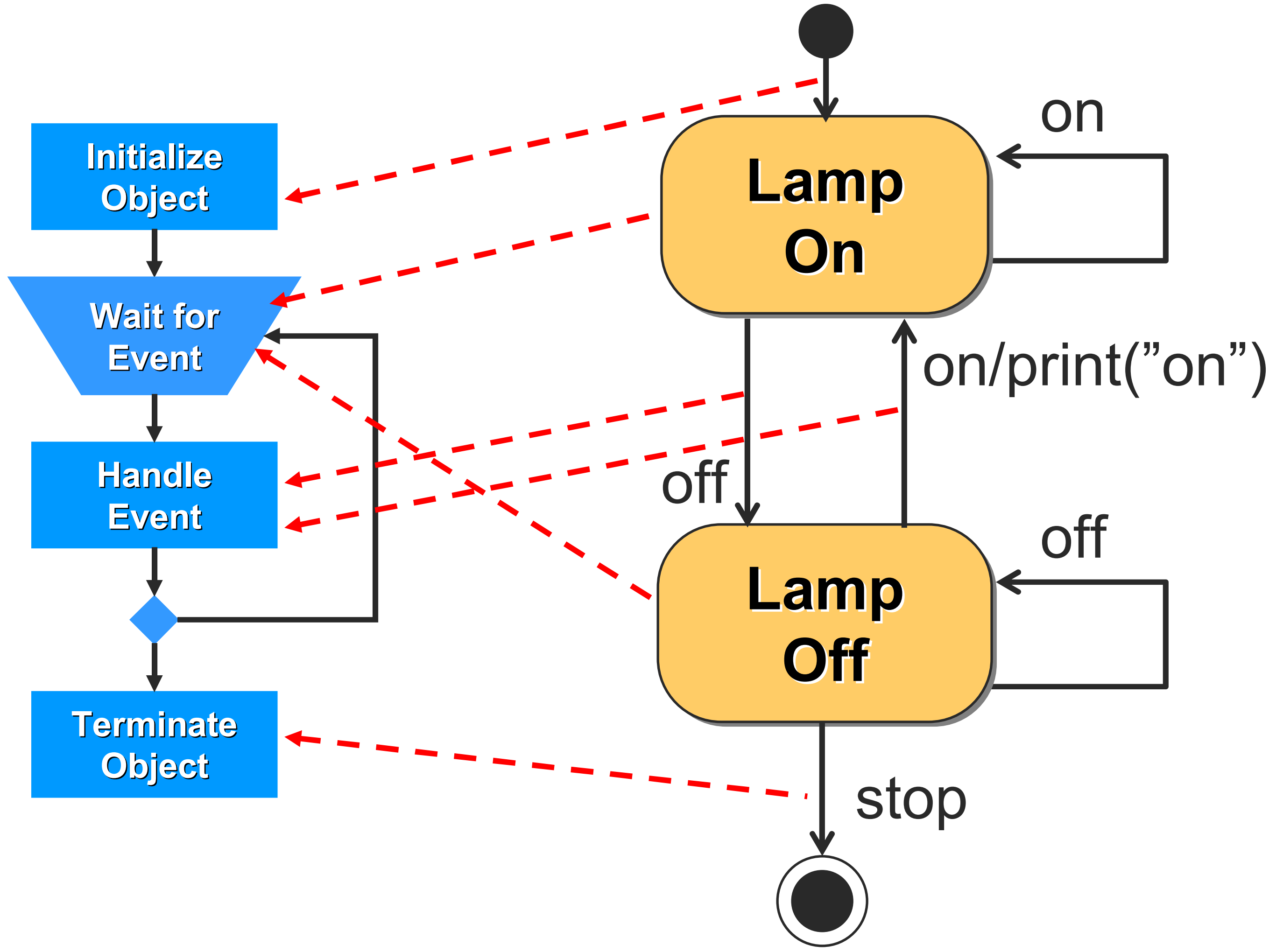


Object Behavior Model

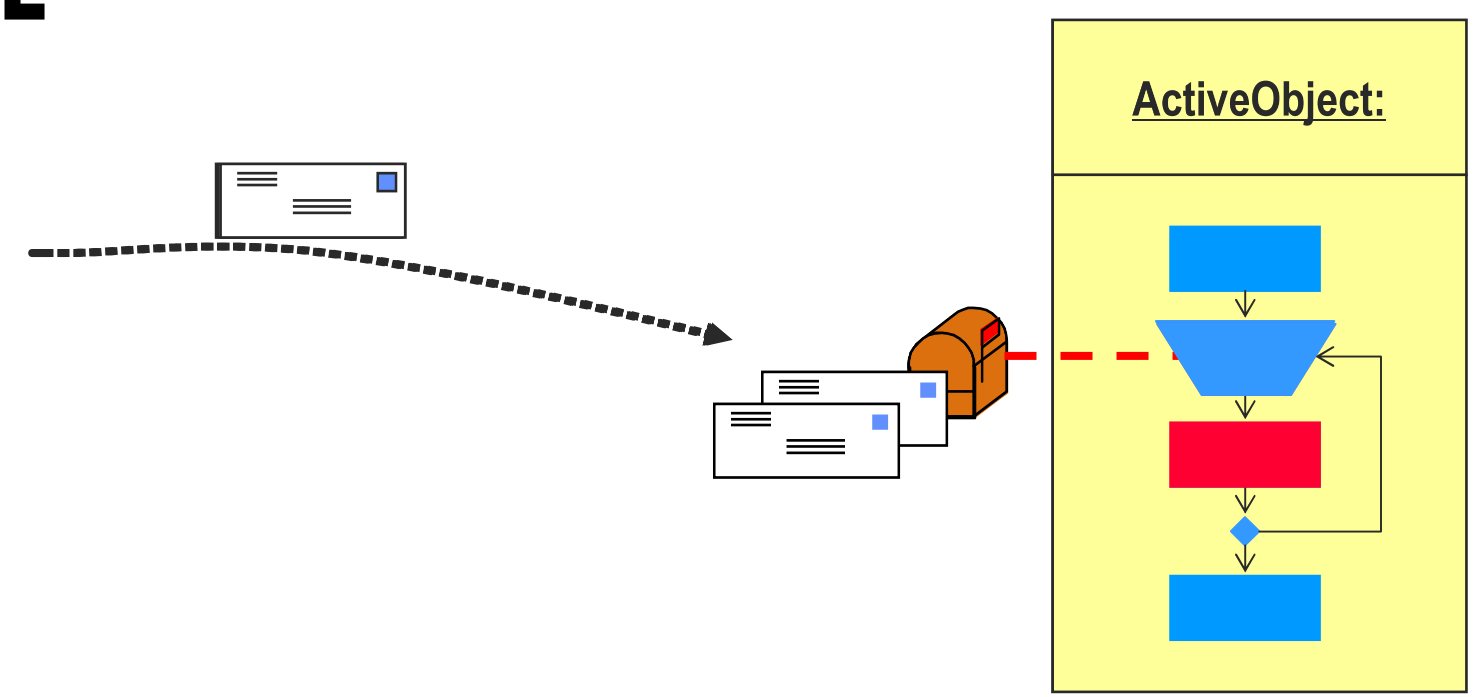


```
void offHook();
{
        busy = true;
        earPiece.reqDialtone();
        ...
}
```

