State Diagrams

Software Requirements and Design
T-216-GHOH

Skúli Arnlaugsson | 14. október 2019

State diagrams

- **State diagrams** describe the states a system can be in, and how it moves from one state to another

- State diagrams are good at describing the behavior of an object across several use cases

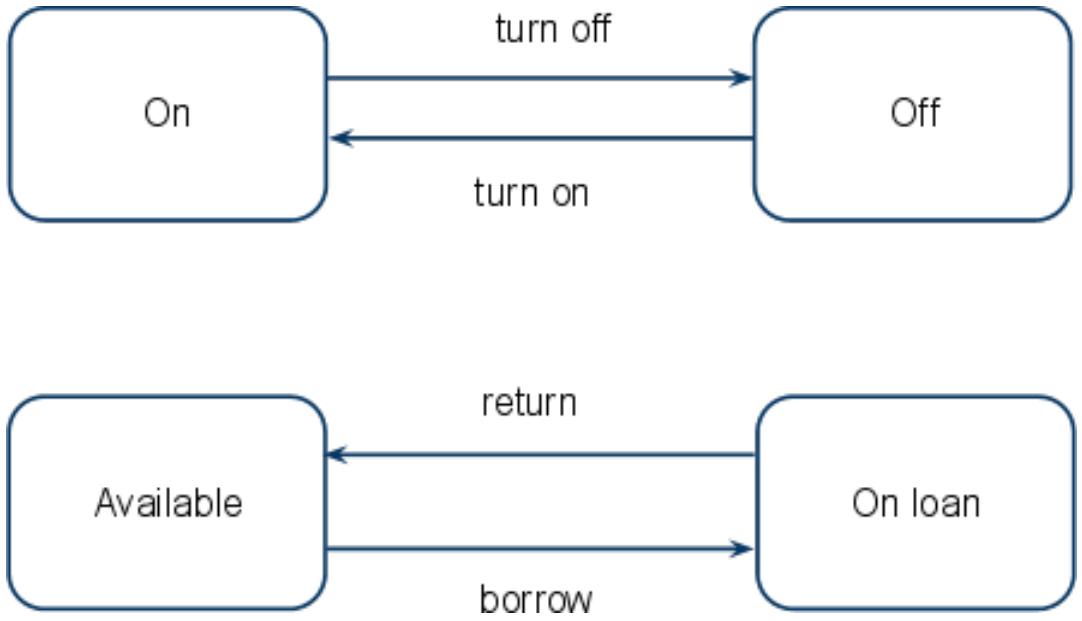
This lecture

- Purpose of State Diagrams
- State Diagram notations
- Reserved action names
- Sub-states
- Concurrent states
- History states
- Examples

Purpose of a state diagrams

- We use it to state the events responsible for change in state.
- We use it to model the dynamic behavior of the system.
- To understand the reaction of objects/classes to internal or external actions.

Simple state diagrams



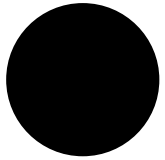
State

- States represent situations during the life of an object.
- We use a rounded rectangle to represent a state.
- A state represents the conditions or circumstances of an object of a class at an instant of time.

Initial State

- Initial state is the one that an object is in when it is first created

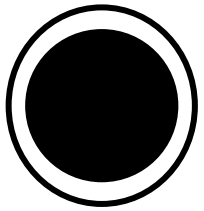
- We use a black filled circle represent initial state



Final State

- A final state is one in which no transitions lead out of.

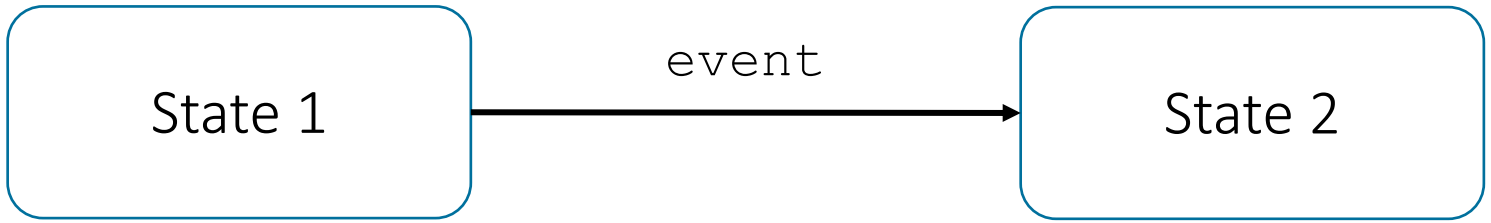
- We use a filled circle within a circle notation to represent the final state in a state machine diagram.



Transition

- We use a solid arrow to represent the transition or change of control from one state to another.

