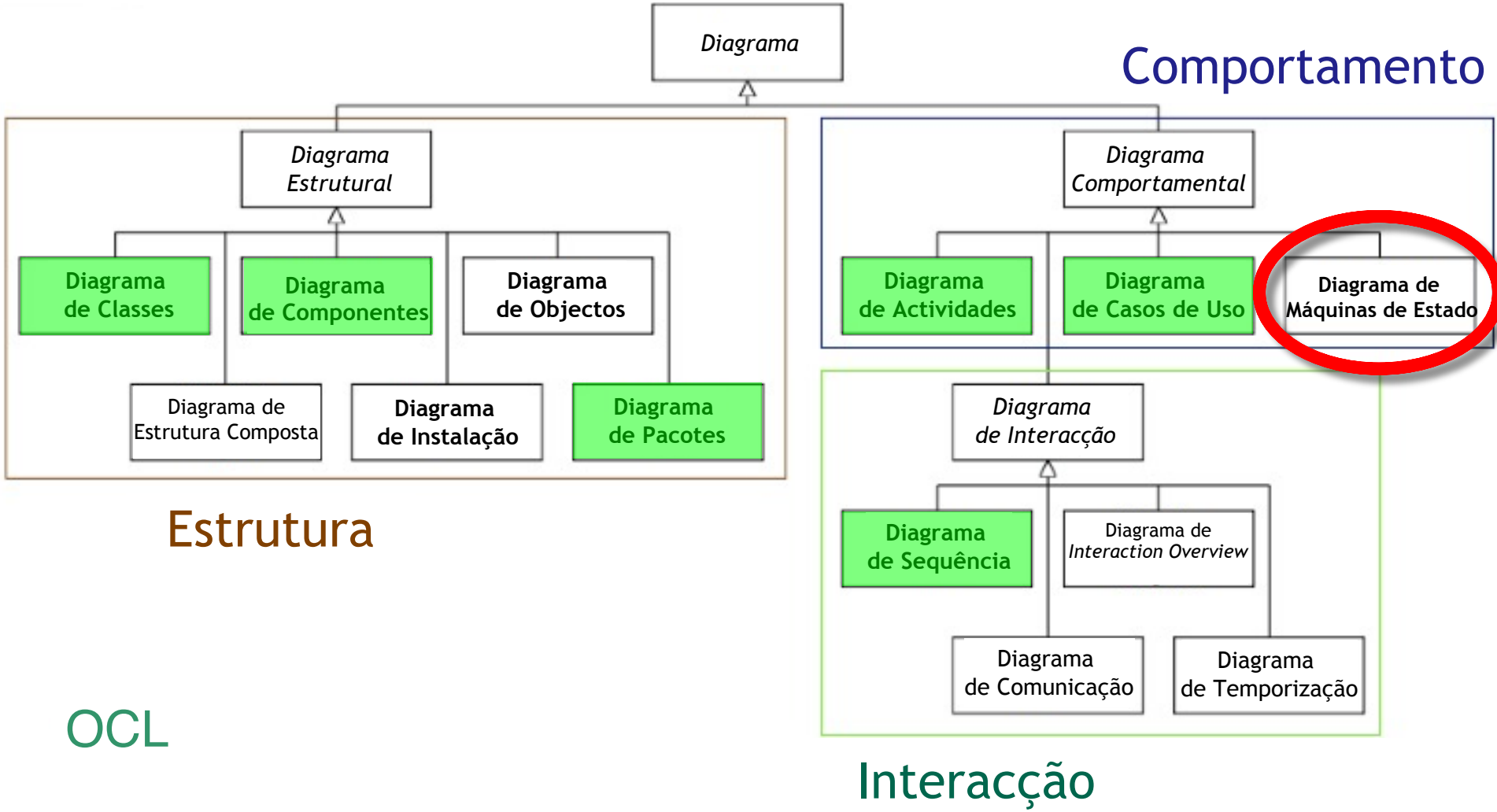
Modelação Comportamental
(Máquinas de Estado)

Desenvolvimento de Sistemas Software
José Creissac Campos / António Nestor Ribeiro

Diagramas da UML 2.x



Introdução aos Diagramas de Estado — Aplicação

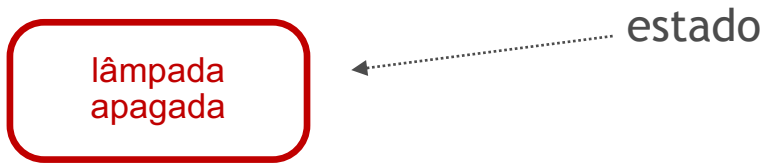
- Os Diagramas de Estado permitem modelar o comportamento de um dado objecto/sistema de forma global.
- A ênfase é colocada no estado do objecto/sistema — modelam-se todos os estados possíveis que o objecto/sistema atravessa em resposta aos eventos que podem ocorrer.

- Úteis para modelar:
  - O comportamento de um objecto de forma transversal aos use case do Sistema
  - O Sistema como um todo
- Devem utilizar-se para entidades/classes em que se torne necessário compreender o comportamento do objecto de forma global ao sistema.
  - Nem todas as entidades/classes vão necessitar de diagramas de estado.