Object Behavior and State Machines

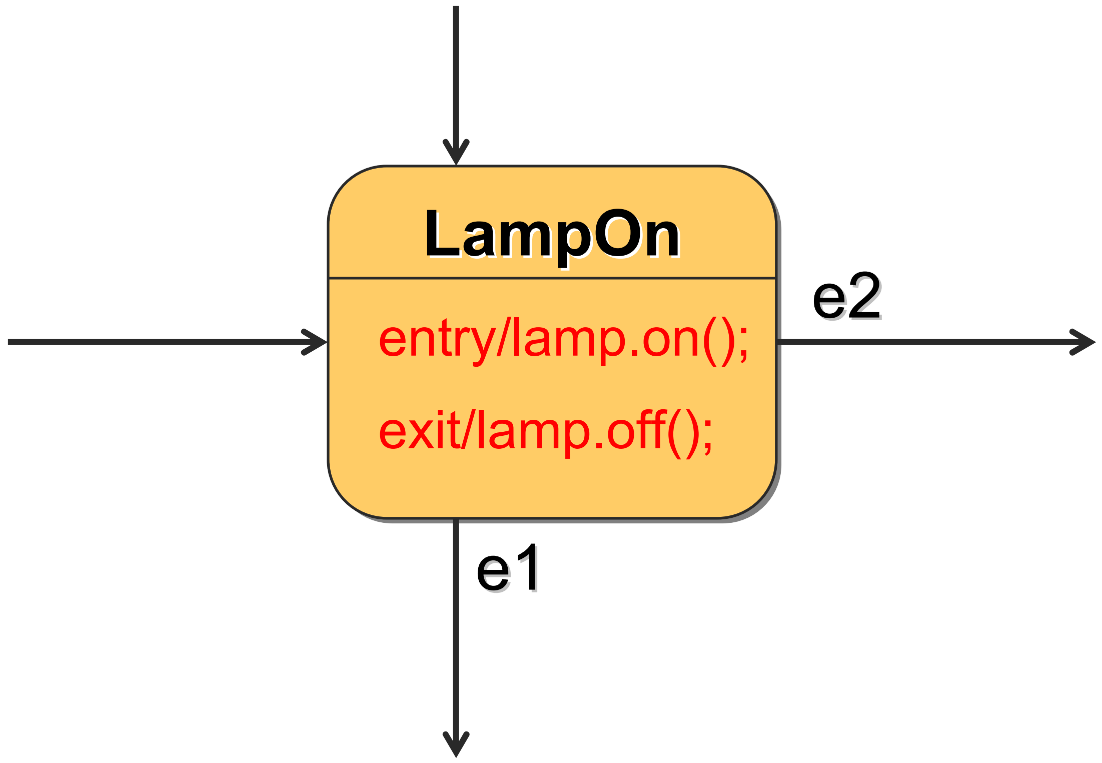


Dynamic Semantics of Active Objects



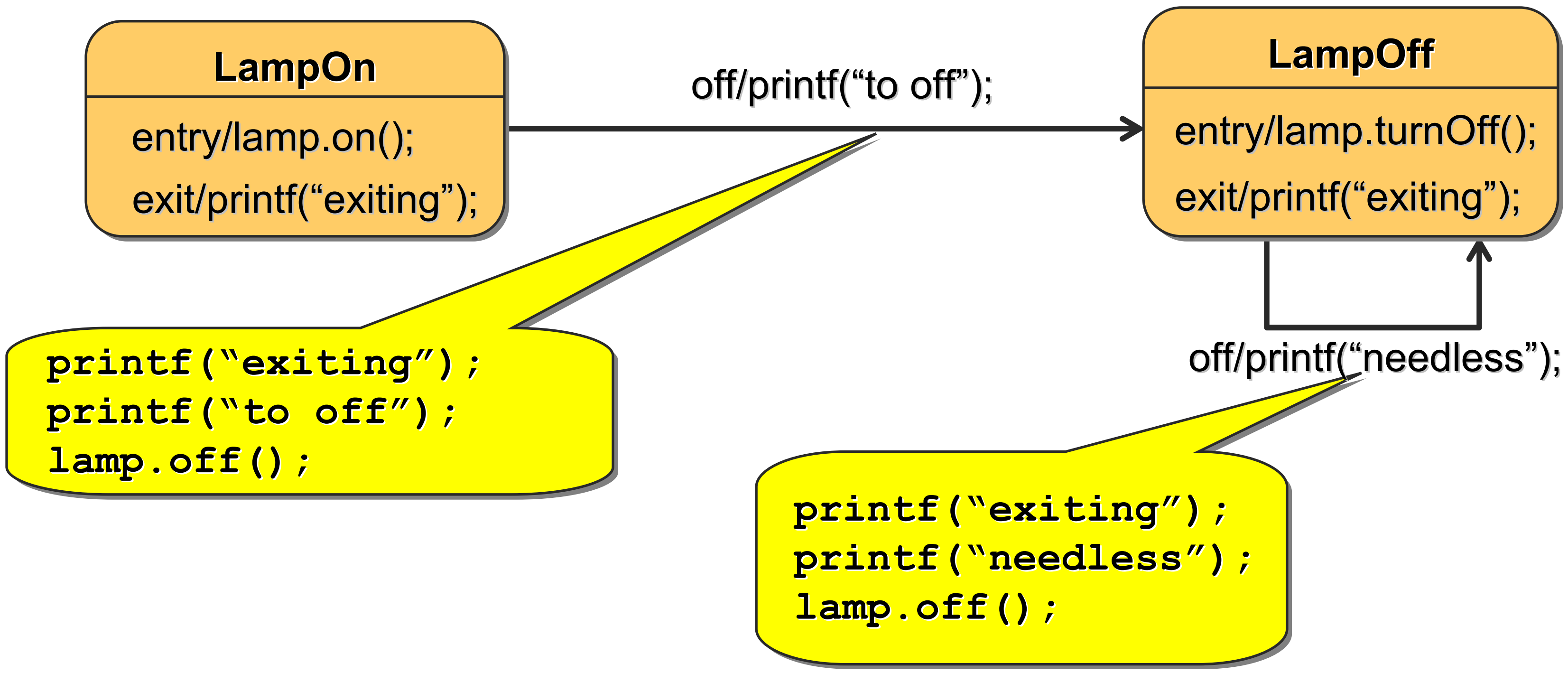
- **Run-to-completion model**
  - Serialized event handling
  - Eliminates internal concurrency
  - Minimal context switching overhead