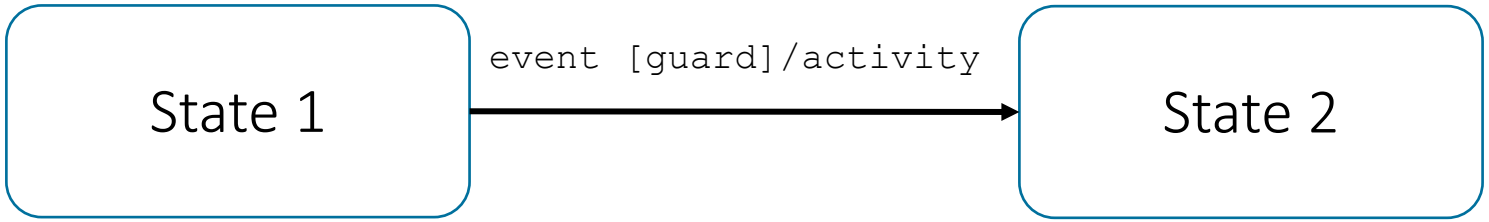
- The arrow is labelled with the event which causes the change in state.



Transition, contd.

- Each transition has a label that comes in three parts
  `event [guard]/activity`

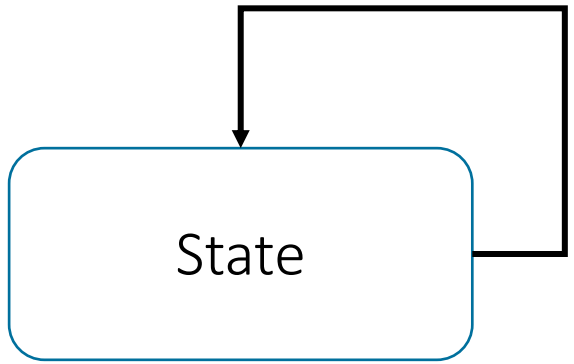
  - **Event**: a single event that triggers a potential change of state
  - **Guard:** a Boolean condition that must be true for the transition to be taken
  - **Activity:** some behavior that's executed during the transition



Self transitioning

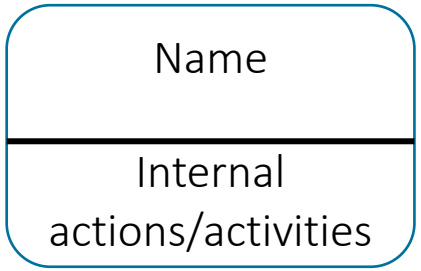
- We use a solid arrow pointing back to the state itself to represent a self transition.

- There might be scenarios when the state of the object does not change upon the occurrence of an event. We use self transitions to represent such cases.



State

- The **name** of the state (which describes the state) is placed inside the rectangle
- The state may have **internal actions/activities**

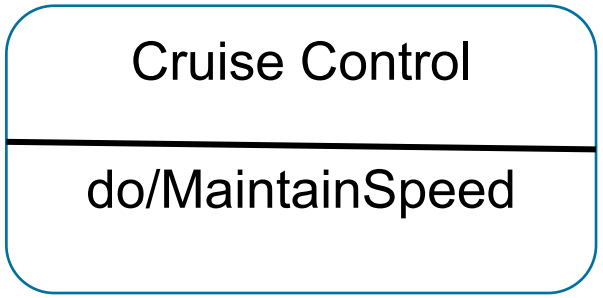


State internal actions

- States can react to events without transition
  `[action-name] / [action-definition]`

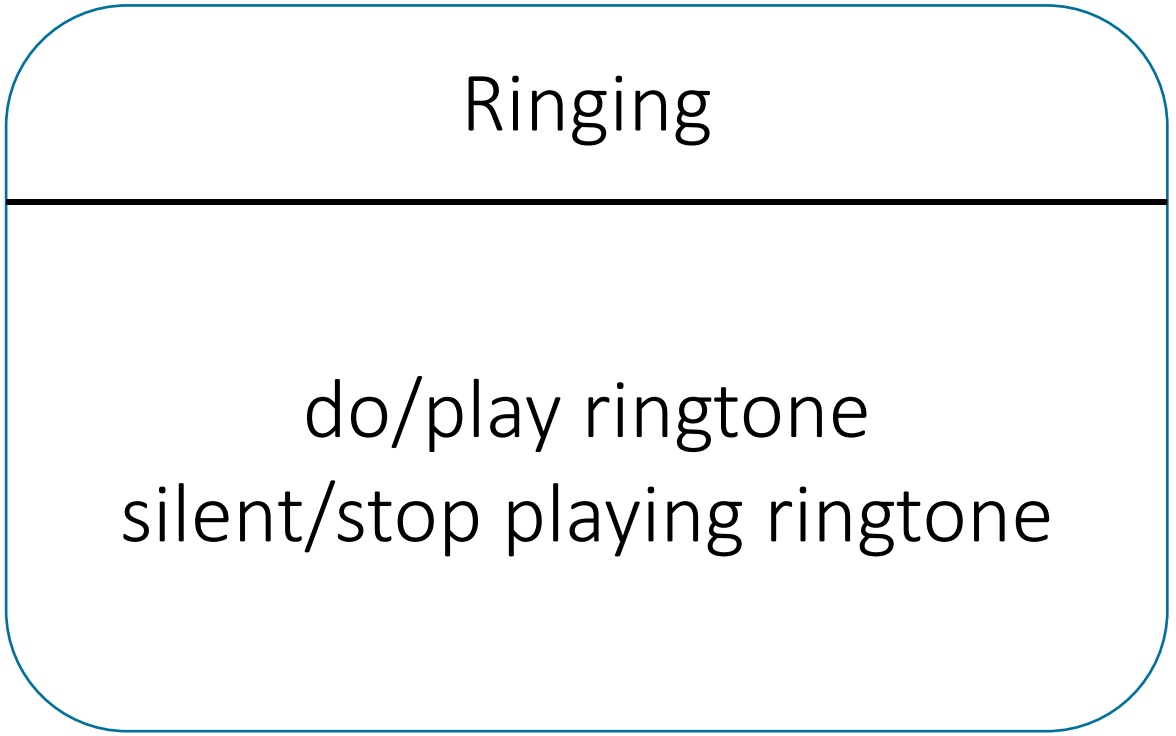
- **action-name**
  - specifies what initiates the action, could be an event

