

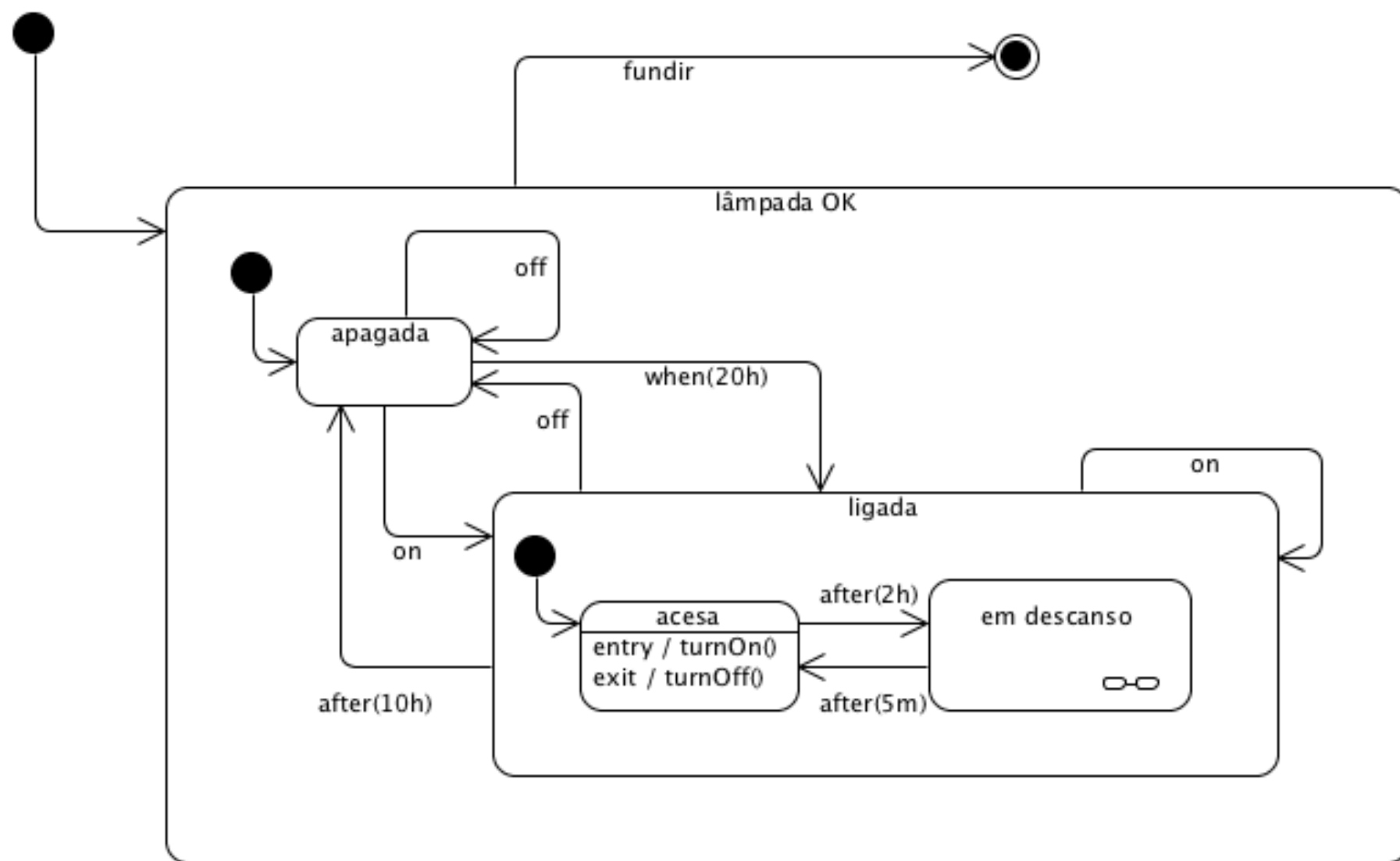


Desenvolvimento de Sistemas Software

Aula Teórica 10: Modelação de comportamento / Máquinas de Estado (II)

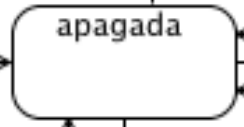
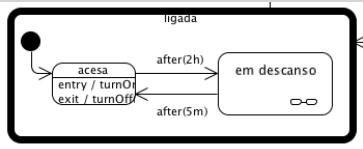
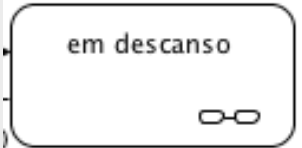
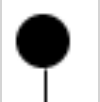

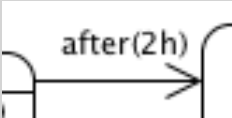



Resumo da aula anterior



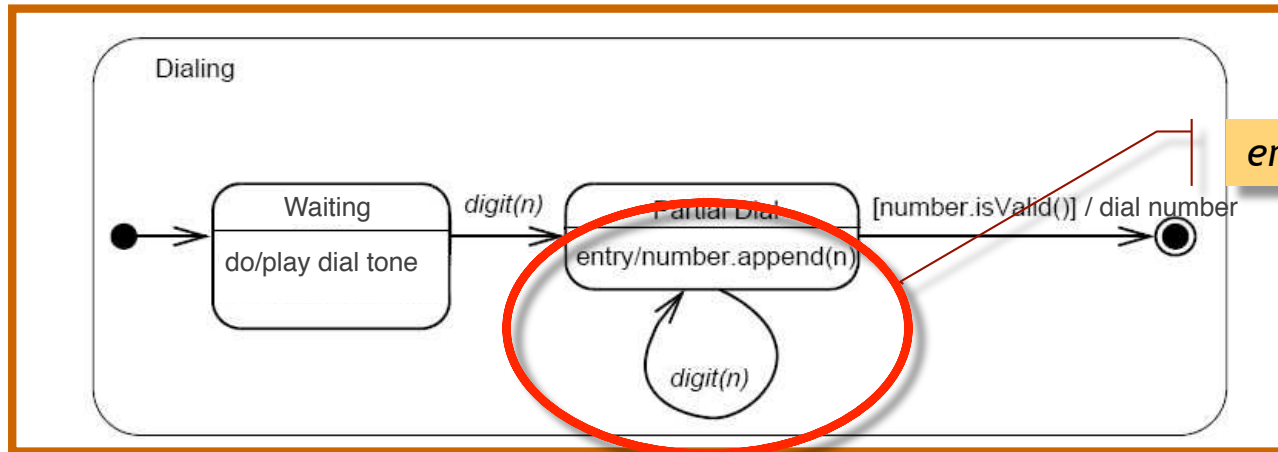


Resumo da notação (até agora)