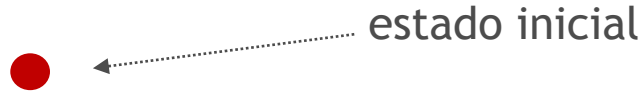
Diagramas de Estado

Notação base

- Estado – define uma possível estado do objecto (normalmente traduz-se em valores específicos dos seus atributos)



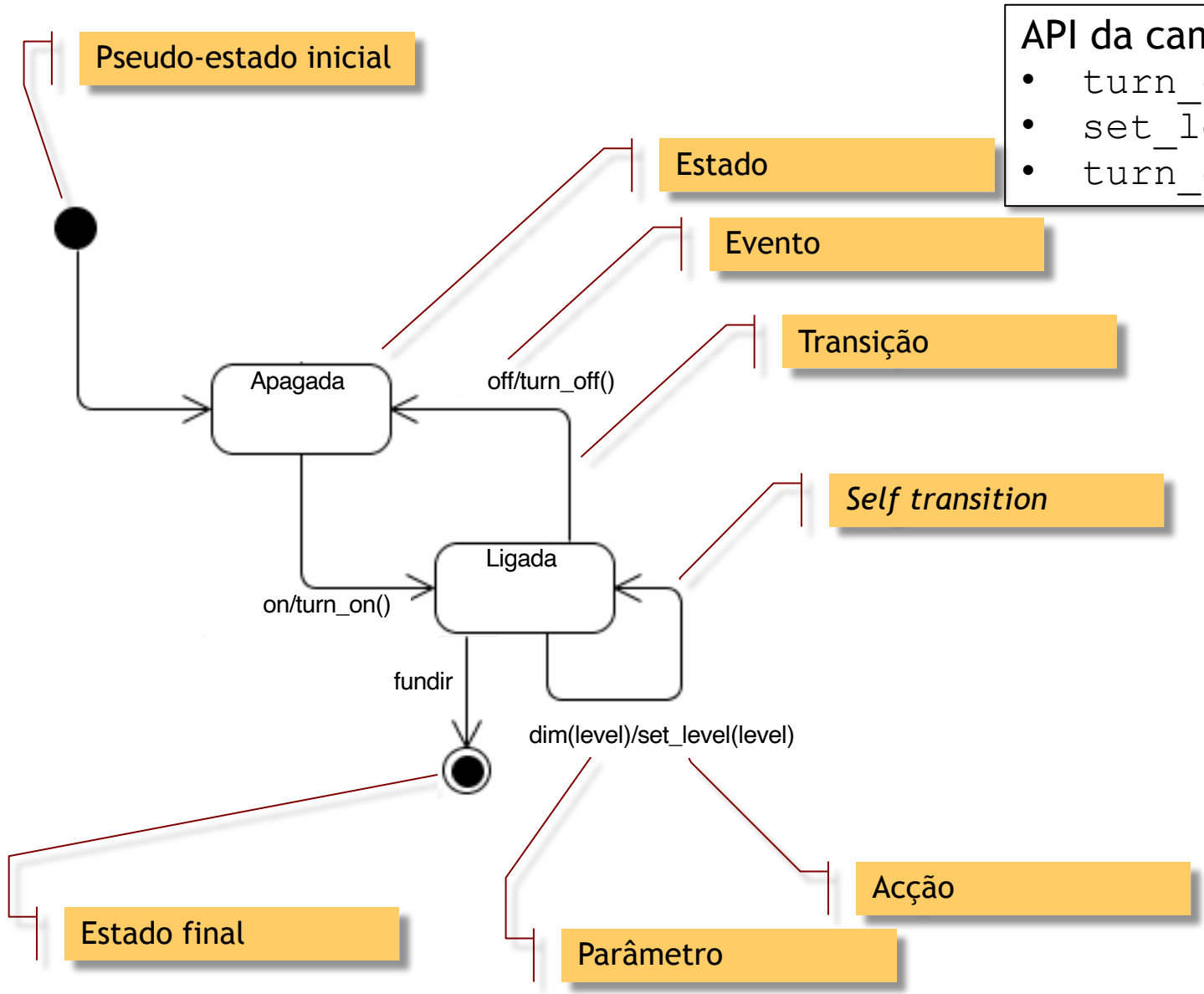
  - Estado inicial – estado do objecto quando é criado



  - Estado final – destruição do objecto

  - Transições – evento[guarda]/acção (todos são opcionais!)