- **action-definition**
  - specifies what should happen when this event occurs



State internal actions

- Internal actions are performed after the ”entry” section and are aborted when the state is left



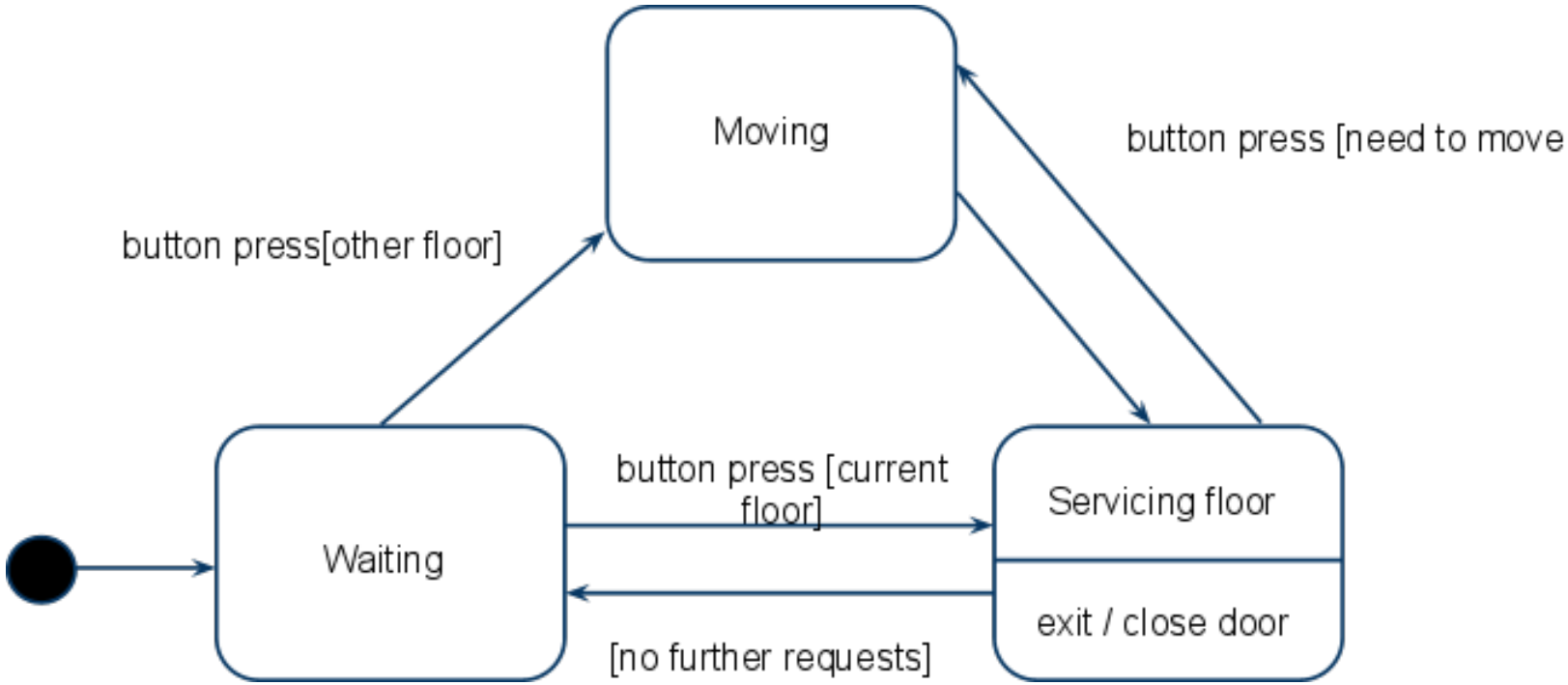
Reserved action names

- A couple of action names are reserved for specific events:

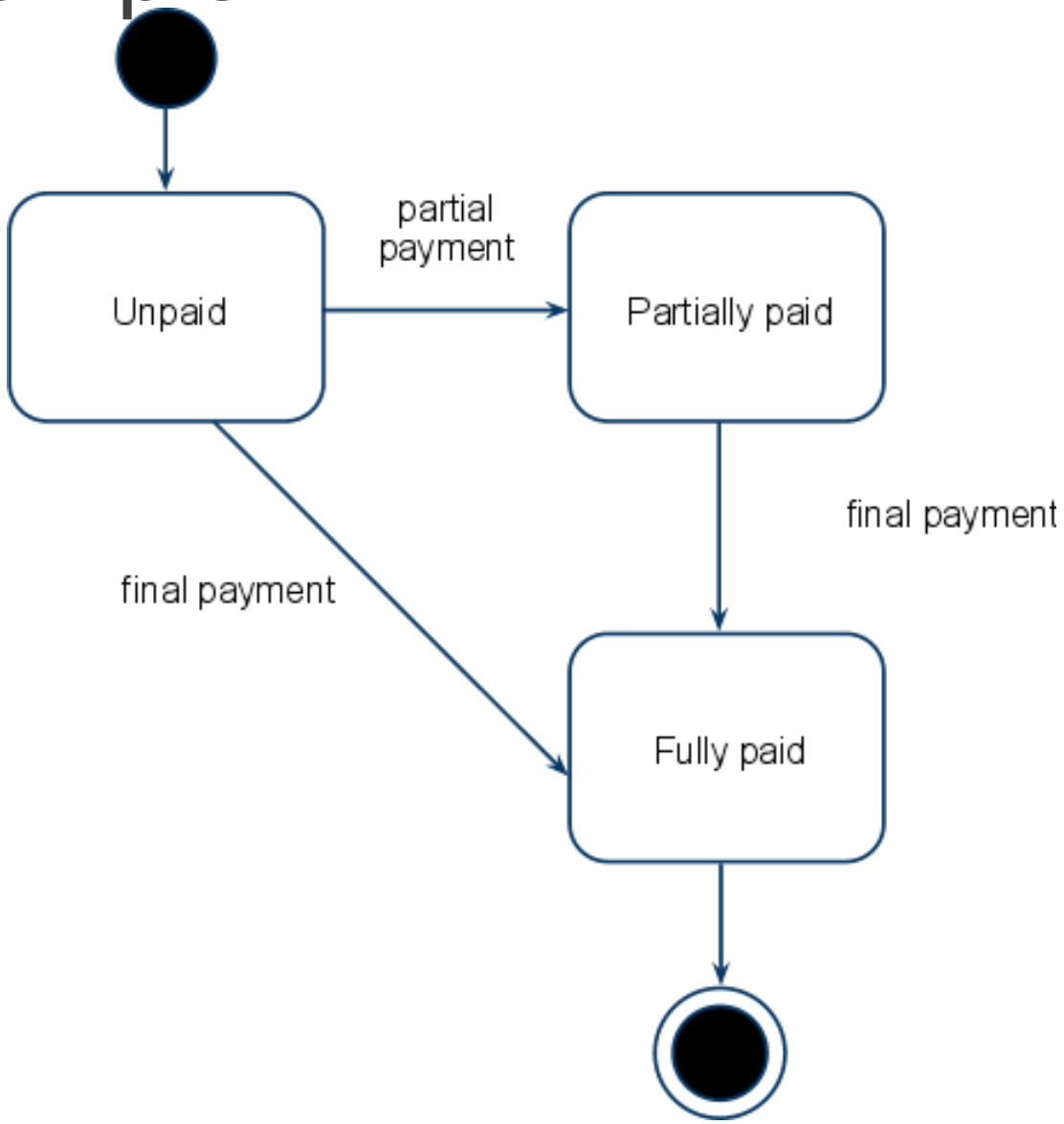
- **entry**
  - defines what should happen when the state is entered
- **exit**
  - defines what should happen when the state is exited
- **do**
  - what should happen while the state is active
- **include**
  - used if a sub-state machine is included in the state