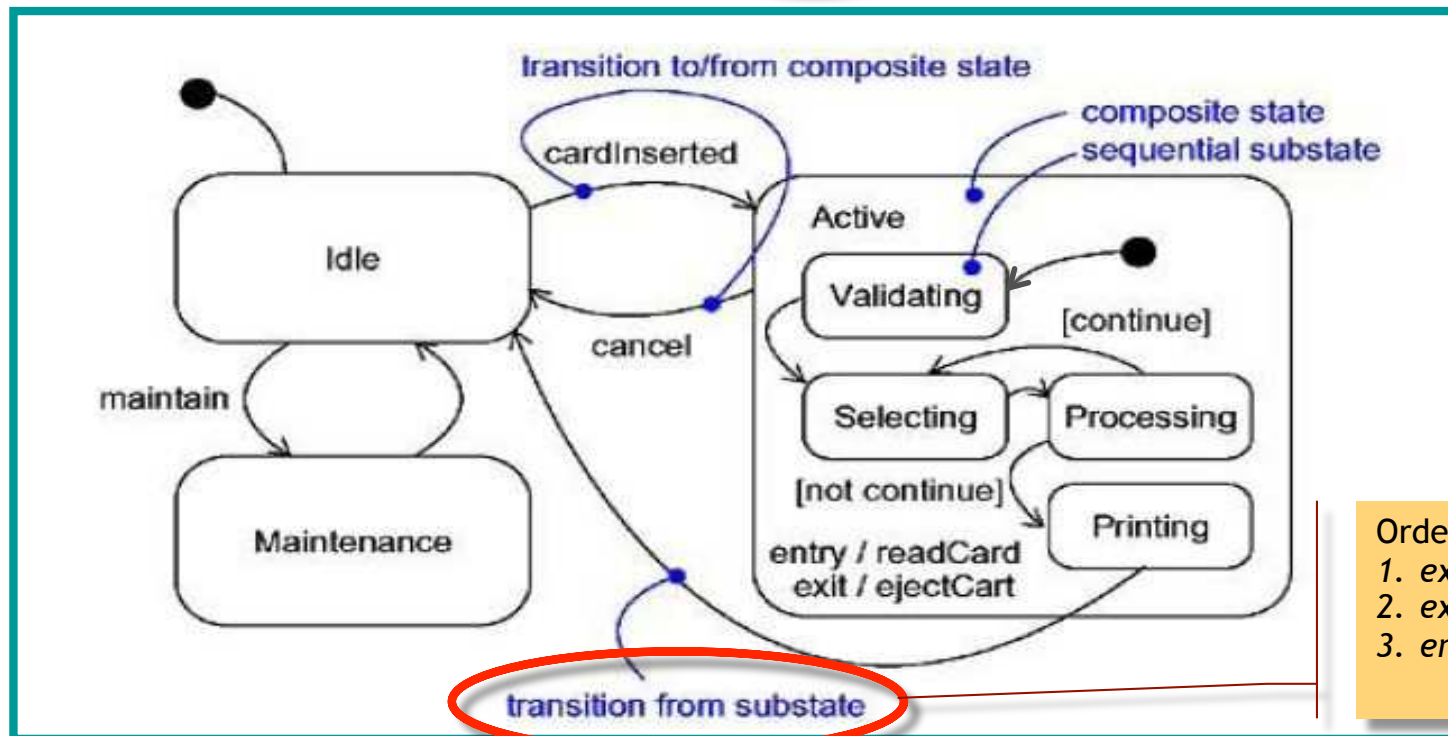
	Estado
	Estado composto
	Estado submáquina
	Pseudoestado inicial
	Estado final
	Transição (evento [condição] / acção)
	Transição para o próprio (evento [condição] / acção)



Transições vs. actividades internas



entry é executado!

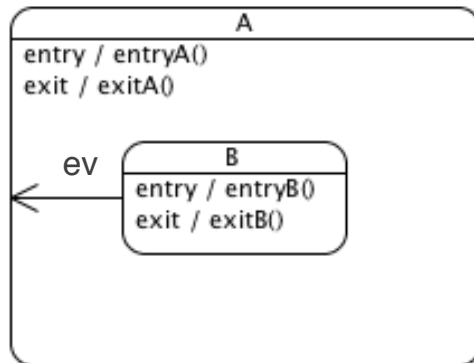


Ordem de Processamento:
1. exit de Printing
2. exit de Active
3. entry de Idle



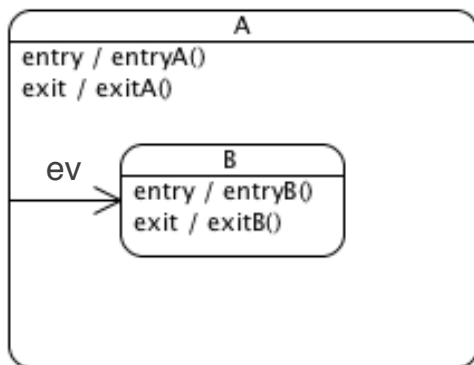
Transições locais vs. transições externas

Transições locais



Executa:
1. exitB()
2. ...

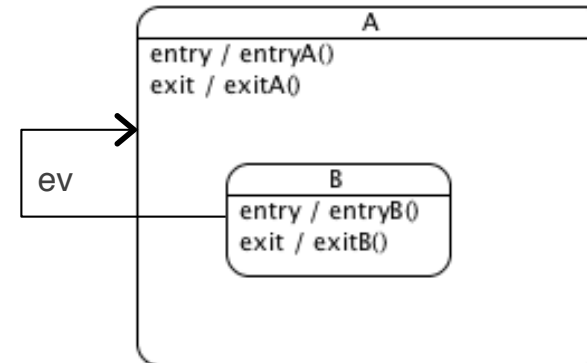
(sub-estado para super-estado)



Executa:
1. ...
2. entryB()

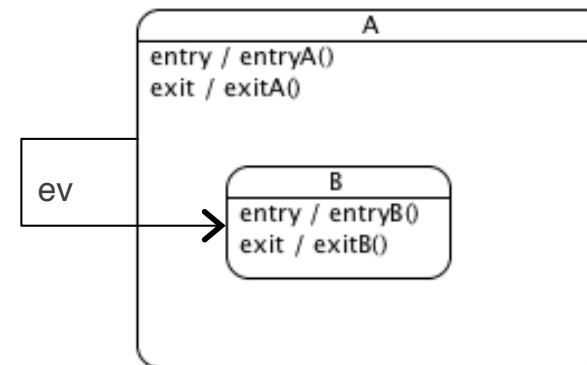
(super-estado para sub-estado)

Transições externas



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

(sub-estado para super-estado)