State entry and exit actions



Order of actions

- Exit actions prefix transition actions
- Entry actions postfix transition actions

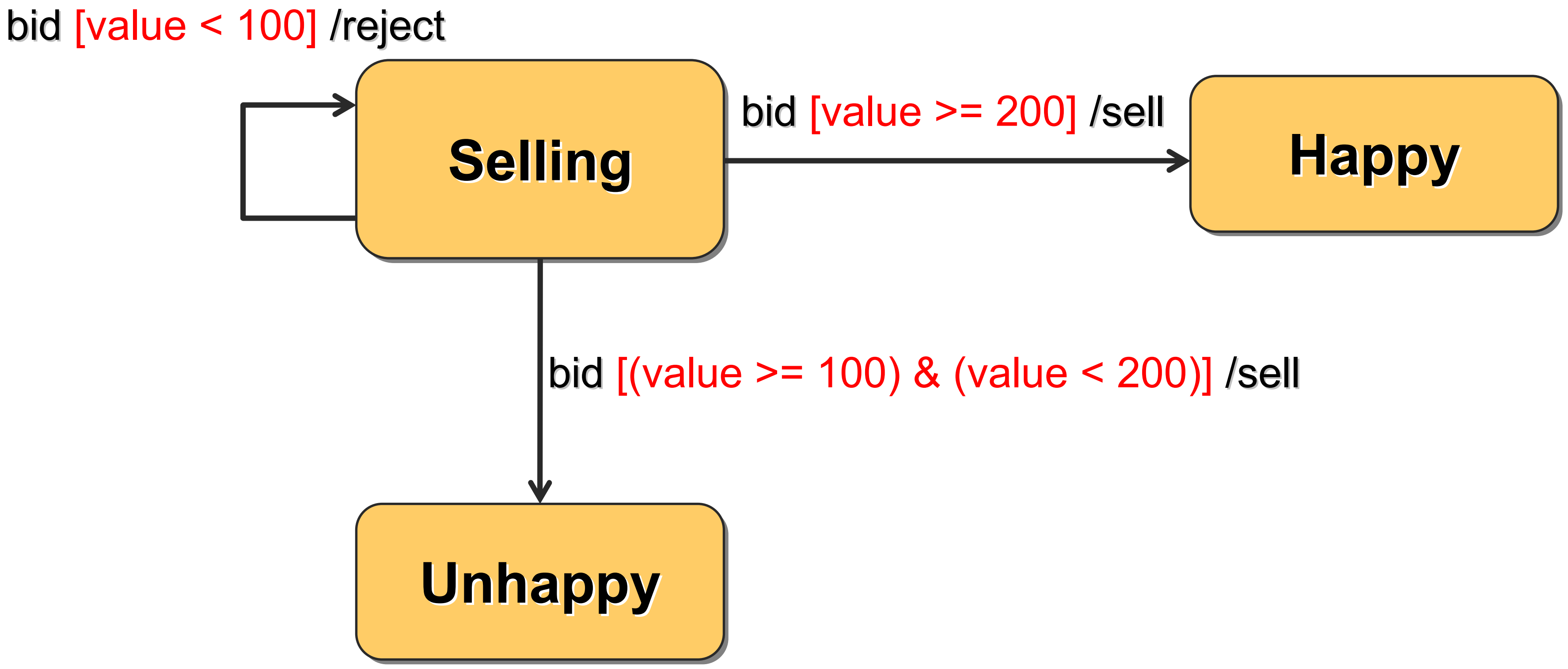


```
printf("exiting");
printf("to off");
lamp.off();
```

```
printf("exiting");
printf("needless");
lamp.off();
```

