

Getting Dynamic, State Machine Diagram

PB007 Software Engineering I

Lukáš Daubner
daubner@mail.muni.cz

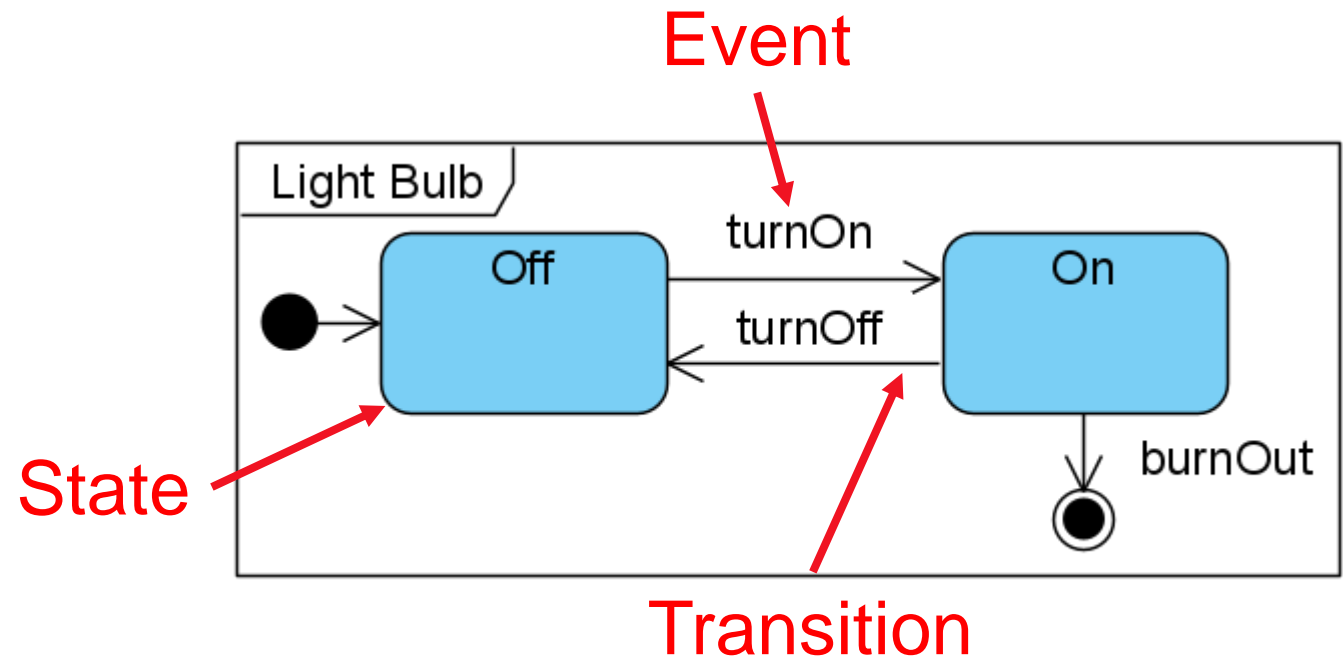
State Machine Diagram

— Models the dynamic behavior (life cycle) of one subject

- Class instantiation (Object)
- Use Case
- System
- Subsystem
- Component
- ...

— Main components are:

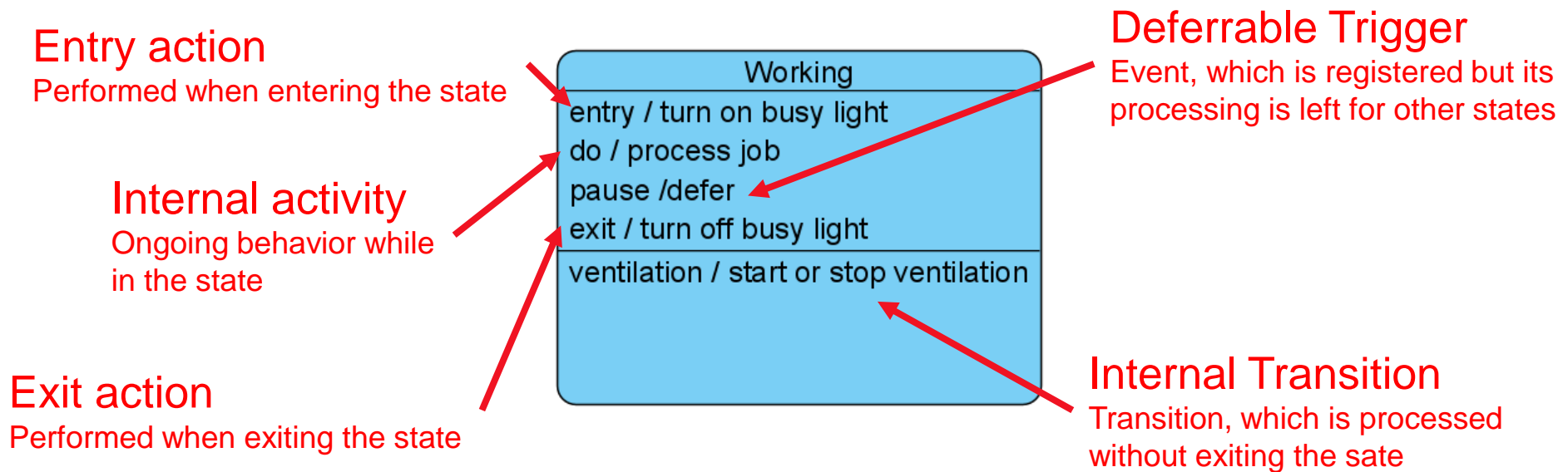
- States
- Transitions
- Events



State Machine Diagram

States

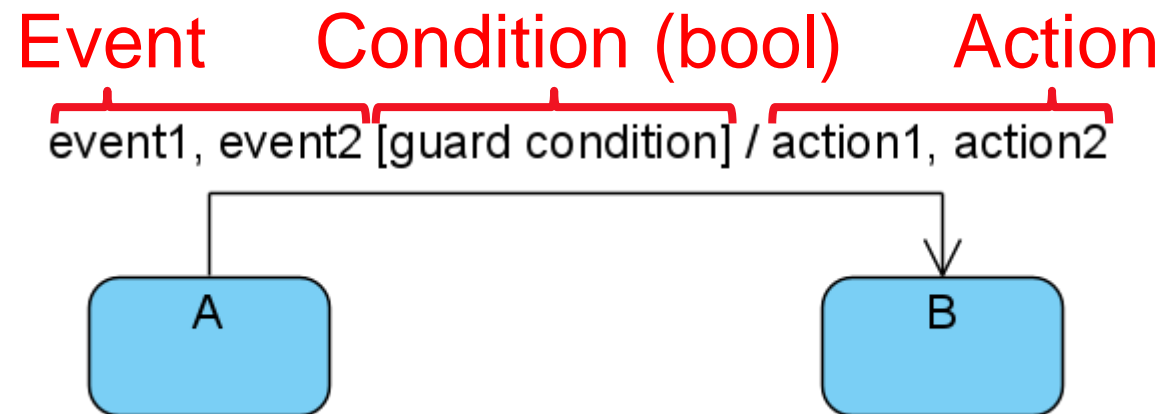
- Represents semantically important situation
- In case of (OOP) object, it is determined by attribute values, relations with others, and performed activity.



State Machine Diagram

Transitions

- Defines how to get from one state to another
- **Syntax:** *event [guard condition] / action*
- **Semantics:** At the occurrence of *event*, if the *guard condition* holds, perform *action* and go to the new state.