Example

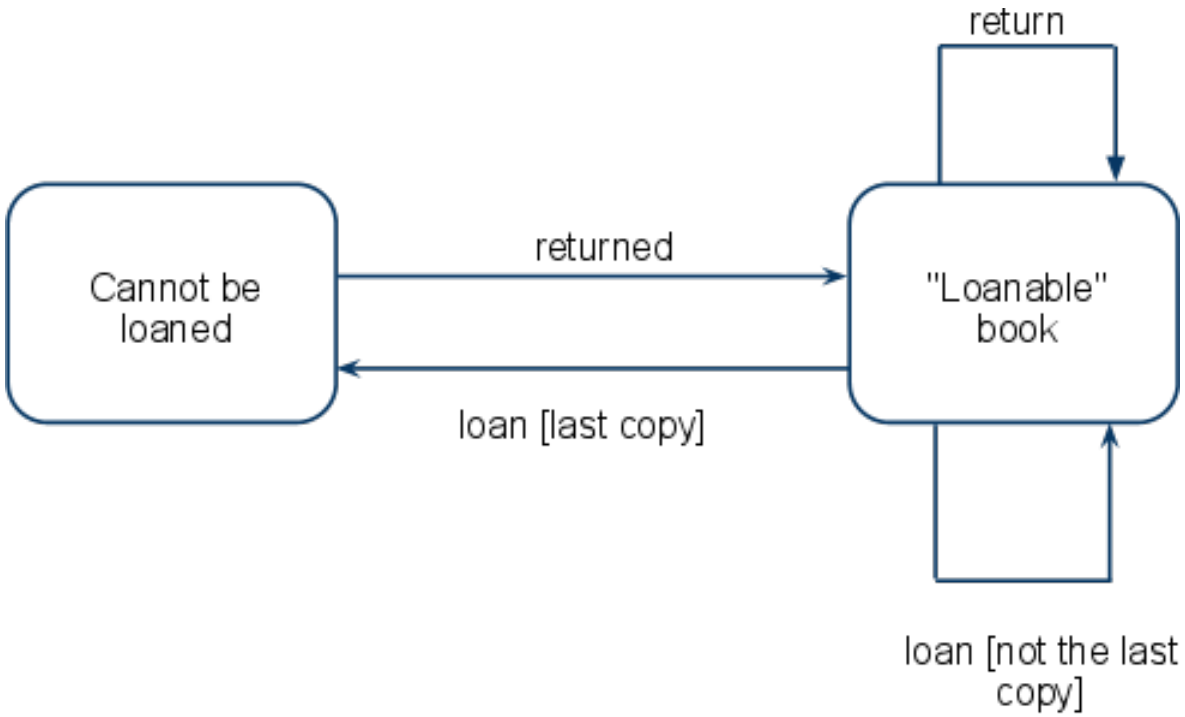
This is a simple state diagram for an elevator:



Another example



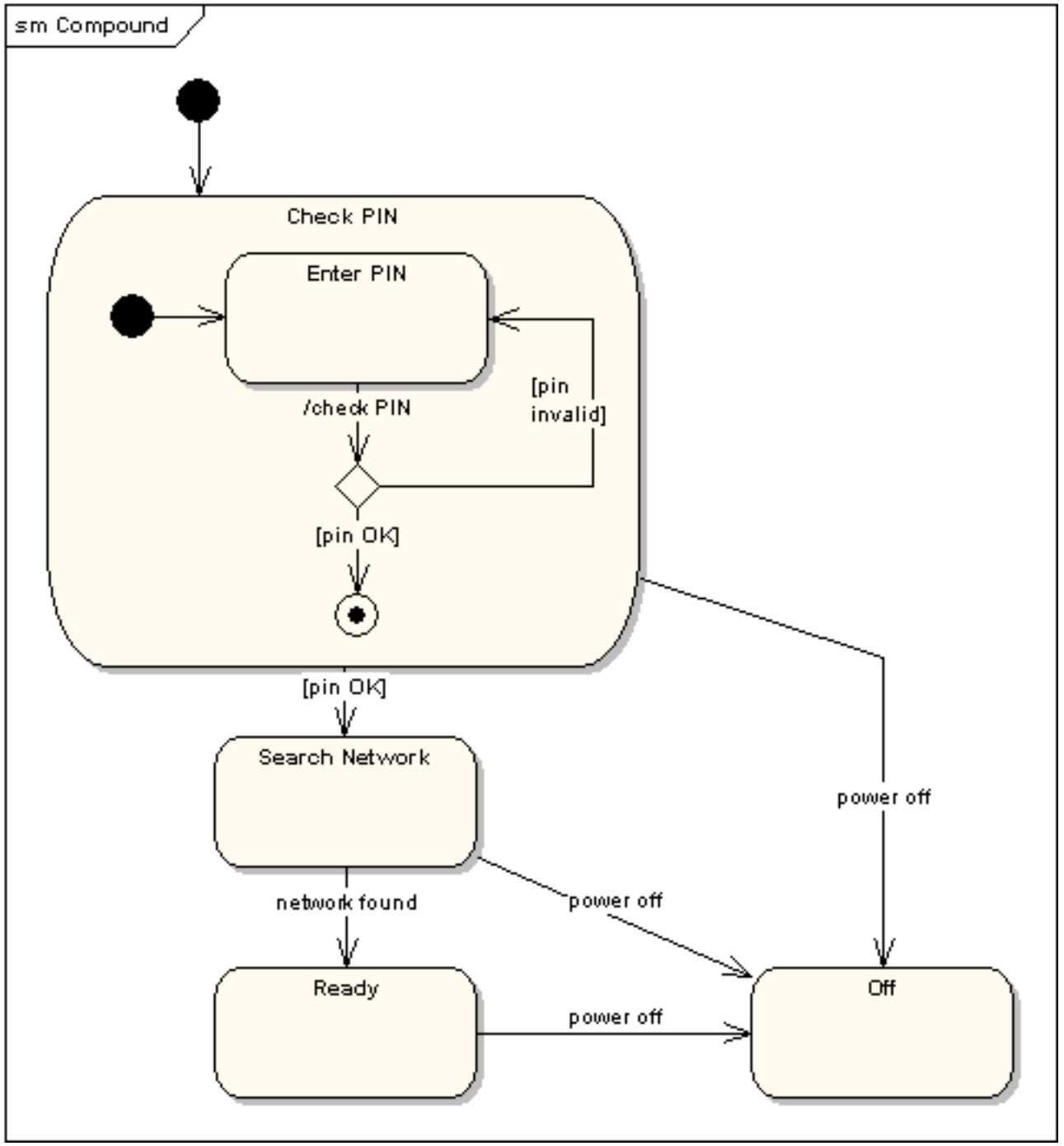
Another example - with guards



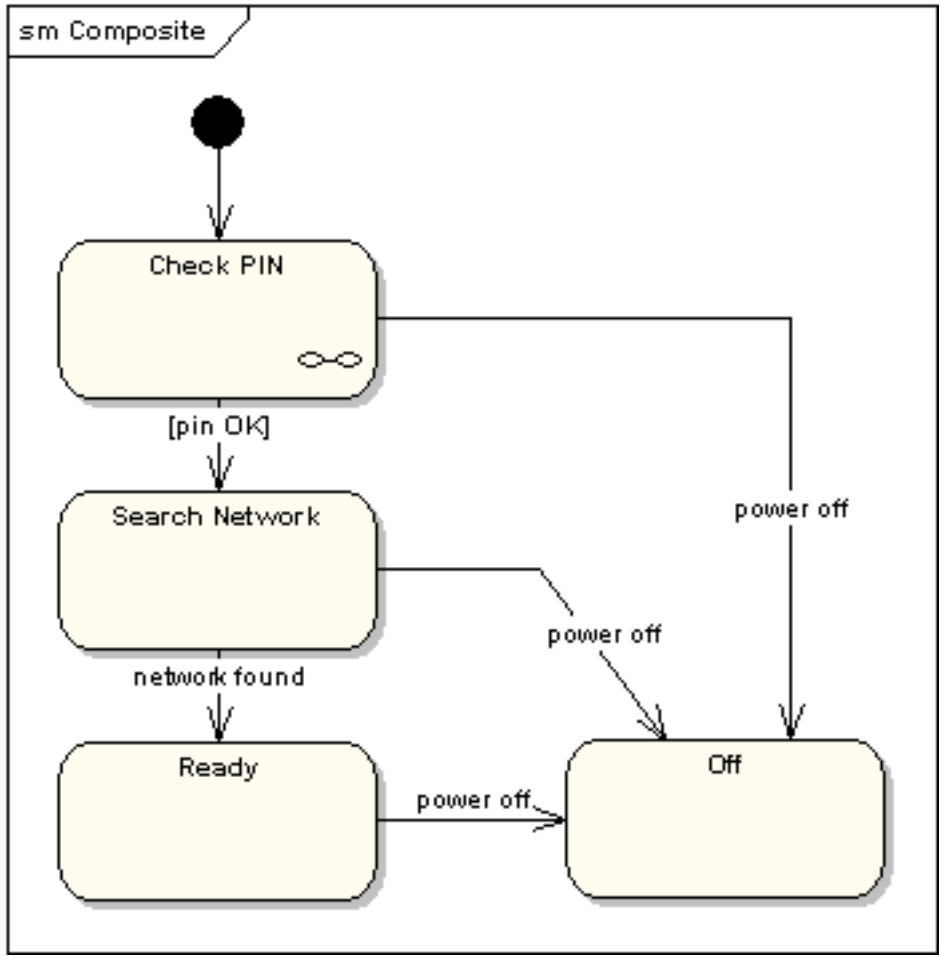
Sub-states

- Substates are used to represent a state machine inside another state machine
- Certain behaviour can then be represented with a separate state diagram
- The parent state is sometimes called superstate