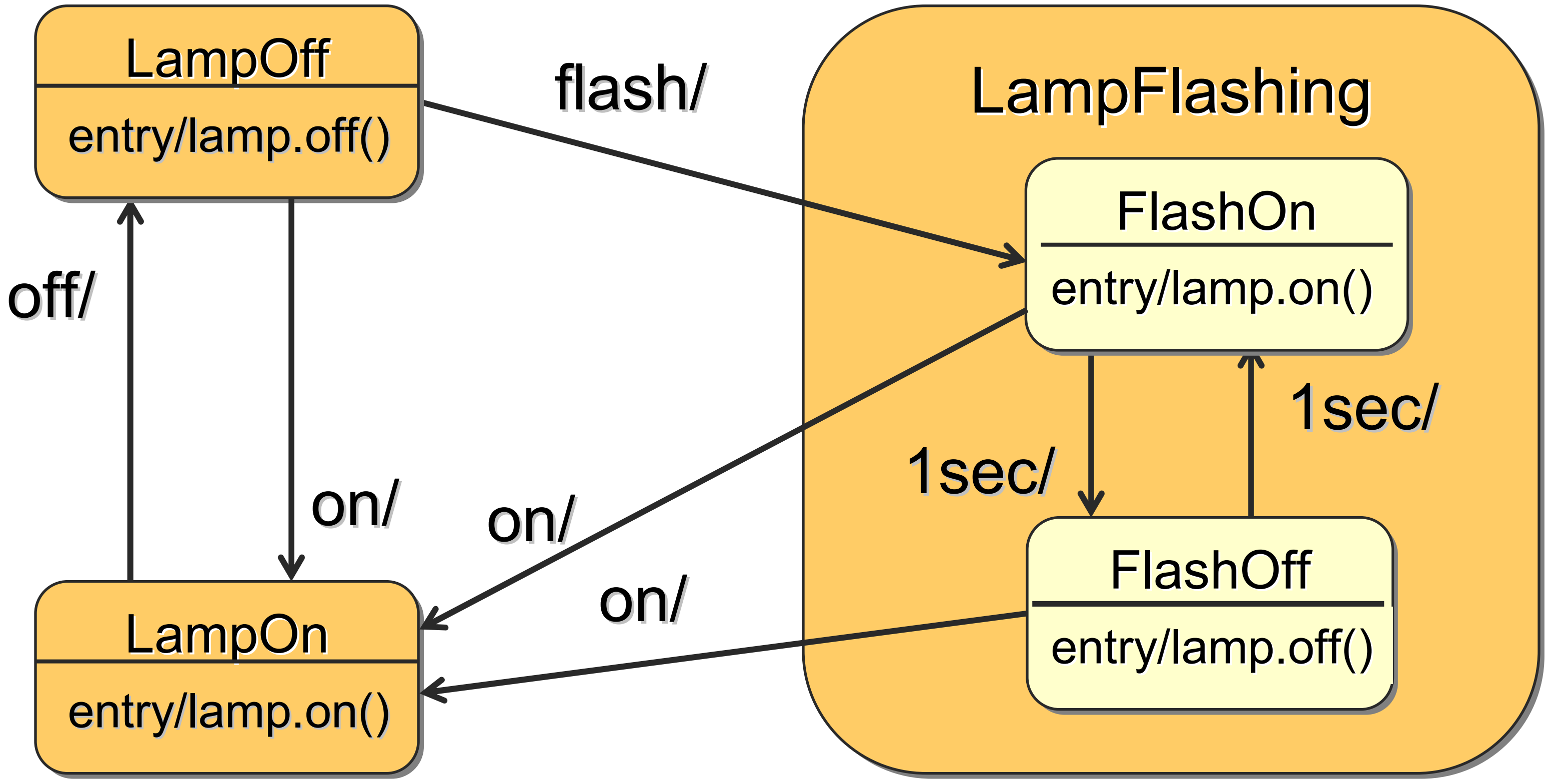
Guards

- Boolean predicates on transitions
- Must be side-effects free

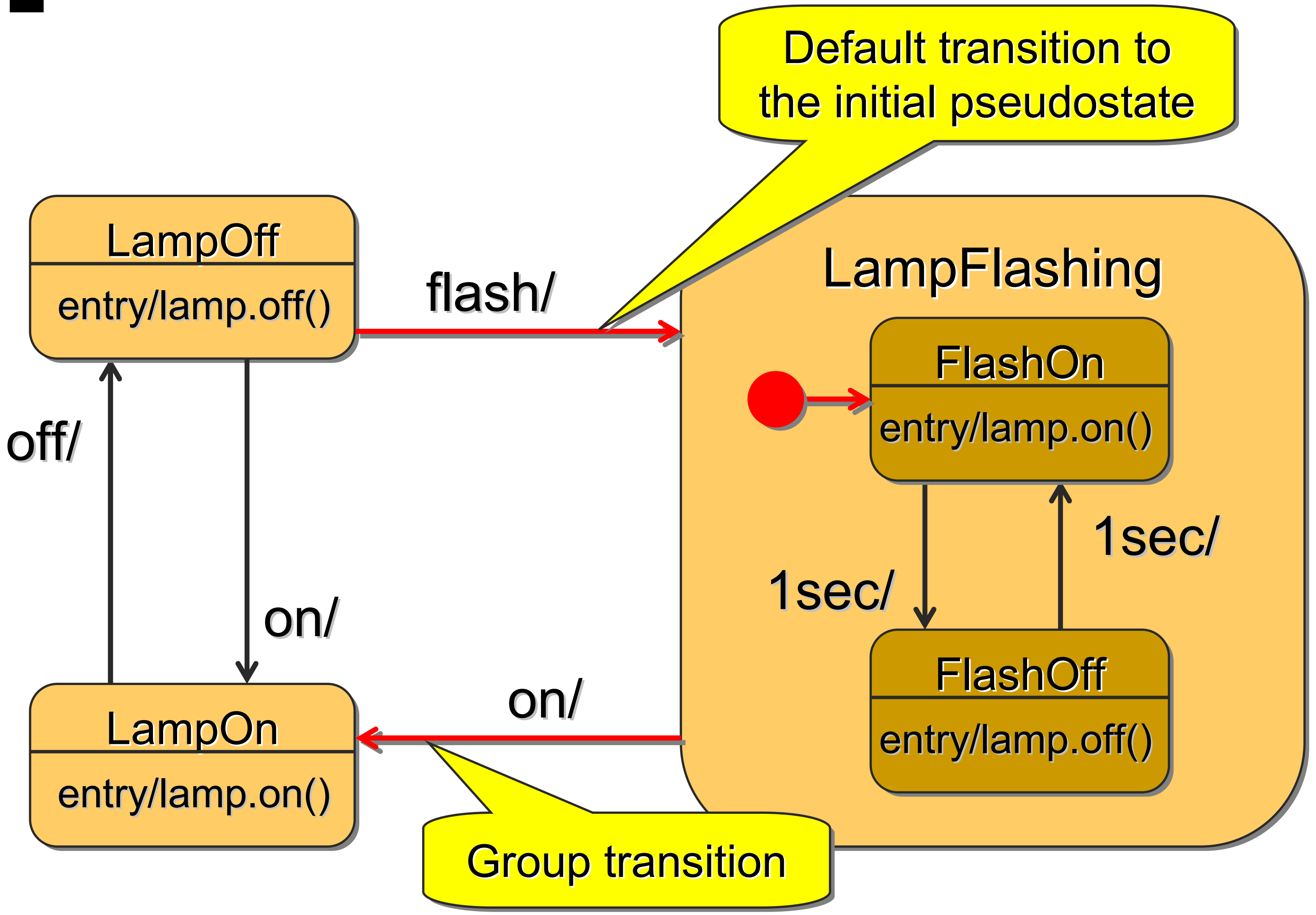

bid [value < 100] /reject
Selling
bid [value >= 200] /sell
Happy
bid [(value >= 100) & (value < 200)] /sell
Unhappy