State Machine Diagram

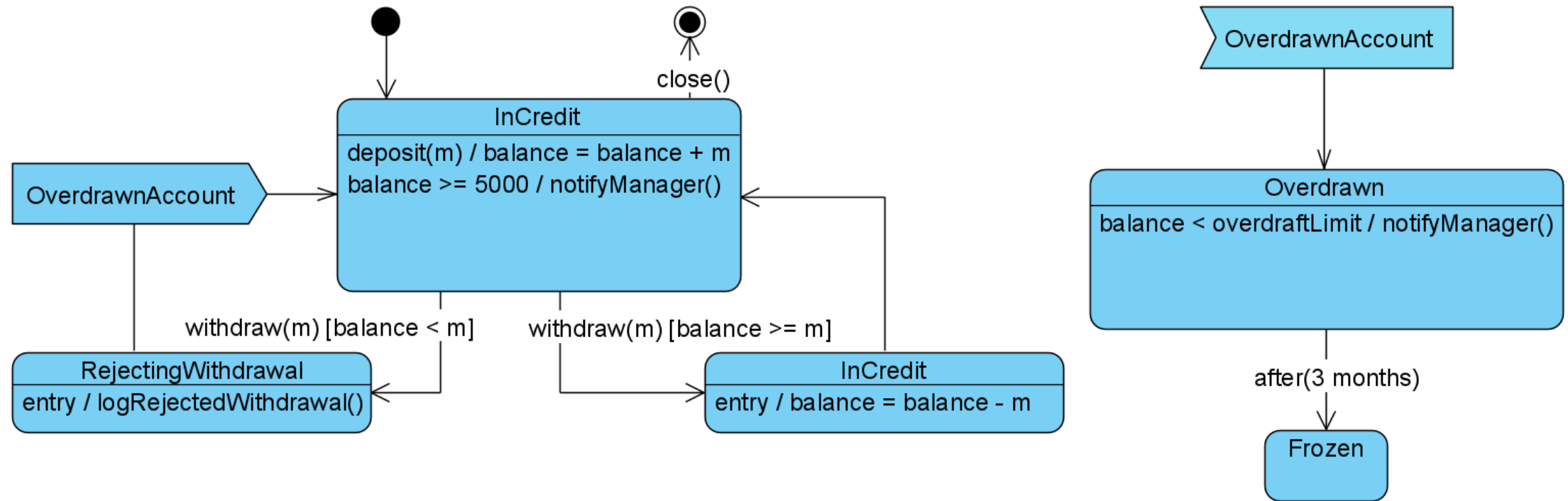
Events

- Stimulus on which the subject may react by changing the state or performing an operation.

- Types of events:
 - **Call event** – Calling operation of the subject.
 - **Signal event** – Asynchronous sending a receiving a signal between subjects
 - **Change event** – Boolean expression. The event occurs when the value is changed from false to true.
 - **Time event** – Event occur at a certain time t (*when(t)*) or after a certain time t (*after(t)*).

State Machine Diagram

Events

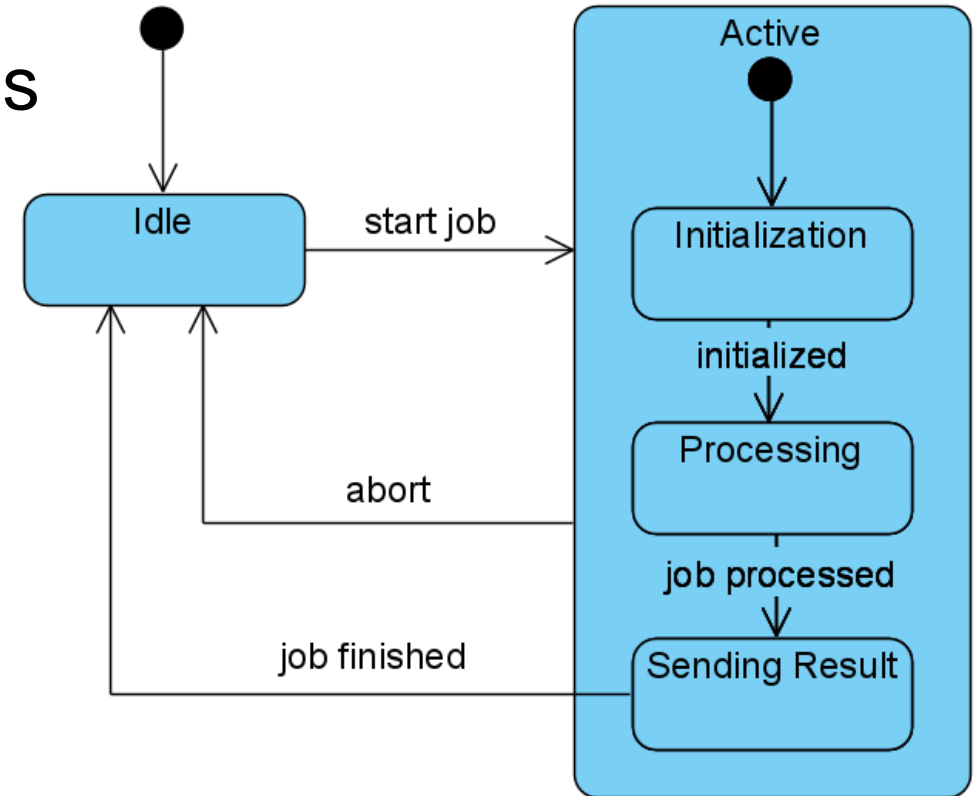


(excerpt from diagram)