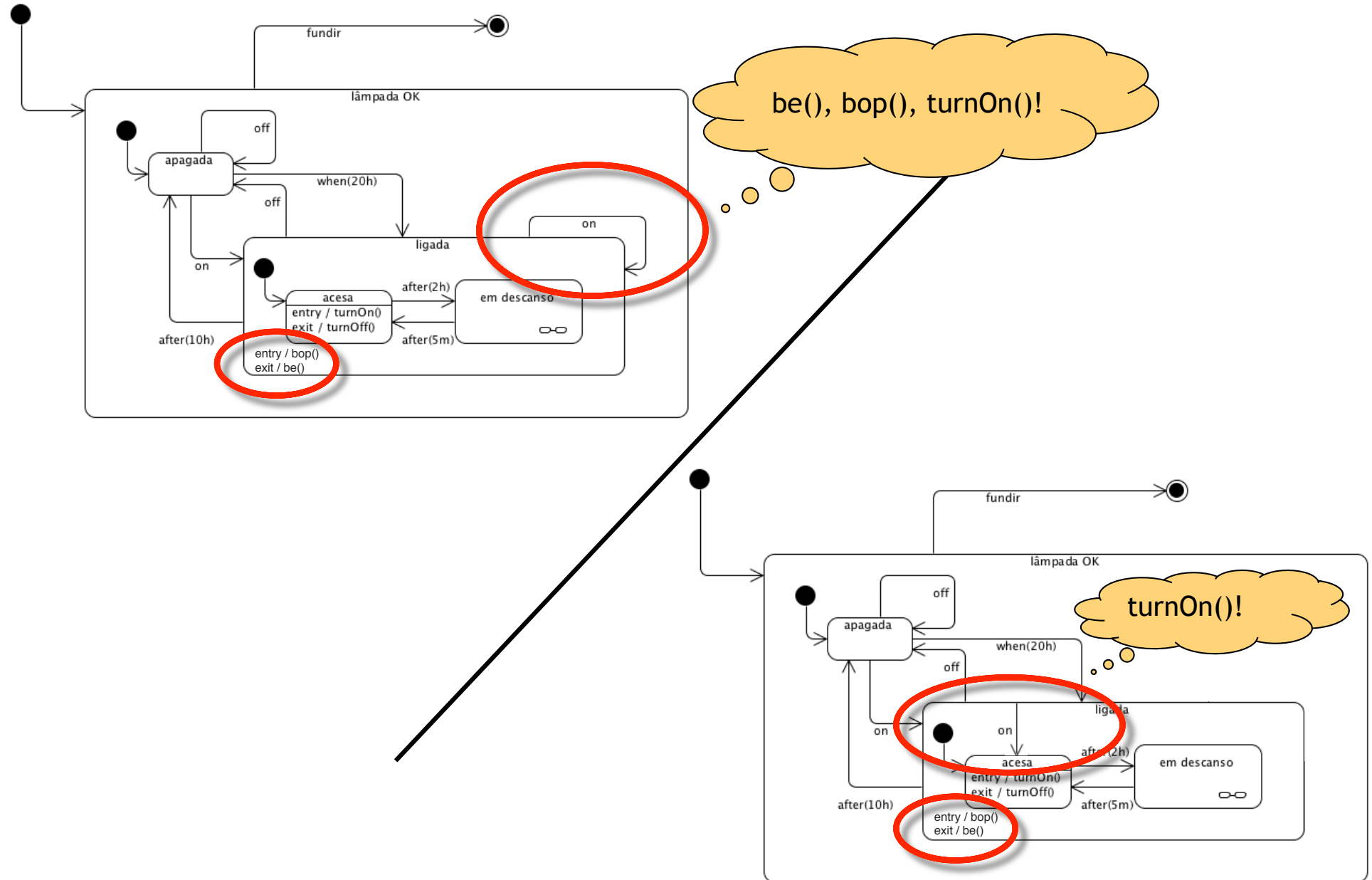
Executa:
1. ...
2. exitA()
3. entryA()
4. exitB()

(super-estado para sub-estado)