Hierarchical State Machines

- Gradual attack on complexity
- States decomposed into state machines

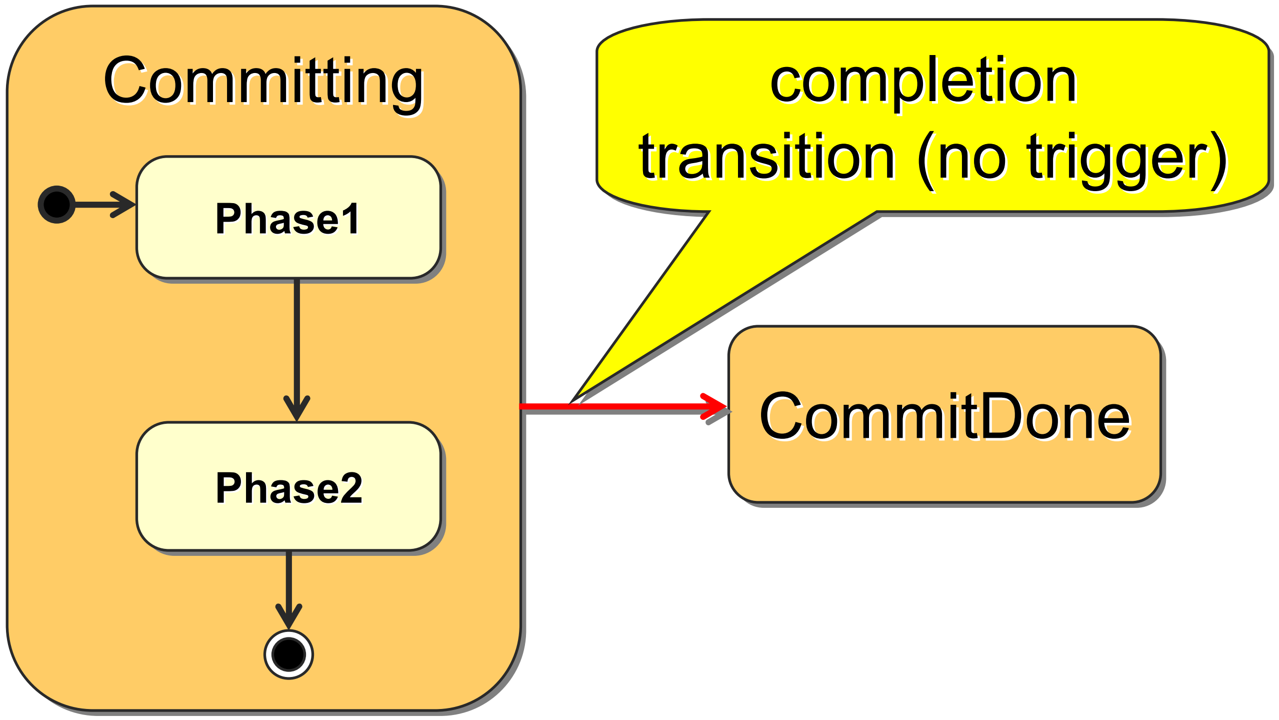


Group Transitions



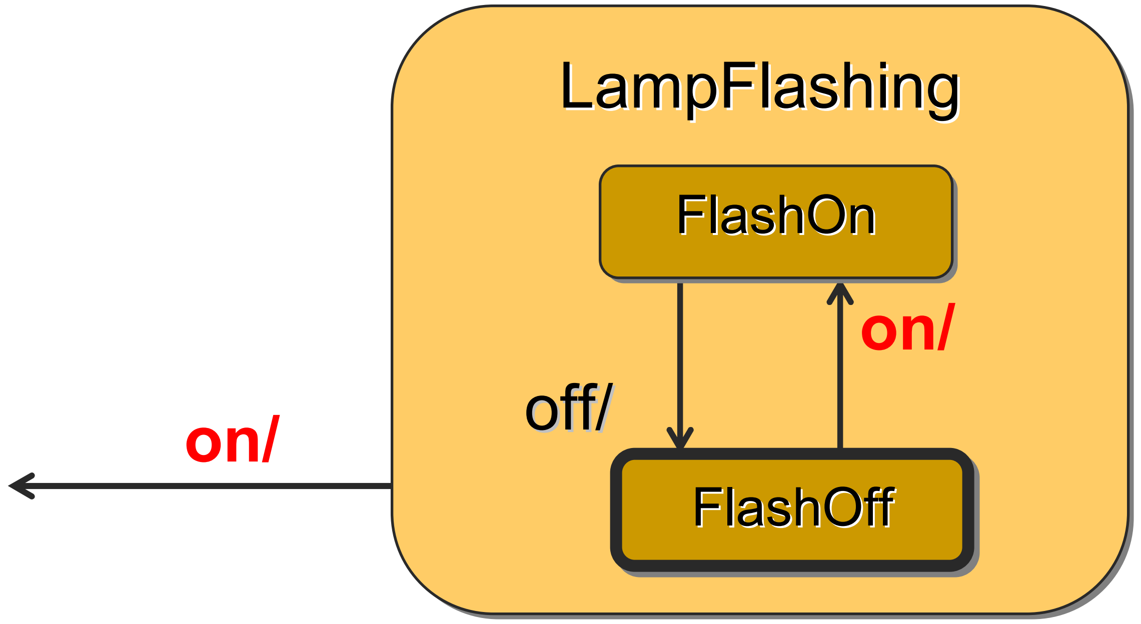
Completion Transitions

- Triggered automatically when a nested state machine reaches the final state