Maquina de Estados básica

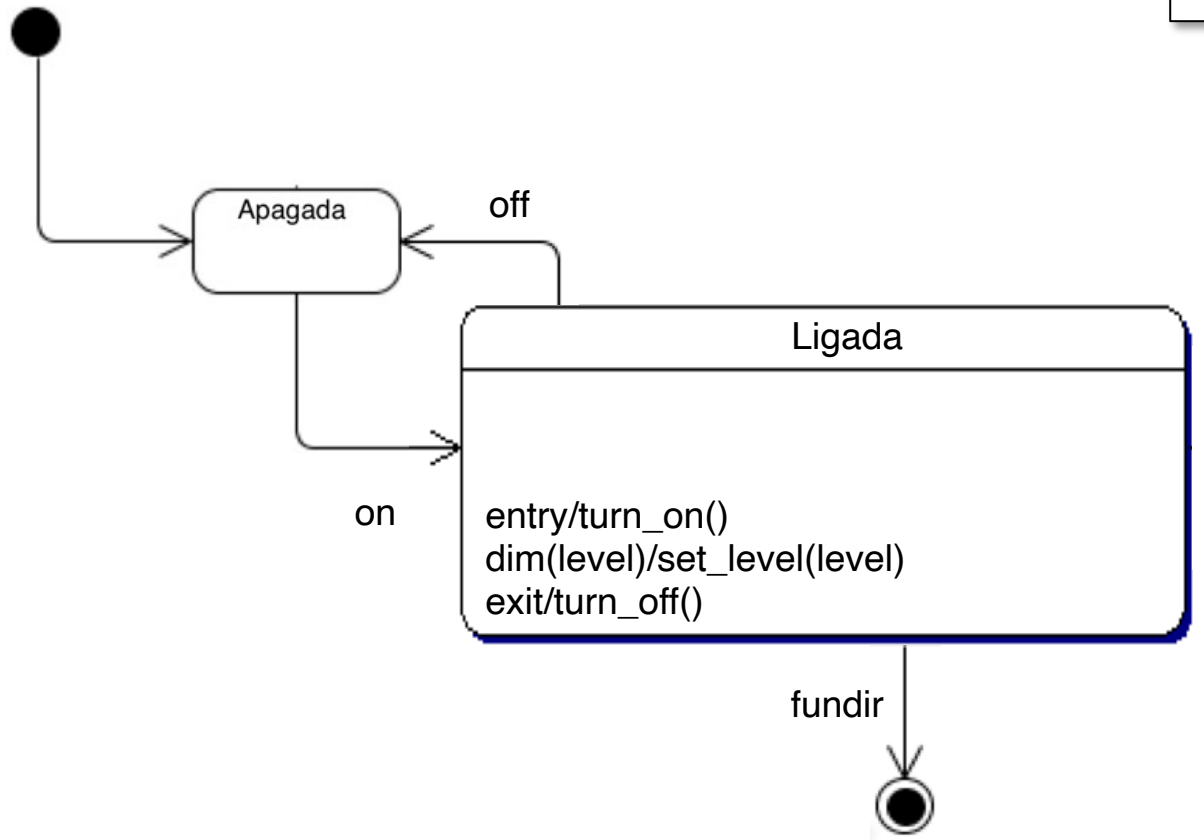


API da camada funcional:

- `turn_on()`
- `set_level(level)`
- `turn_off()`

Actividades internas

API da camada funcional:
- `turn_on()`
- `set_level(level)`
- `turn_off()`

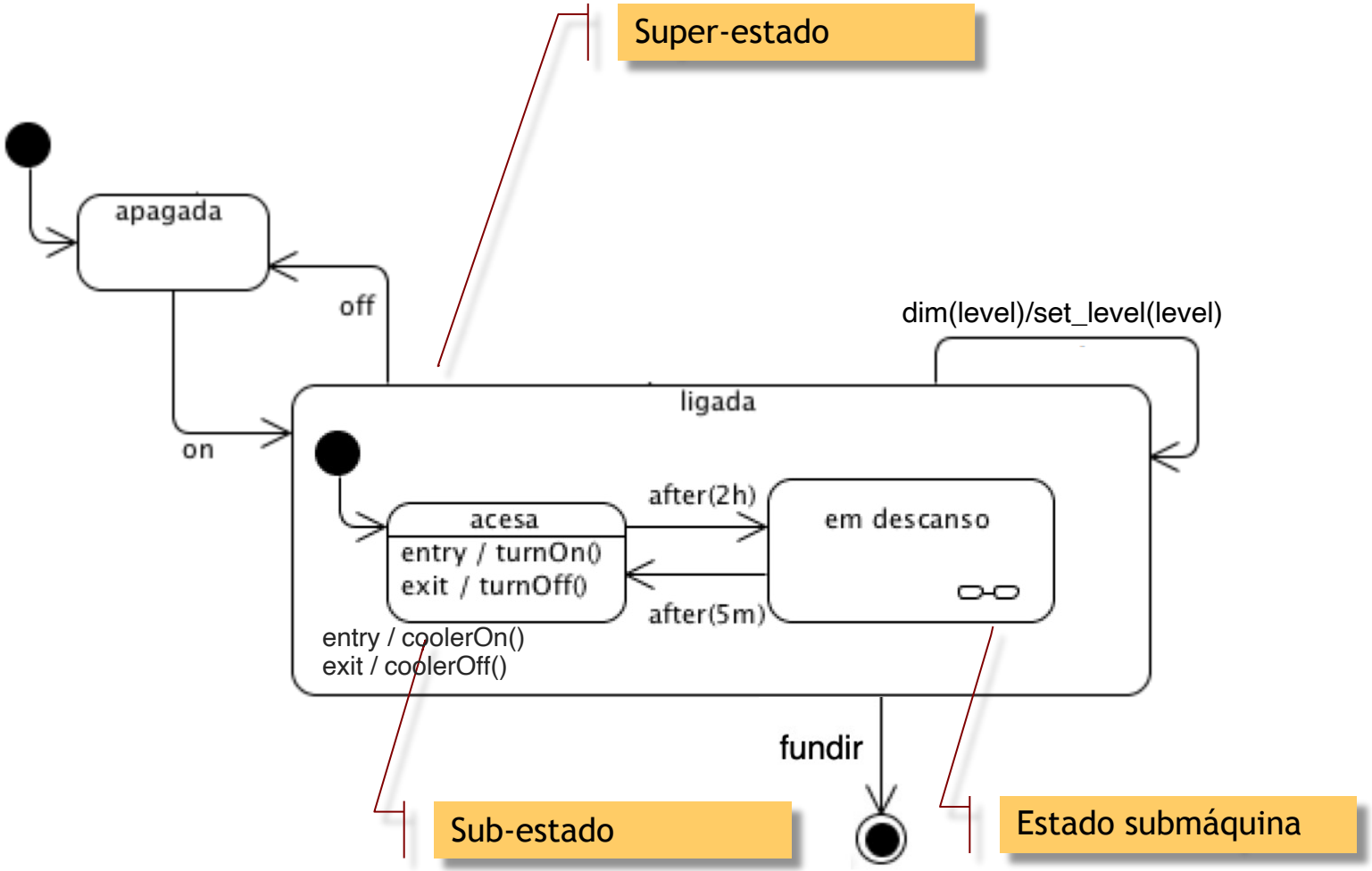


Actividades internas

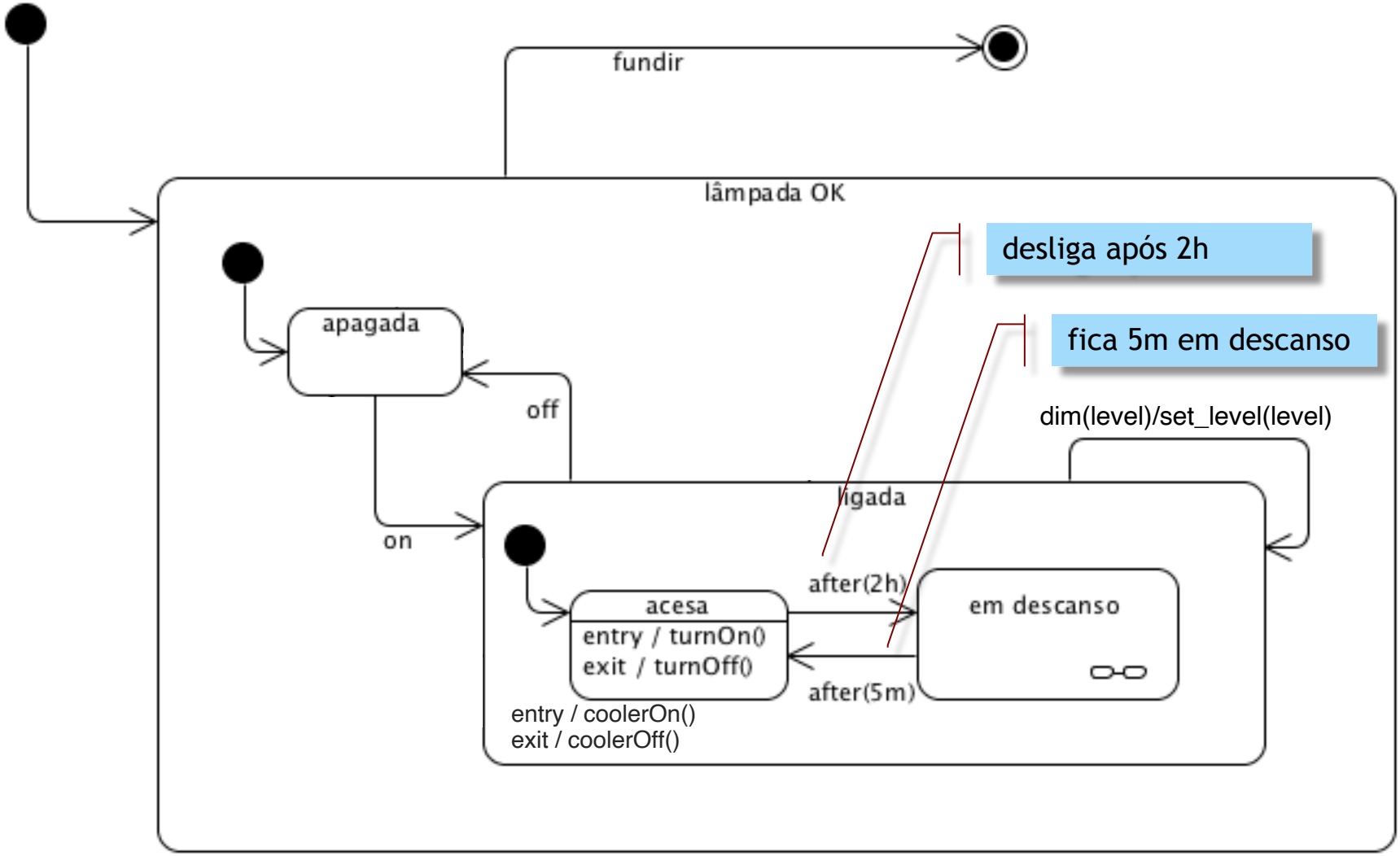
- Actividades que não provocam transições de estado…