234

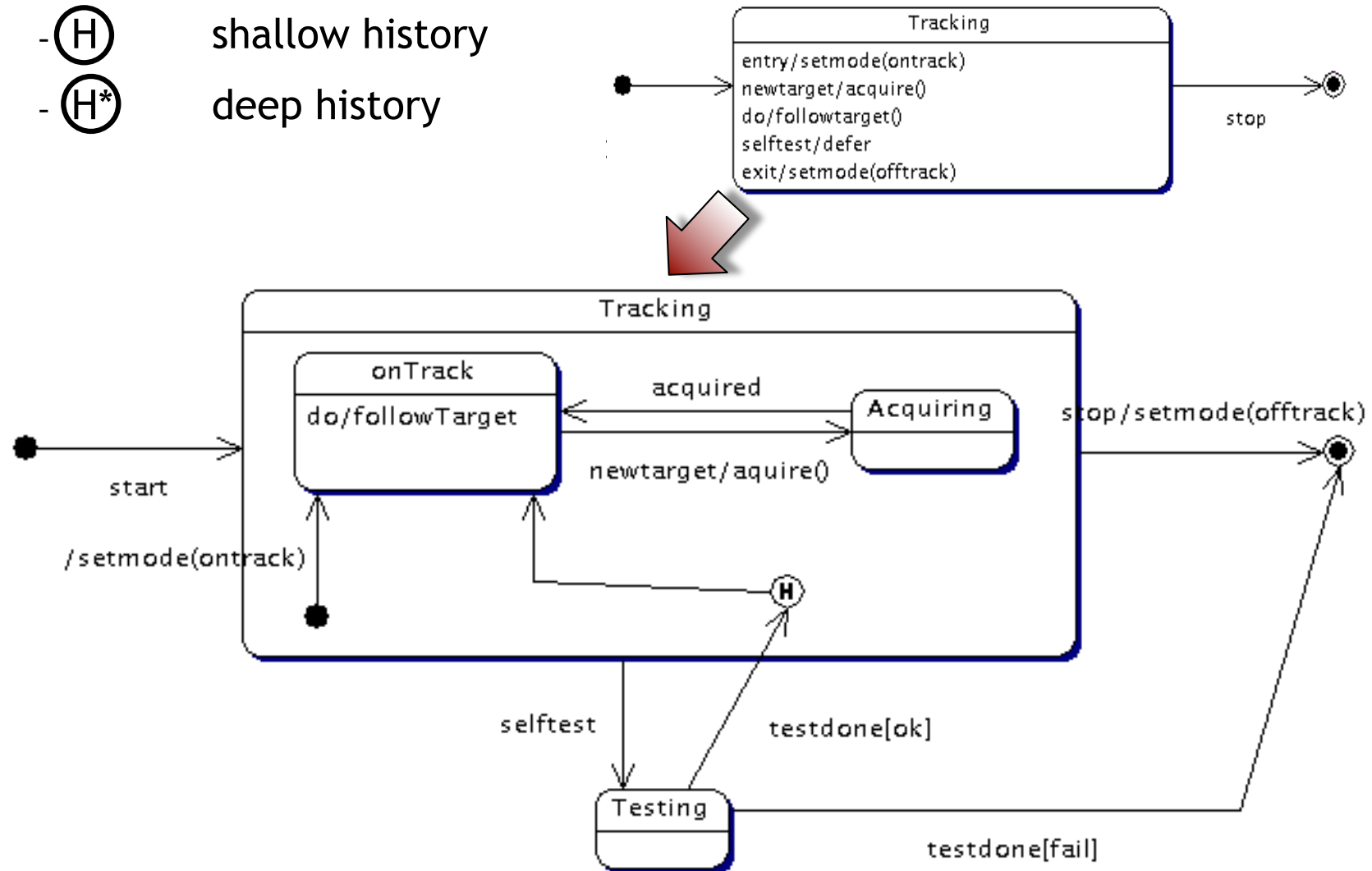
Exemplo



Pseudoestados de História

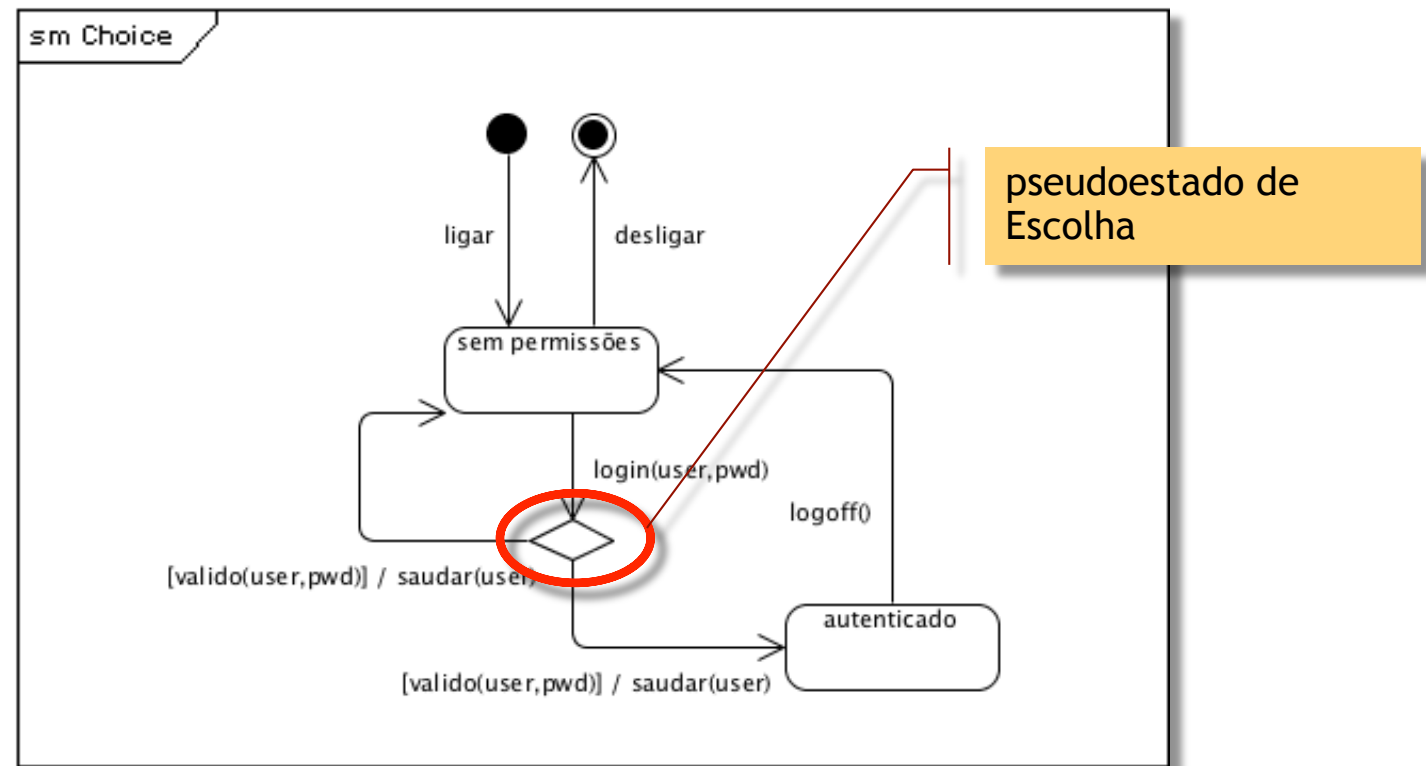
- Permitem modelar interrupções – actividade da máquina é retomada no estado em que se encontrava aquando da última saída