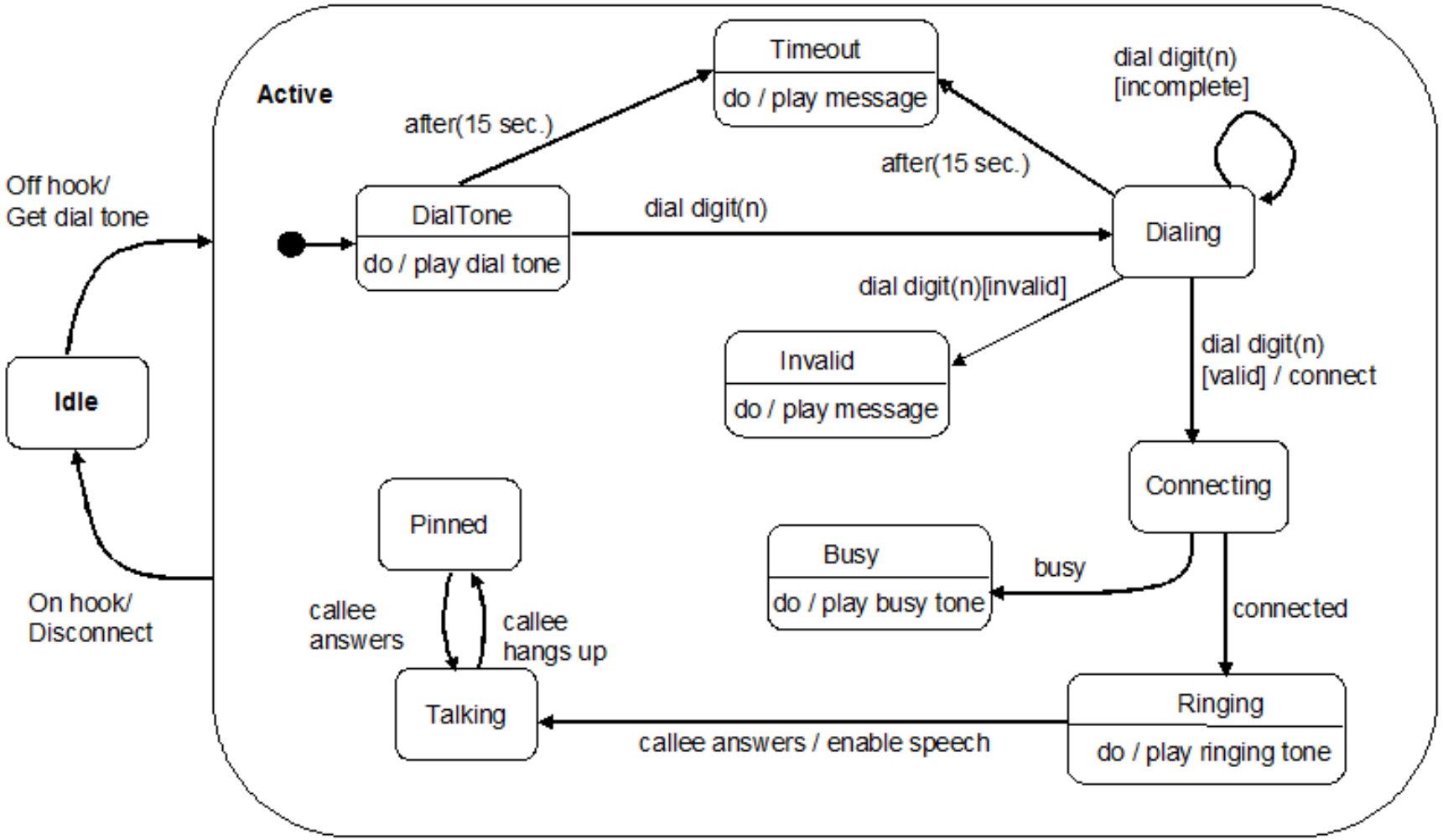
Compound State



Composite state

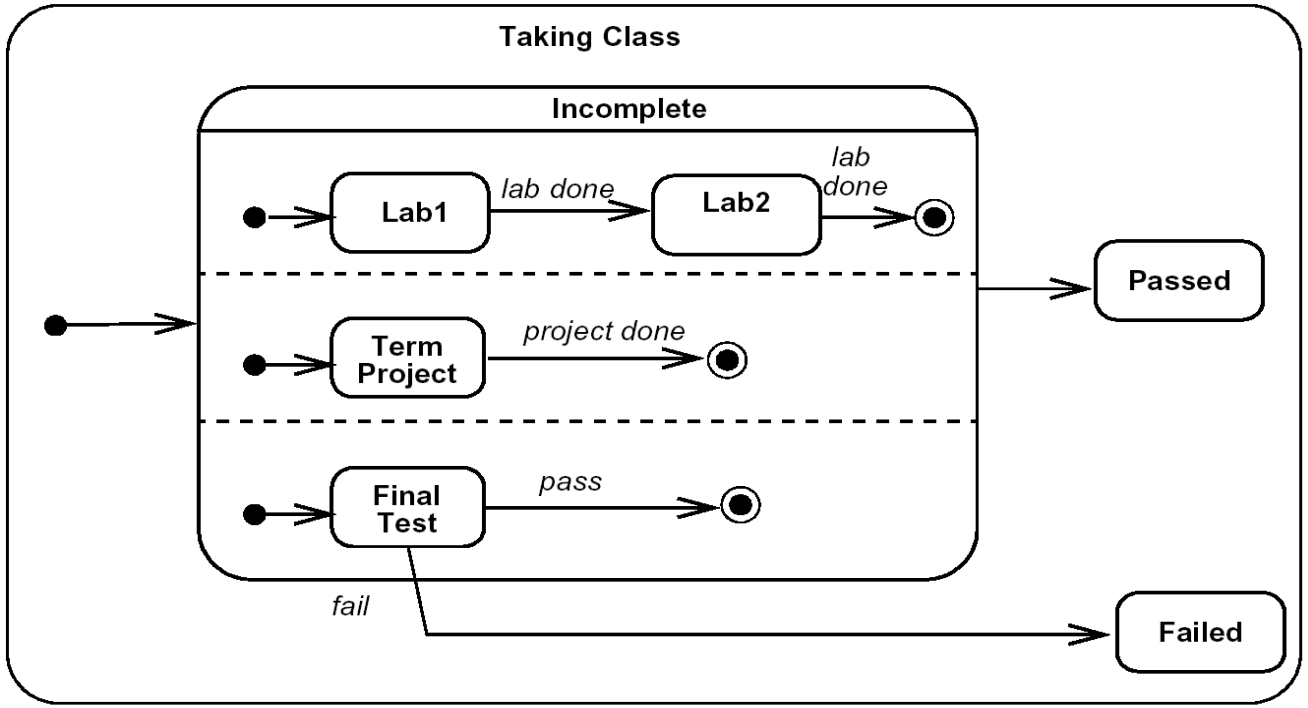


Example - phone state diagram



Concurrent states

- When two or more states are entered simultaneously, they are said to be concurrent
- Each state is in its separate "process", separated by a dotted line