entry/*acção*	• "*acção*" é automaticamente executada quando o objecto entra no estado;
do/*acção*	• "*acção*" é continuamente executada enquanto o objecto estiver no estado;
exit/*acção*	• "*acção*" é automaticamente executada quando o objecto sai do estado;
evento/*acção*	• "*acção*" é automaticamente executada se "*evento*" ocorrer (transição interna);
evento/defer	• "*evento*" é deferido até o estado actual ser abandonado.

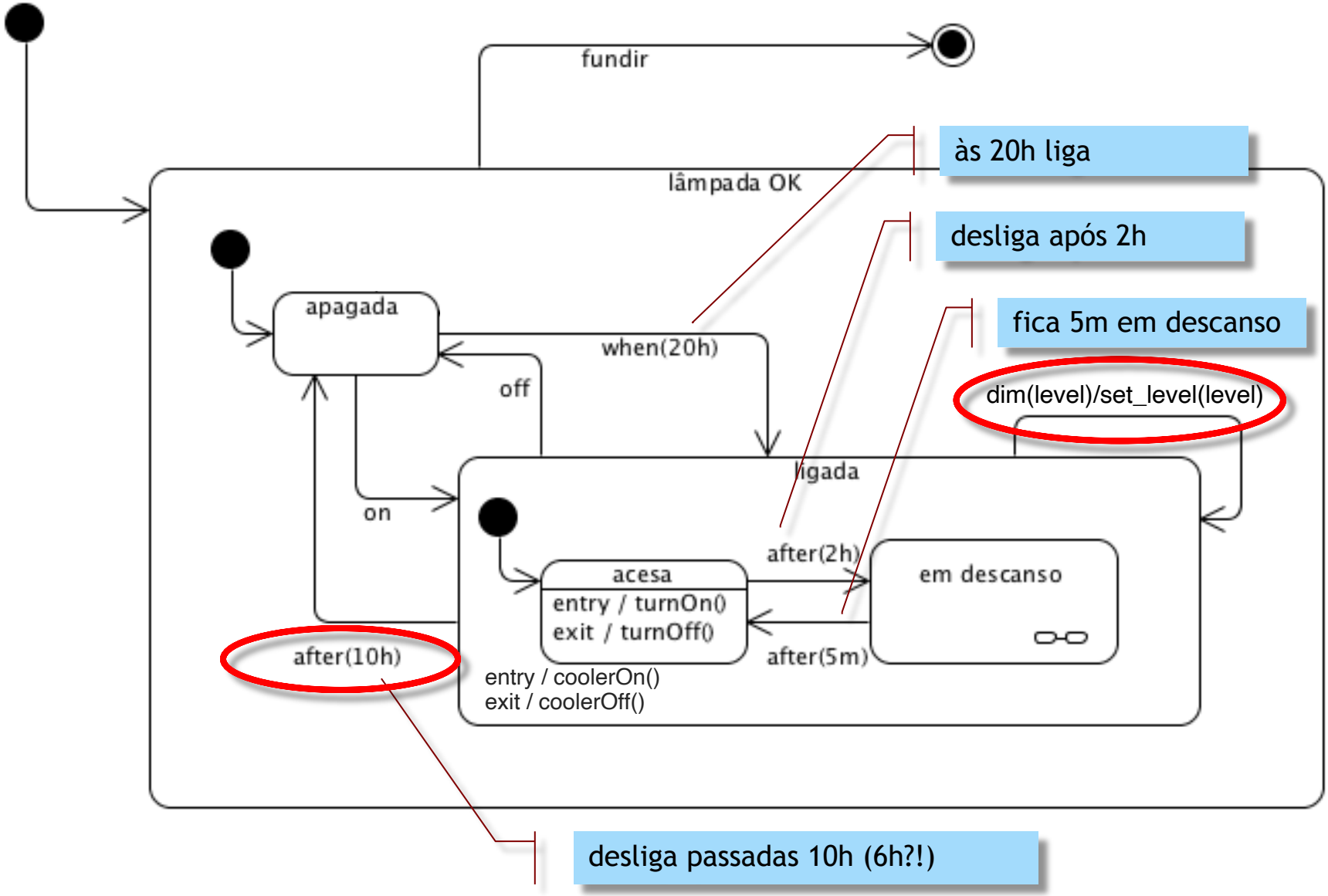
Estados e estados compostos (super-estados)