- \textcircled{H} shallow history
- $\textcircled{H^*}$ deep history



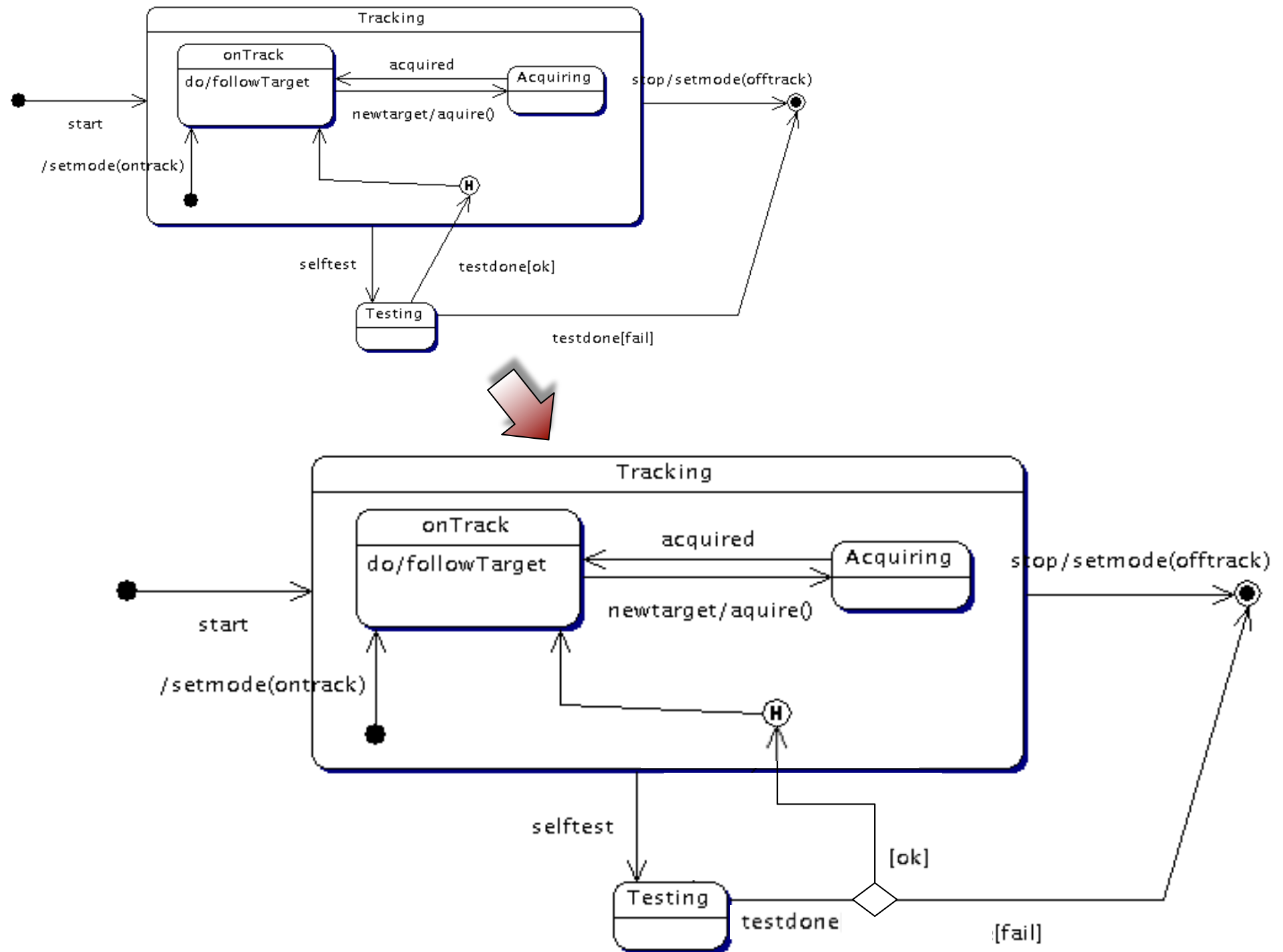
Pseudoestado de Escolha

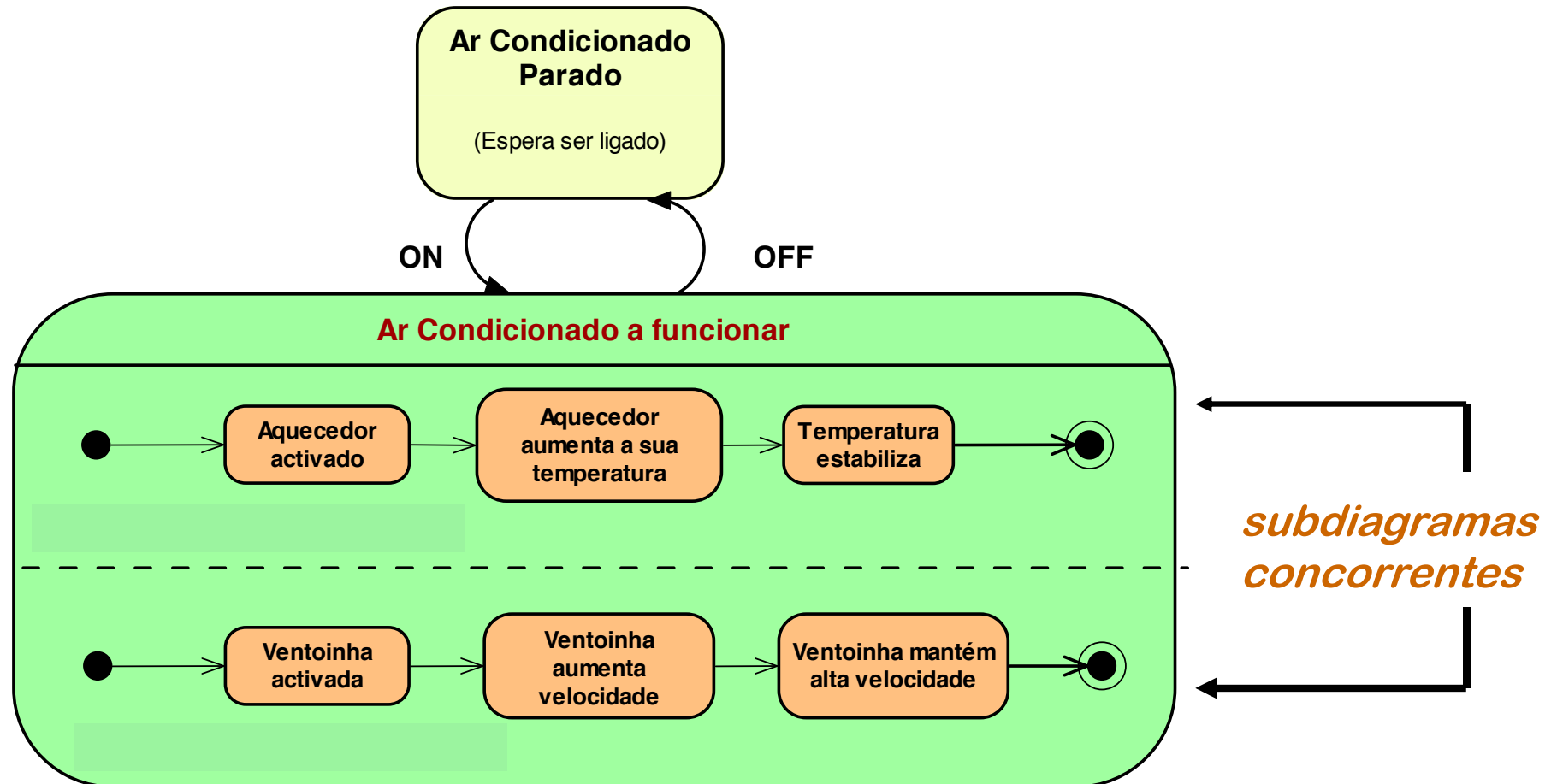
- Ramificação condicional (dinâmica!) em função do valor de uma expressão.
- Decisão pode ser uma função de acções anteriores.
- Caso mais que uma guarda verdadeira, a escolha é não determinística.
- Se nenhuma guarda for verdadeira, o modelo está mal formado ([else]!)





