UML

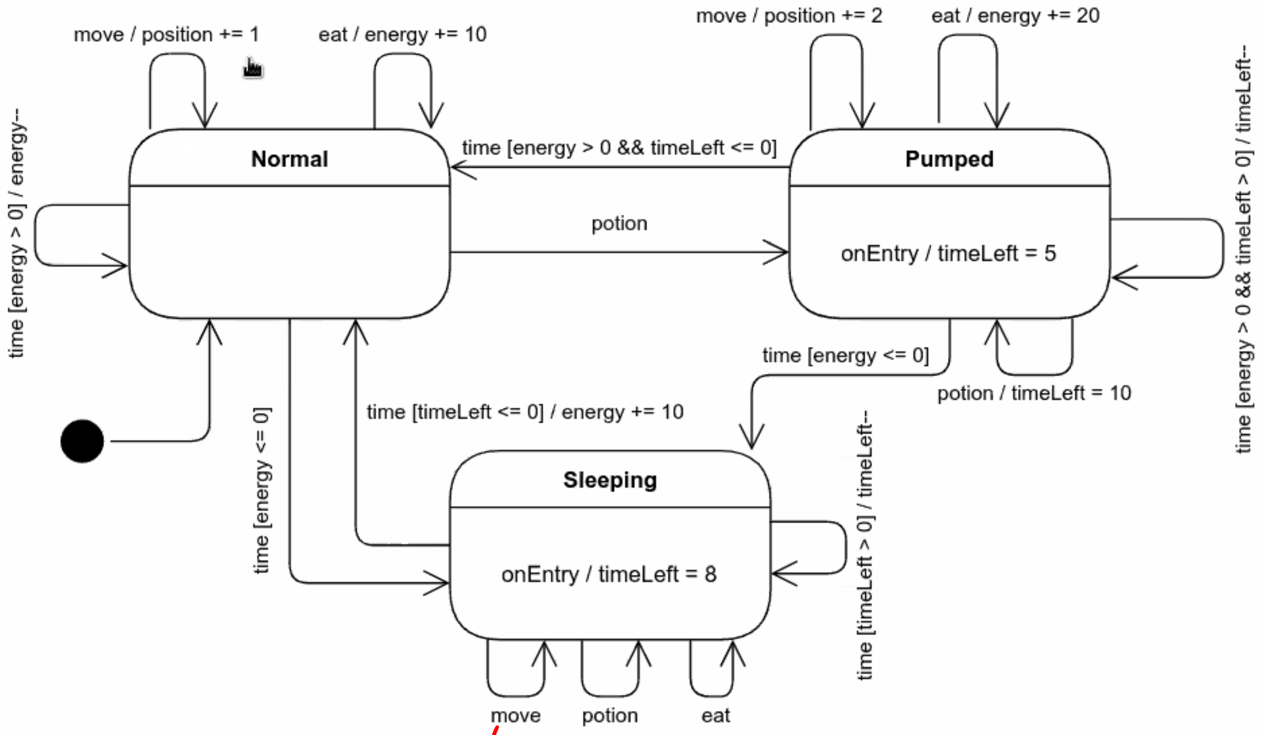
State Diagrams

André Restivo

Exemplo

State

Sometime behaviors can depend on the current state of an object, as if the object could change class during runtime. We can represent this in the form of a state diagram.



→ decrementa, mas está no mesmo estado

→ As guardas têm de ser mutuamente exclusivas, não podem ocorrer + que 1 ao mesmo tempo

Quando dorme, não pode fazer nada disto, mantém a dormir

Introduction

Types of Diagrams

In UML, there are two basic categories of diagrams:

- **Structure** diagrams show the static structure of the system being modeled: *class*, *component*, *deployment*, *object* diagrams, …
- **Behavioral** diagrams show the dynamic behavior between the objects in the system: *activity*, *use case*, *communication*, *state machine*, **sequence** diagrams, …

State Diagrams

UML State machine diagrams are behavioral diagrams which show the discrete behavior of a part of designed system through finite state transitions.

They are particularly useful when defining the behavior of the State Pattern.

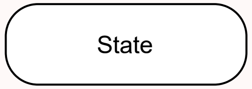
While UML Sequence diagrams usually depict generic or specific instances of interactions, UML State diagrams aim to depict the **complete inner-workings** of a system.

- O que acontece de um estado para outro
- O que faz mudar

Basic Components

State

A state is denoted by a round-cornered rectangle with the name of the state written inside it.

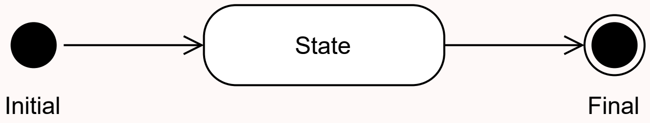


Initial and Final States

The initial state is denoted by a filled black circle.