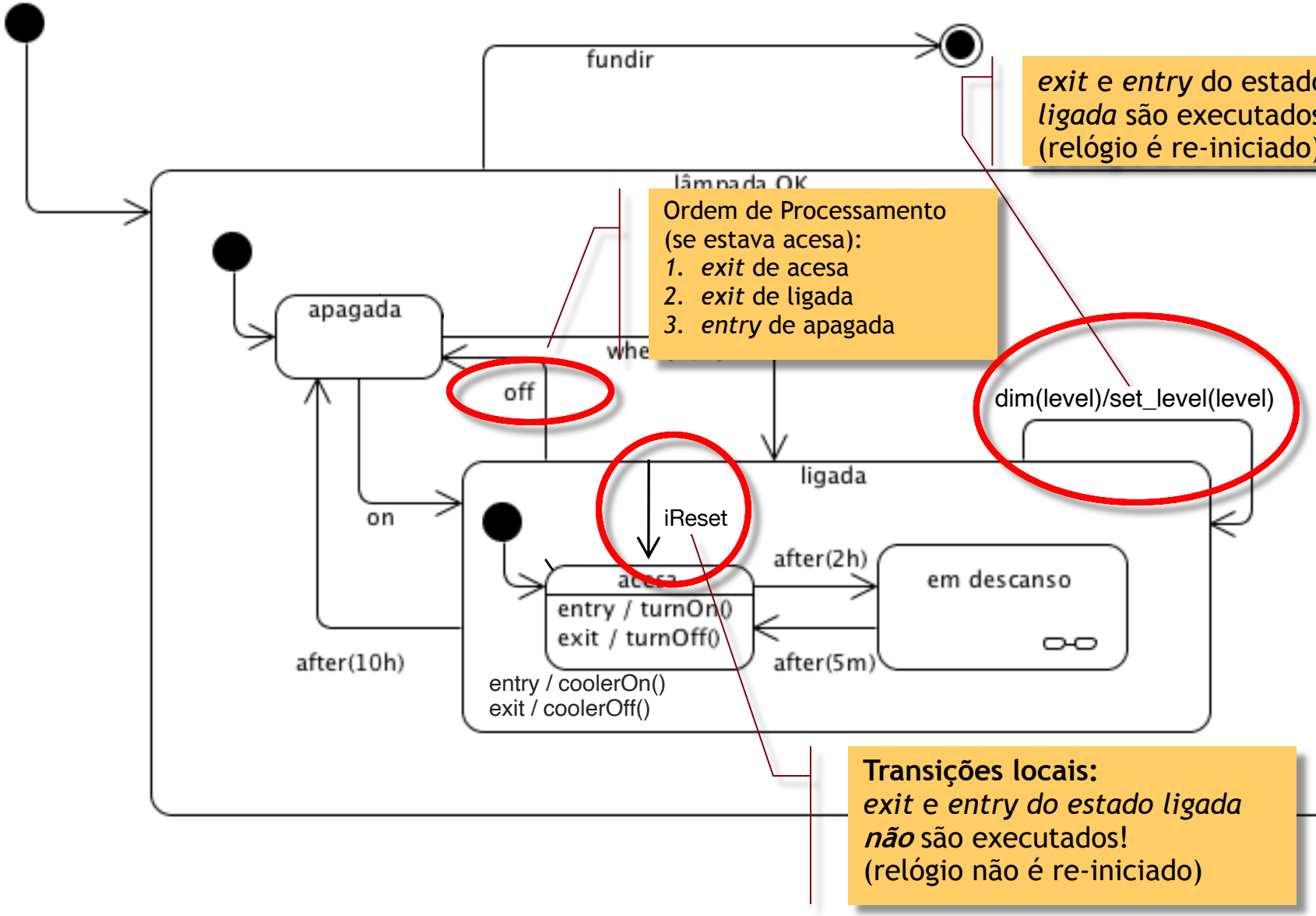
Eventos *when / after*



Eventos *when / after*



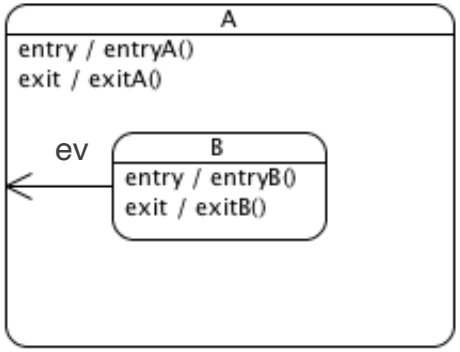
Transições vs. actividades internas



Transições locais vs. transições externas

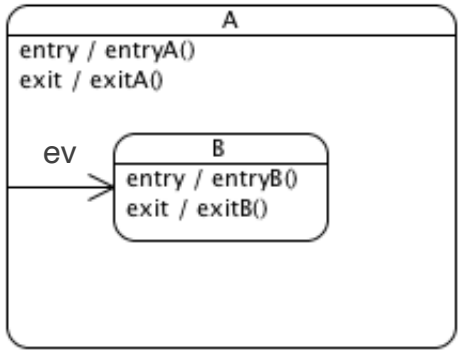
Em resposta ao evento **ev**, o modelo…

Transições locais



Executa:
- exitB()
- …

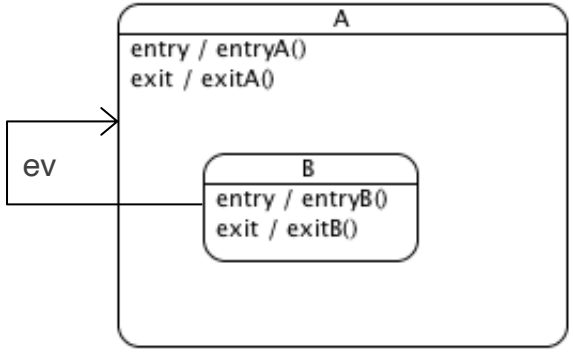