237

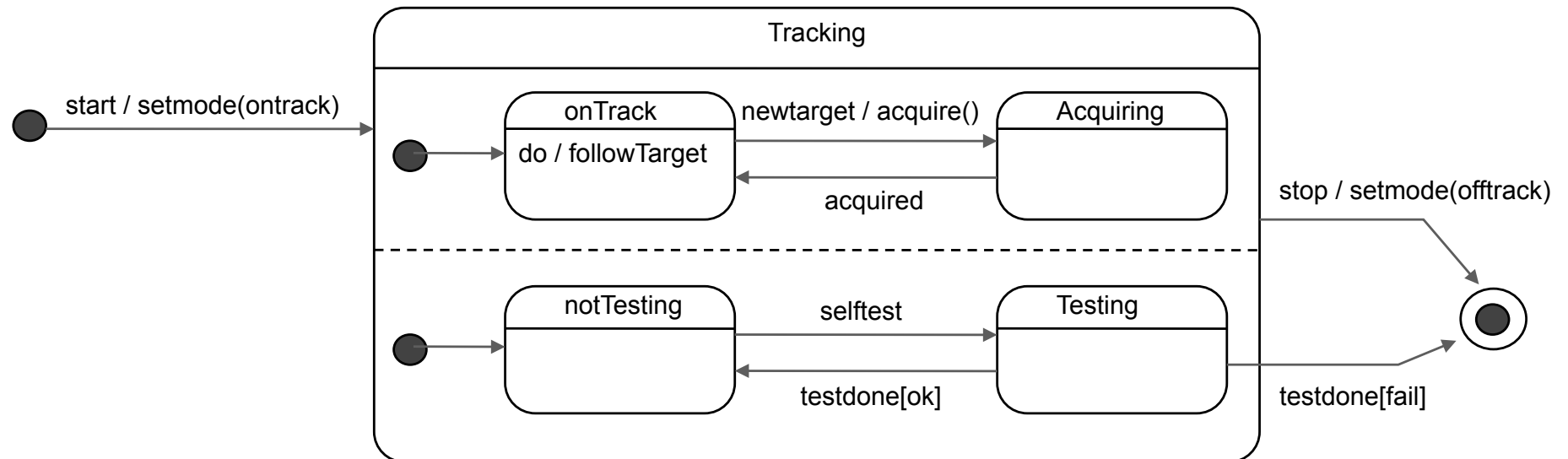




Quando se entra neste estado, os dois subdiagramas são executados de forma concorrente. O comportamento termina quando terminarem os 2.

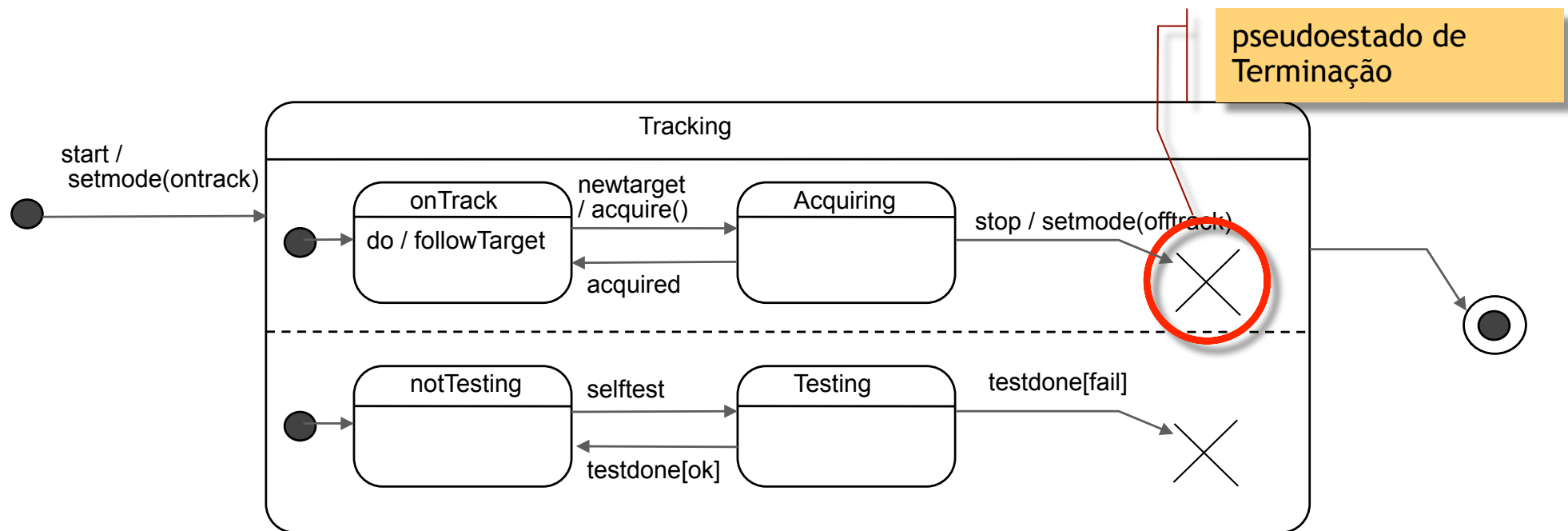
Estados com concorrência...

- Um estado pode ser dividido em “regiões” ortogonais
- Cada região contém um sub-diagrama
- Os diagramas das regiões são *executados* de forma concorrente



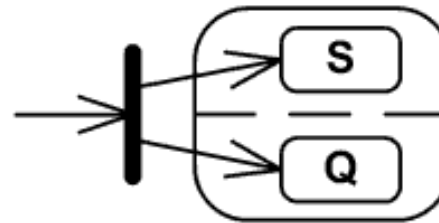
Pseudoestado de terminação

- Indica que a execução da máquina de estados termina.
- Não são executadas acções de saída a não ser as da transição para o estado de terminação

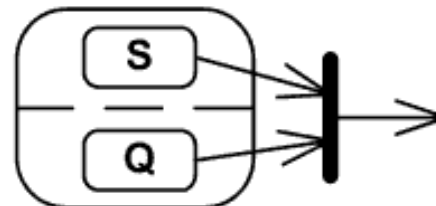


Pseudoestados *fork* e *join*

- Permitem gerir concorrência.
- Fork - divide uma transição de entrada em duas ou mais transições
 - Transições de saída têm que terminar em regiões ortogonais distintas

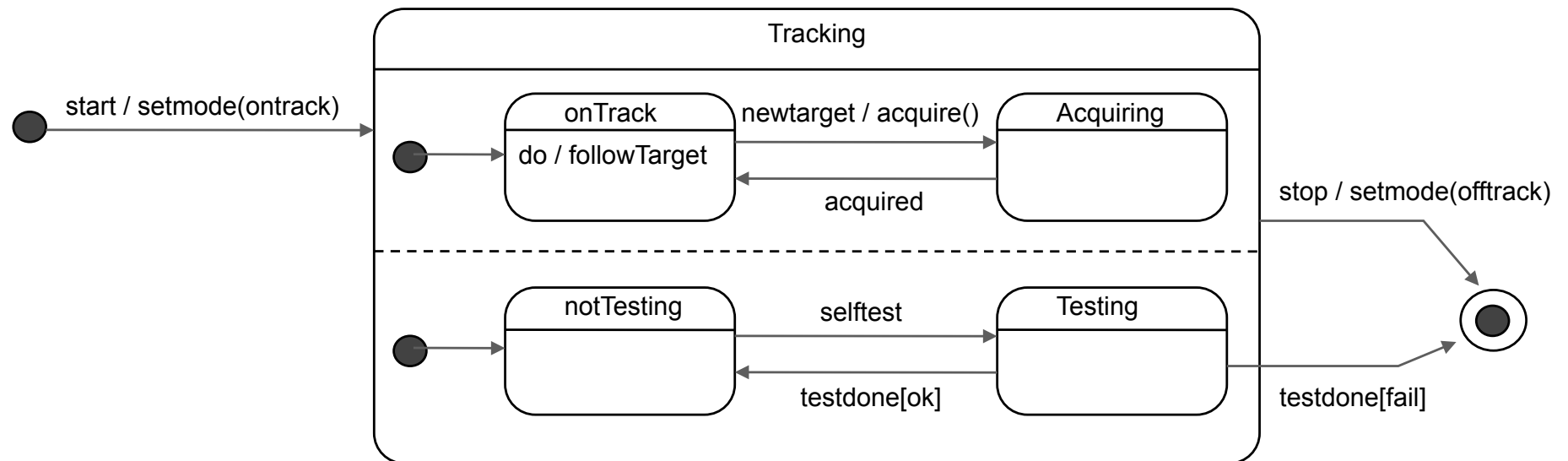


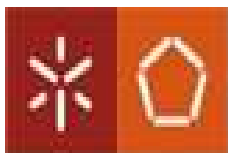
- Join - funde duas ou mais transições de entradas numa só transição e saída
 - Transições de entrada têm que originar em regiões ortogonais distintas