Composite States

Simple composite state

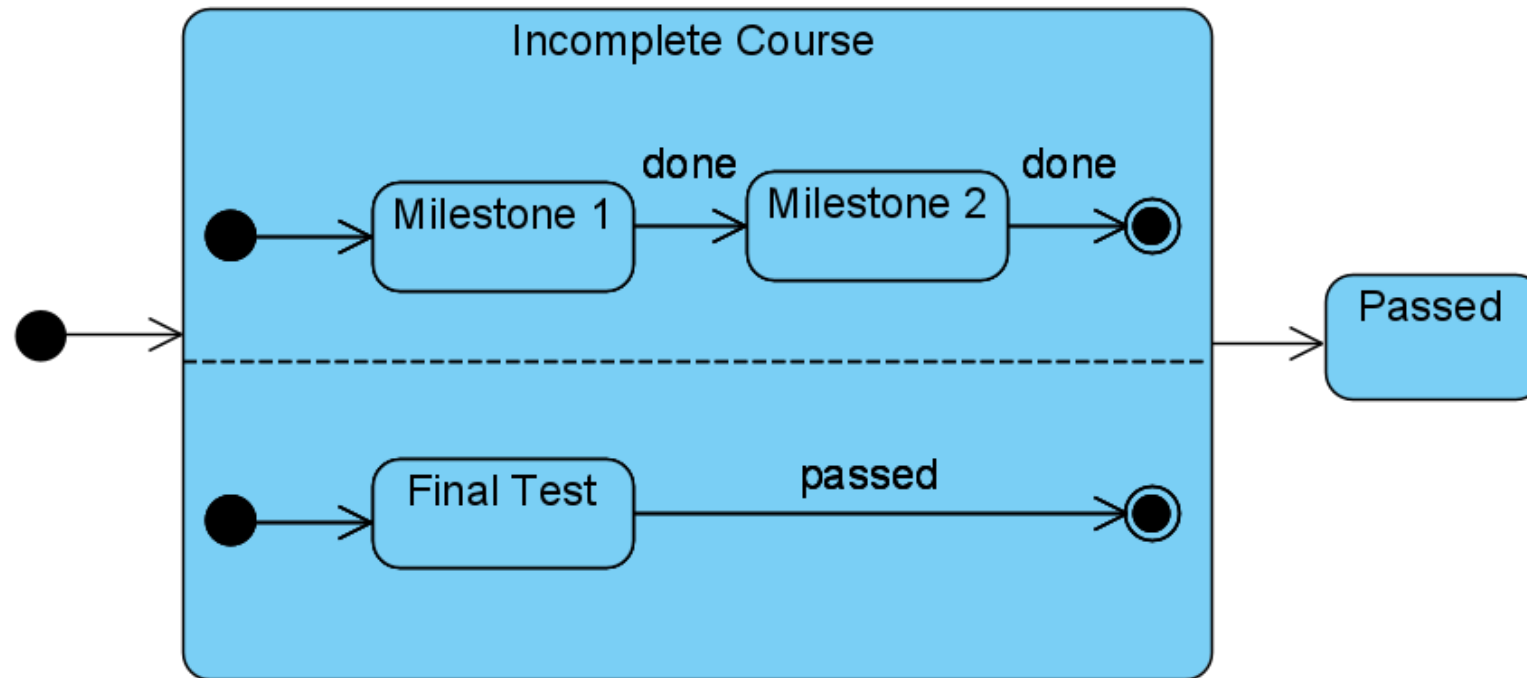
- Useful for simplifying the diagram
- Capturing inheritance between states
- Consist of a single region