(sub-estado para super-estado)



Executa:
- …
- entryB()

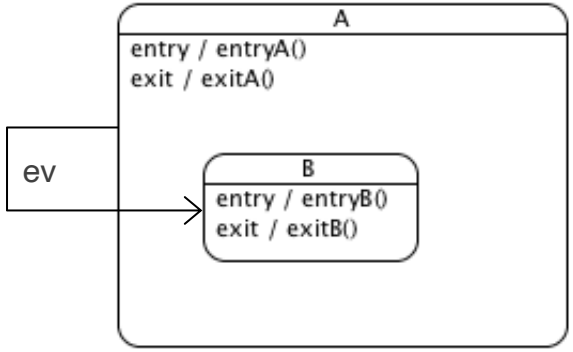
(super-estado para sub-estado)

Transições externas



Executa:
1. exitB()
2. exitA()
3. entryA()
4. …

(sub-estado para super-estado)

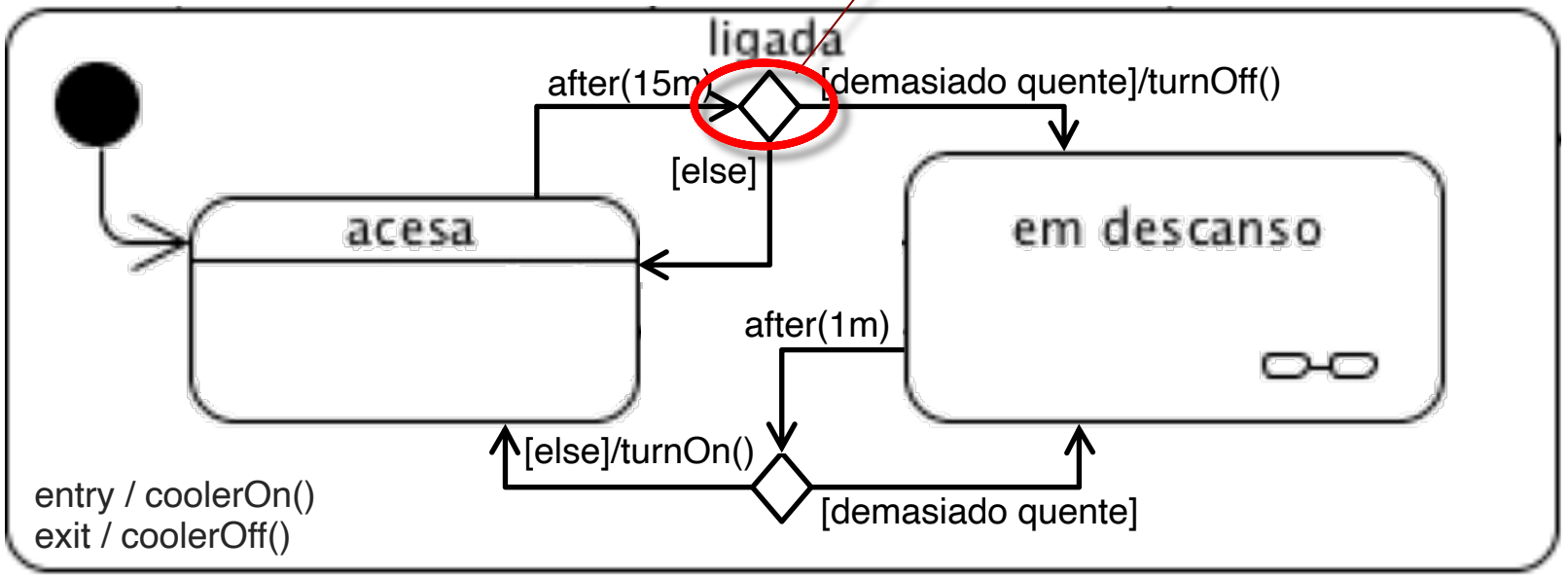


Executa:
1. …
2. exitA()
3. entryA()
4. entryB()

(super-estado para sub-estado)