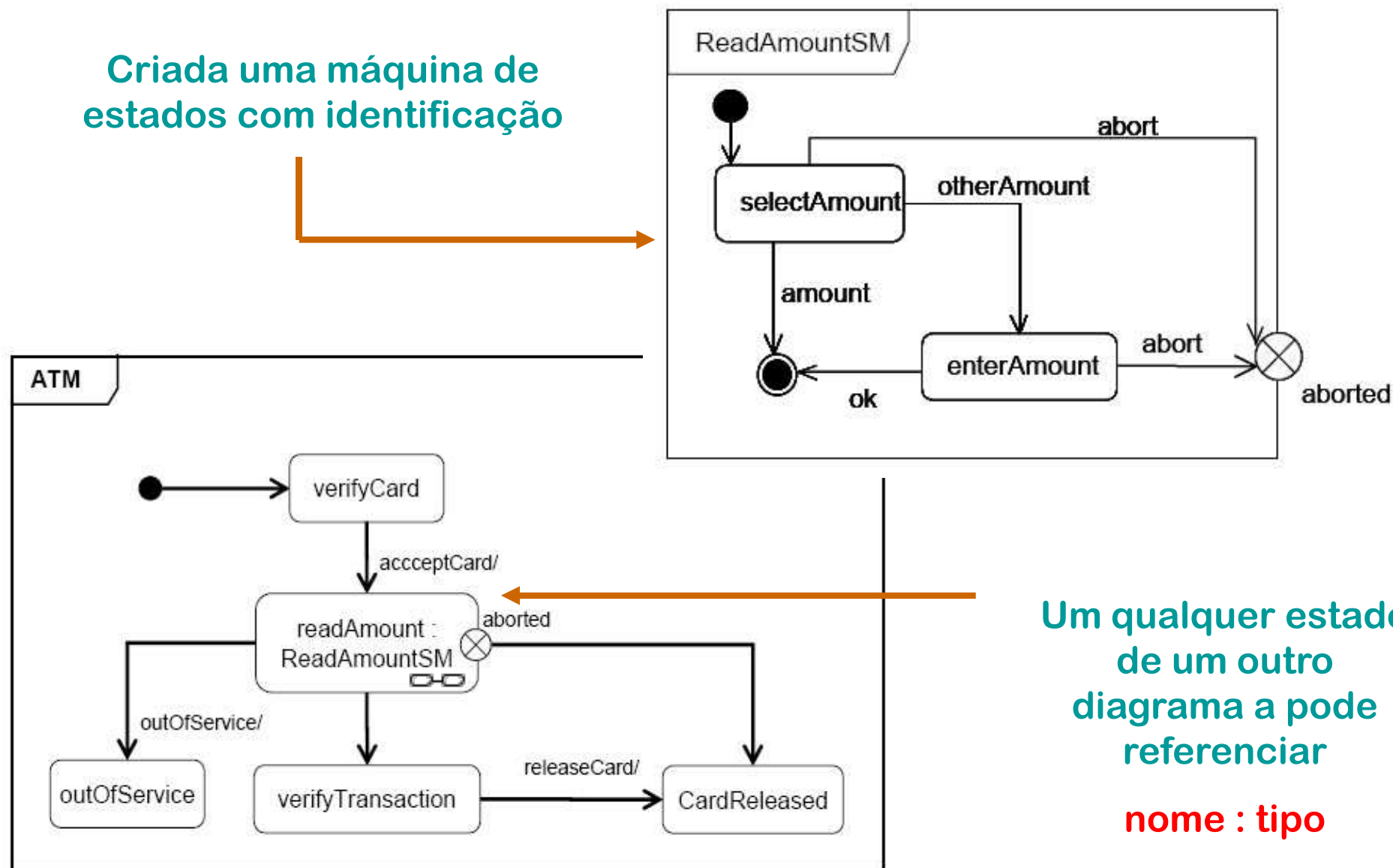
Pontos de entrada e saída

- Como fazer para "esconder os detalhes" do estado Tracking?
- Transição a partir do sub-estado Testing levanta problemas...






Criada uma máquina de estados com identificação




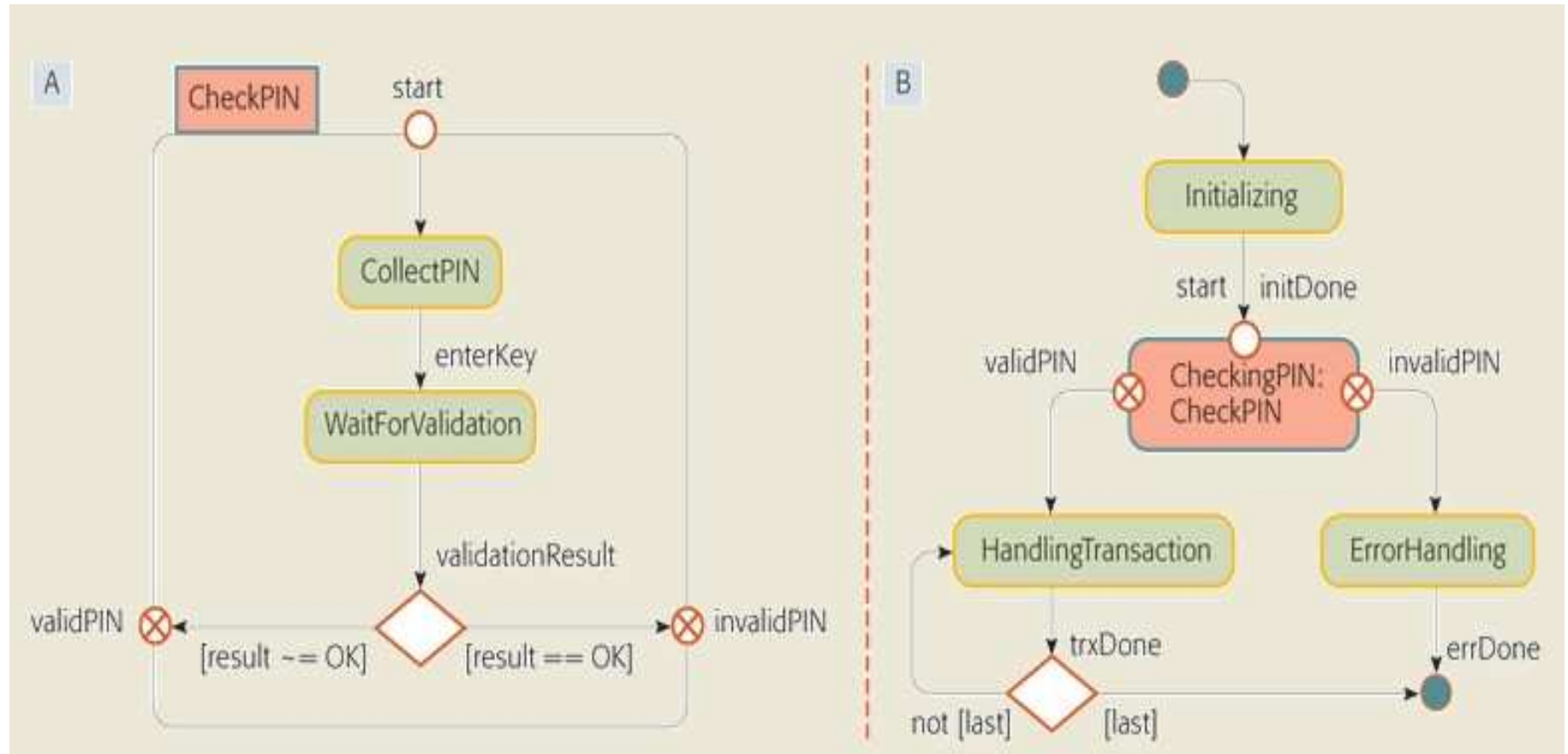
Um qualquer estado de um outro diagrama a pode referenciar