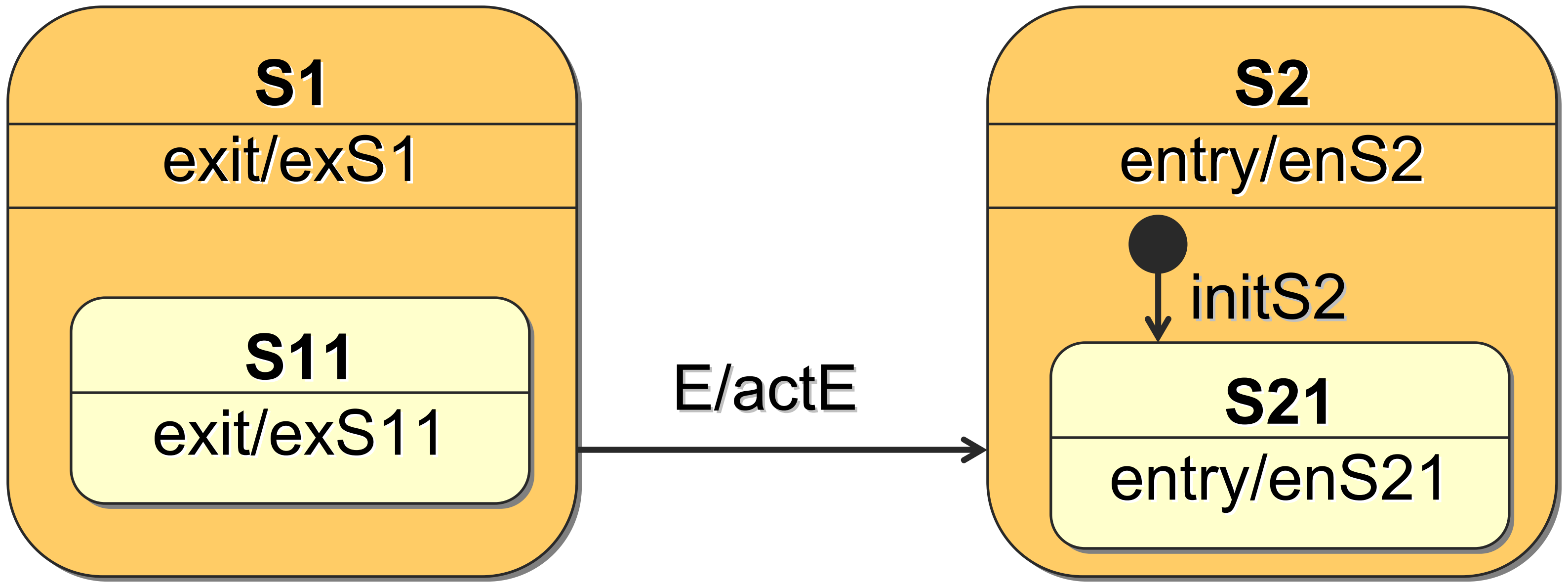


Triggering Rules

- Two or more transitions may have same trigger
  - Innermost takes precedence
  - Event is discarded whether or not a transition is triggered



Order of Actions (Complex Case)



The sequence of actions is as follows:

exS11 ⇨ exS1 ⇨ actE ⇨ enS2 ⇨ initS2 ⇨ enS21