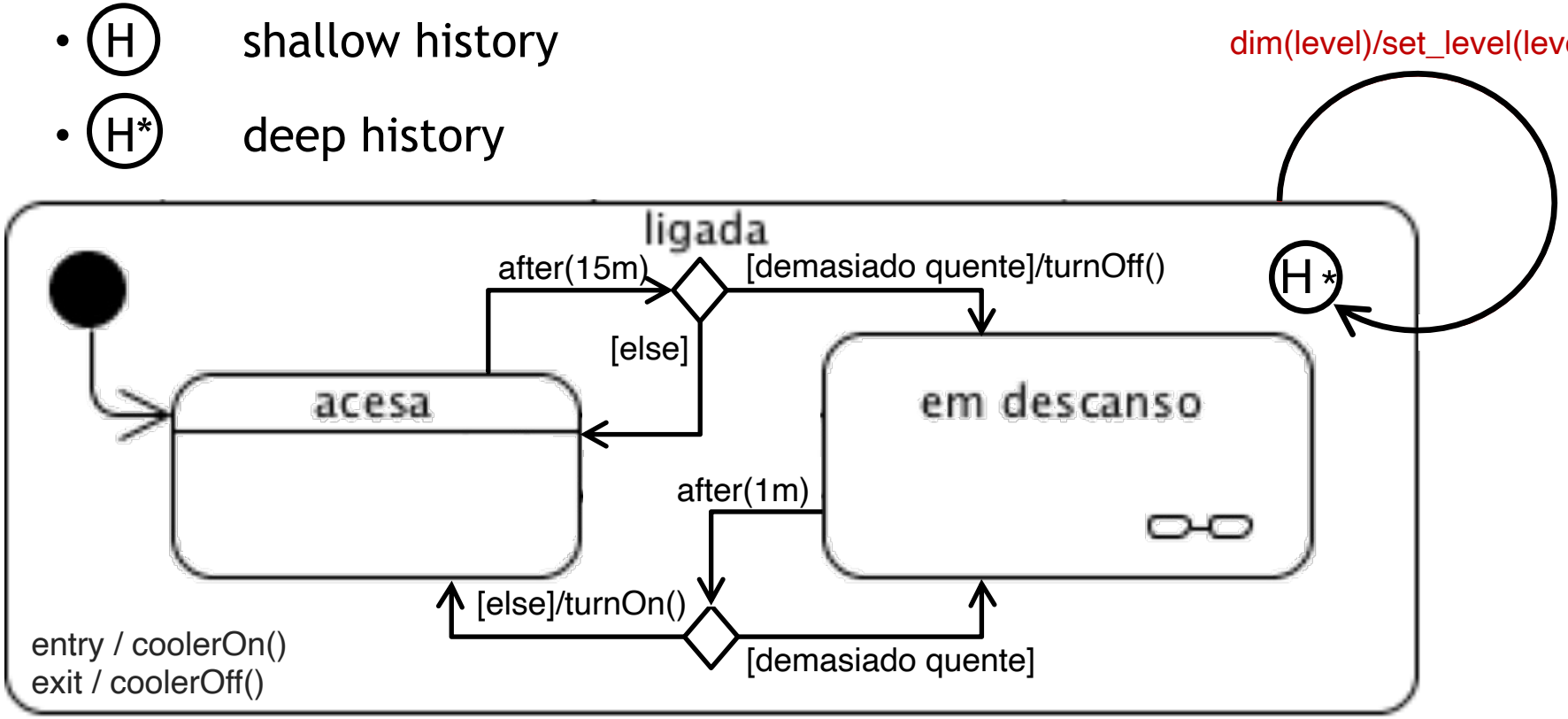
Pseudo-estado de Escolha

- Ramificação condicional (dinâmica!) em função do valor de uma expressão.
- Decisão pode ser uma função de acções anteriores.
- Caso mais que uma guarda verdadeira, a escolha é não deterministica.
- Se nenhuma guarda for verdadeira, o modelo está mal formado ([else]!)

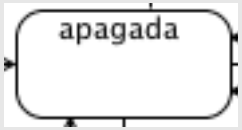
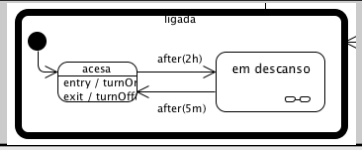
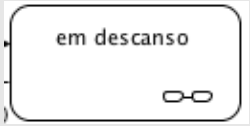
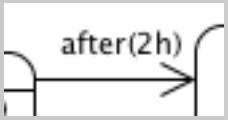
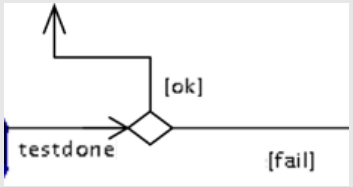


Pseudo-estados de História

- Permitem modelar interrupções – actividade da máquina é retomada no estado em que se encontrava aquando da última saída
  - (H) shallow history
  - (H*) deep history



Resumo da notação (até agora)

	Estado
	Estado composto
	Estado submáquina
	Pseudo-estado incial
	Estado final
	Transição (evento [condição] / acção) (entre estados vs. para o próprio estado vs. locais)
	Pseudo-estado de escolha
	Pseudo-estados de história (shallow/deep)