nome : tipo

Pseudoestados Ponto de entrada e Ponto de saída

- Ponto de entrada 
 - Permite definir um ponto de entrada numa máquina de estados ou num estado composto
 - O ponto de entrada é identificado por nome
 - O ponto de entrada transita para um estado interno que poderá ser diferente do definido pelo estado inicial

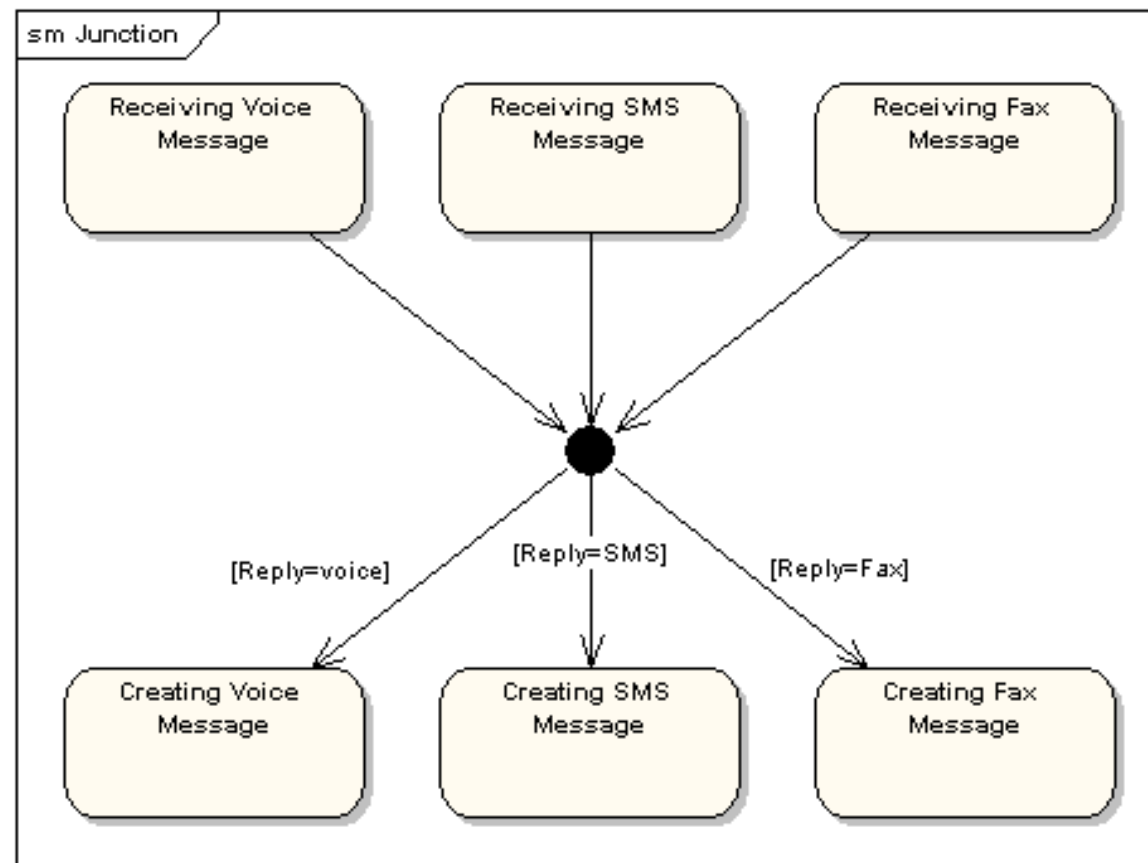
- Ponto de saída 
 - Permite definir um ponto de saída alternativo ao estado final
 - O ponto de saída é identificado por nome

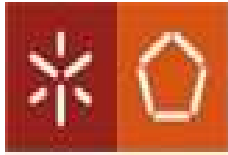


Submáquina CheckPIN

Pseudo-estado de Junção

- Ramificação condicional (estática!) em função do valor de uma expressão.
- Caso mais que uma guarda verdadeira, a escolha é não determinística.
- Se nenhuma guarda for verdadeira, o modelo está mal formado ([else]!)





- ▣ **Diagramas de Estado** permitem-nos descrever o comportamento de uma entidade importante do sistema de forma completa, ou seja, trazendo para um único diagrama o comportamento que em geral está especificado de forma dispersa em vários UC ou DS.
- ▣ Os **diagramas de actividade** também permitem uma visão mais sistémica, pois permitem especificar fluxos importantes de actividades que envolvem vários objectos, use cases e até actores.
- ▣ **Diagramas de Estado** não são adequados para descrever ou analisar colaborações entre entidades/objectos.
- ▣ **Diagramas de Estado** não são usados para descrever todas as classes do sistema, mas aquelas que exibam comportamento interessante ou complexo. Alguns autores usam DMEs para especificar a Interface com Utilizador.



Diagramas de Estado (*Statecharts*)

Sumário

- Mais sobre transições
- Transições vs. actividades internas
- Regiões concorrentes
- Mais pseudoestados: História, Escolha, *Fork*, *Join*, Terminação, Pontos de Entrada e Saída, Junção