History state

- A history state is represented with a circle and the letter H inside it

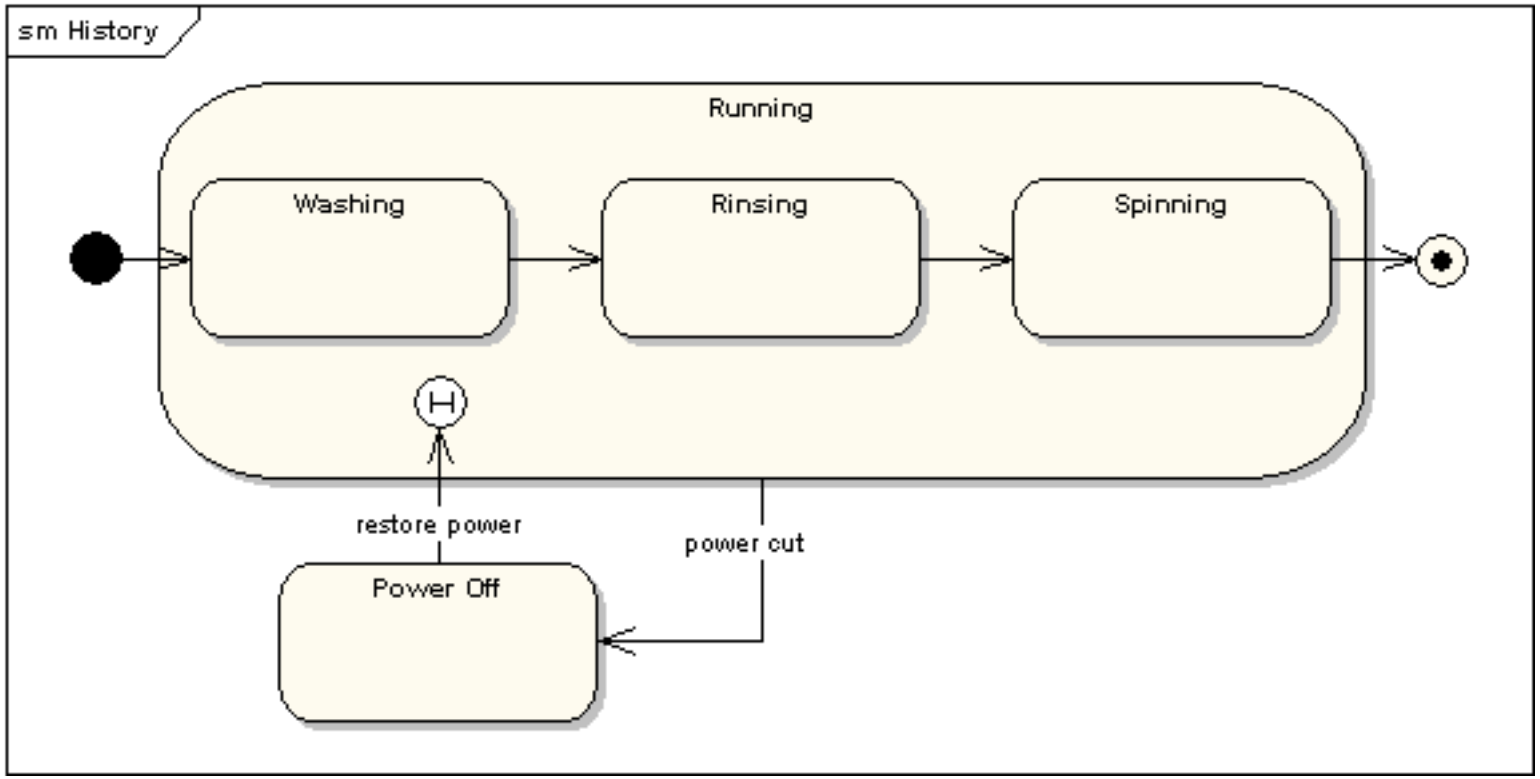
- Represents a state with memory - i.e. when this state is entered, it will activate the state within the superstate which was left the last time

History state

- **Example**: user is in a certain state, but selects a "Help" function. When he leaves the help, he should go back to the state he was previously in, no matter what that state was

- When a history state is entered, the last state will be reactivated, as if it would be entered for the first time

History state



Deep history state

- A regular history state will only transfer execution to the last state of the same superstate (or parent state), i.e. it will not transfer to a substate within that state
- A deep history state will however do exactly that
- It is represented with an asterisk next to the letter H