Composite States

Orthogonal composite state

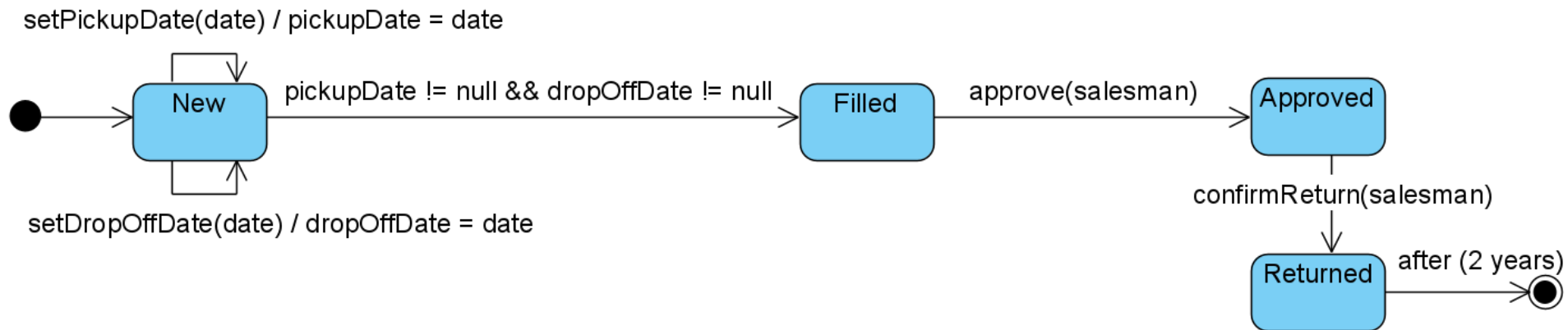
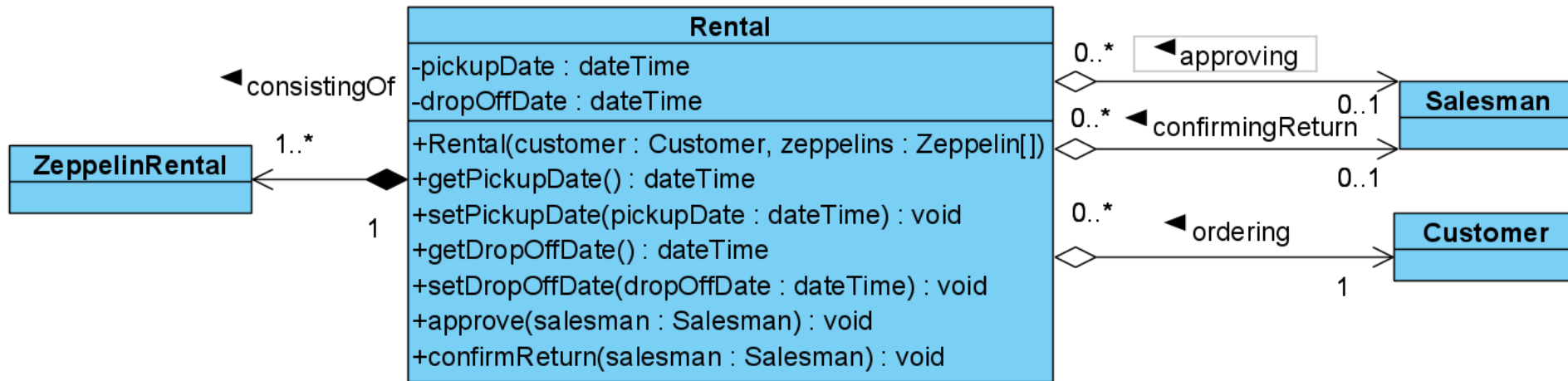
- Capturing parallel behavior
- Consist of a two and more regions



State Machine Diagram in OOP world

- In our case, state machine diagram is used to represent lifecycle of an object
- Context of the diagram is **only the instance** of a class from design class diagram
 - All methods and events must be supported by the design class diagram
- Initial transition means calling the constructor
- Final transition means deleting the object from system
- Object saves its state even outside main memory (persistence)

State Machine Diagram in OOP world



Task for this week

You gotta do what you gotta do

- Process the feedback
- Choose a suitable object for modeling
 - Something with non-trivial lifecycle
- Create a state machine diagram for this object
 - Revise design class diagram if needed
- Do your part in peer review