The final state is denoted by a circle with a dot inside.

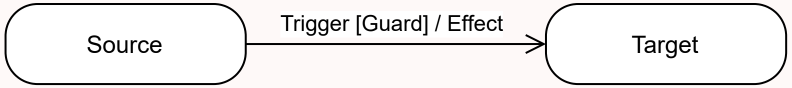
Both can be labeled with a name.



Transitions

Transitions from one state to the next are denoted by lines with arrowheads.

A transition may have a trigger, a guard and an effect.

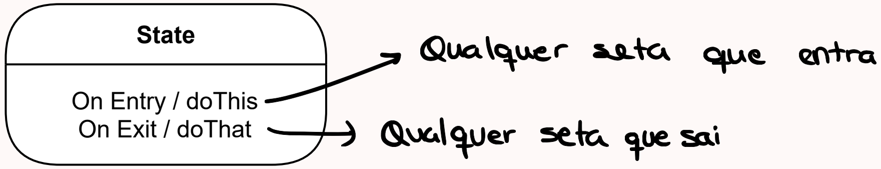


- **Trigger** is the cause of the transition, which could be a signal, an event, a change in some condition, or the passage of time. → Causa da transição
- **Guard**" is a condition which must be true in order for the trigger to cause the transition. → Condição para transitar
- **Effect** is an action which will be invoked directly on the object that owns the state machine as a result of the transition. → o que acontece quando a transição acontece

Actions

If the target state has many transitions arriving at it, and each transition had the same effect associated with it, it would be better to associate the effect with the target state rather than the transitions.

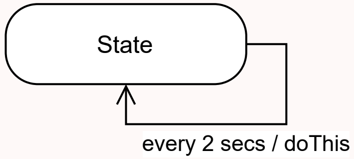
This can be done by defining an entry action for the state. There can also be exit actions, actions that happen when a certain event occurs, and actions that always occur.



Self-Transitions

A state can have a transition that returns to itself.

This is useful when an effect is associated with the transition without the state changing.

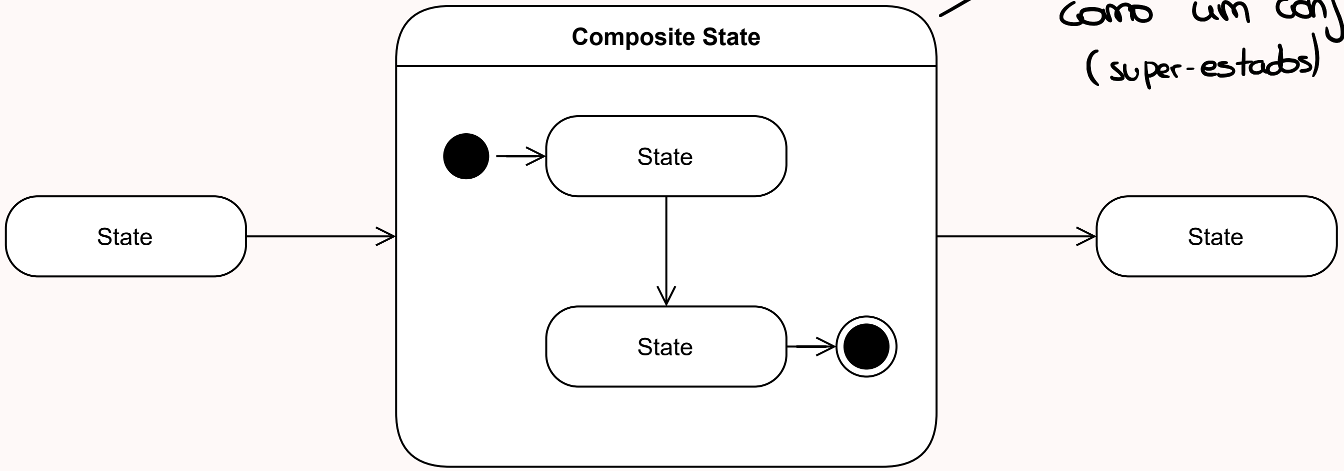


Composite

Composite

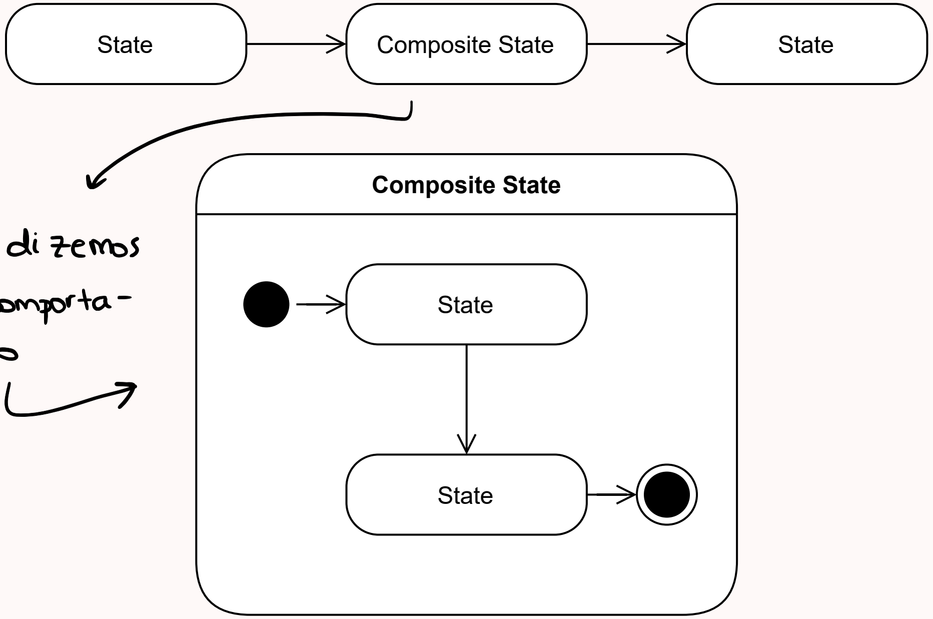
A state machine diagram may include sub-machine diagrams.

Podemos pensar em estados como um conjunto de estados (super-estados)



Separate Diagrams

It is also possible to simplify the diagram by showing composite states as separate diagrams.

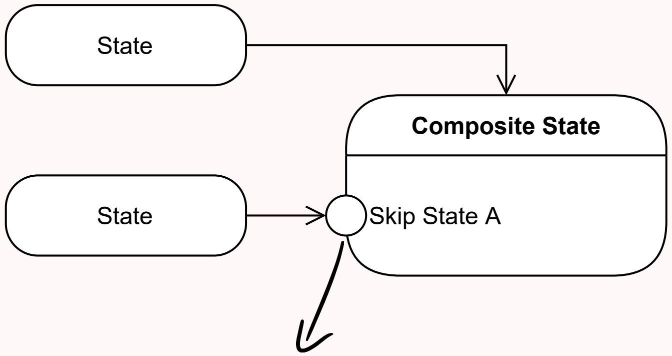


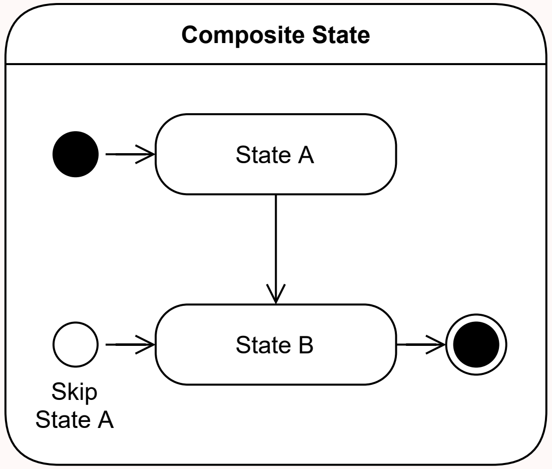
e depois dizemos o seu comportamento

Entry Point

Sometimes you do not want to enter a sub-machine at the normal initial state.

In those cases, you can have an alternative labeled entry point.



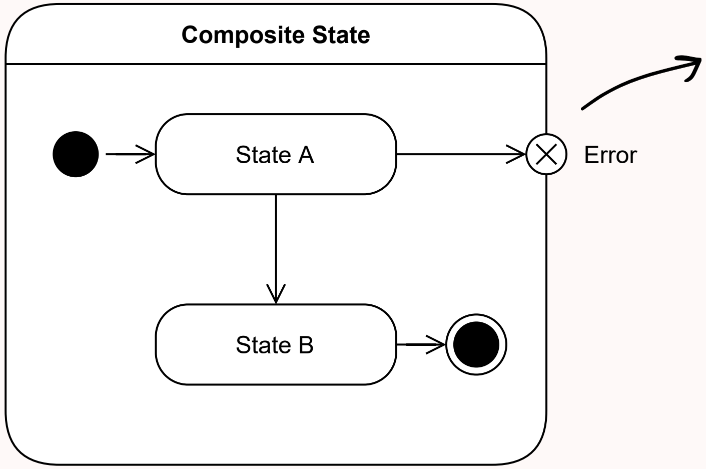


Podemos dizer que vai passar alguns estados à frente

Pode se dar na entrada e na saída

Exit Point

It is also possible to have named alternative exit points.



Saída para erro