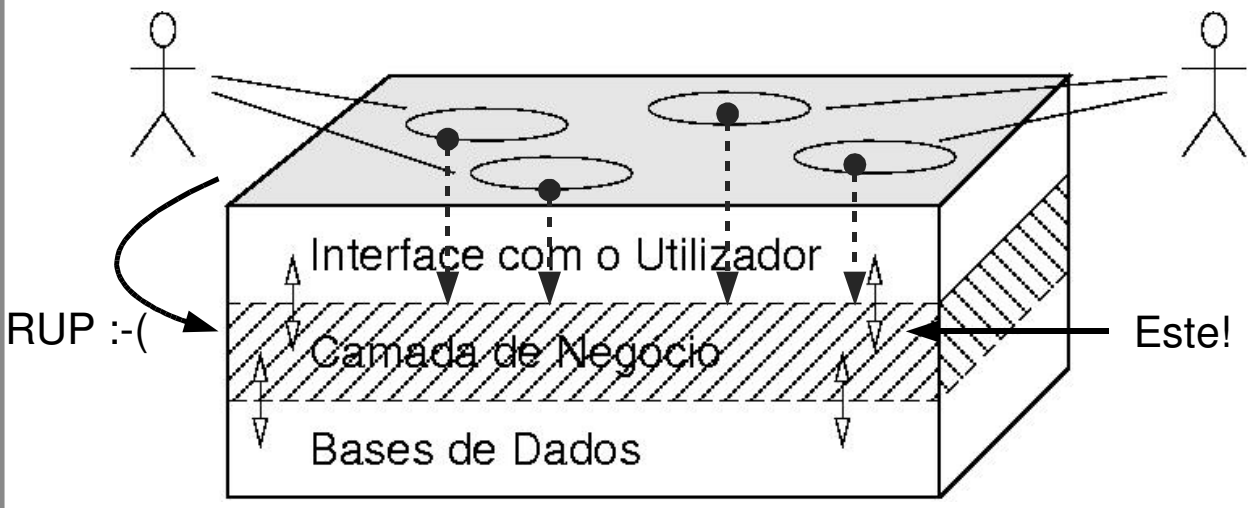
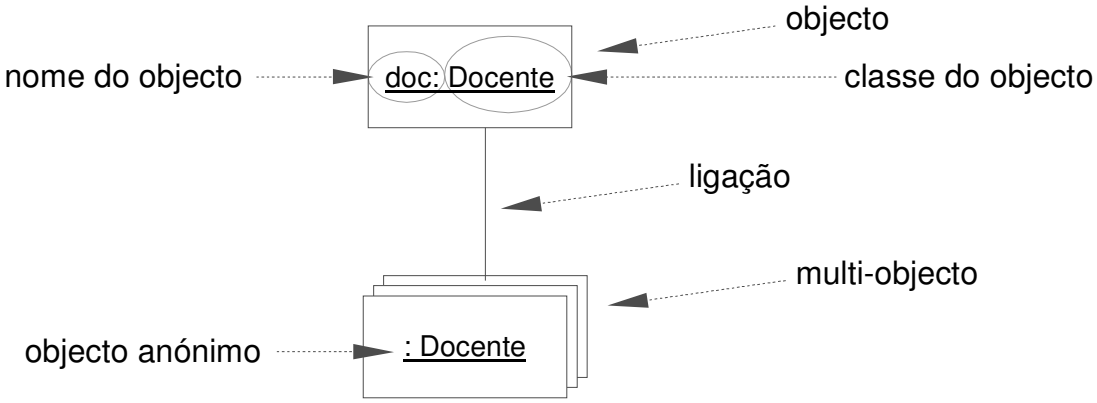
Que nível estamos a modelar?



Atenção: Actor Docente ≠ Classe Docente!

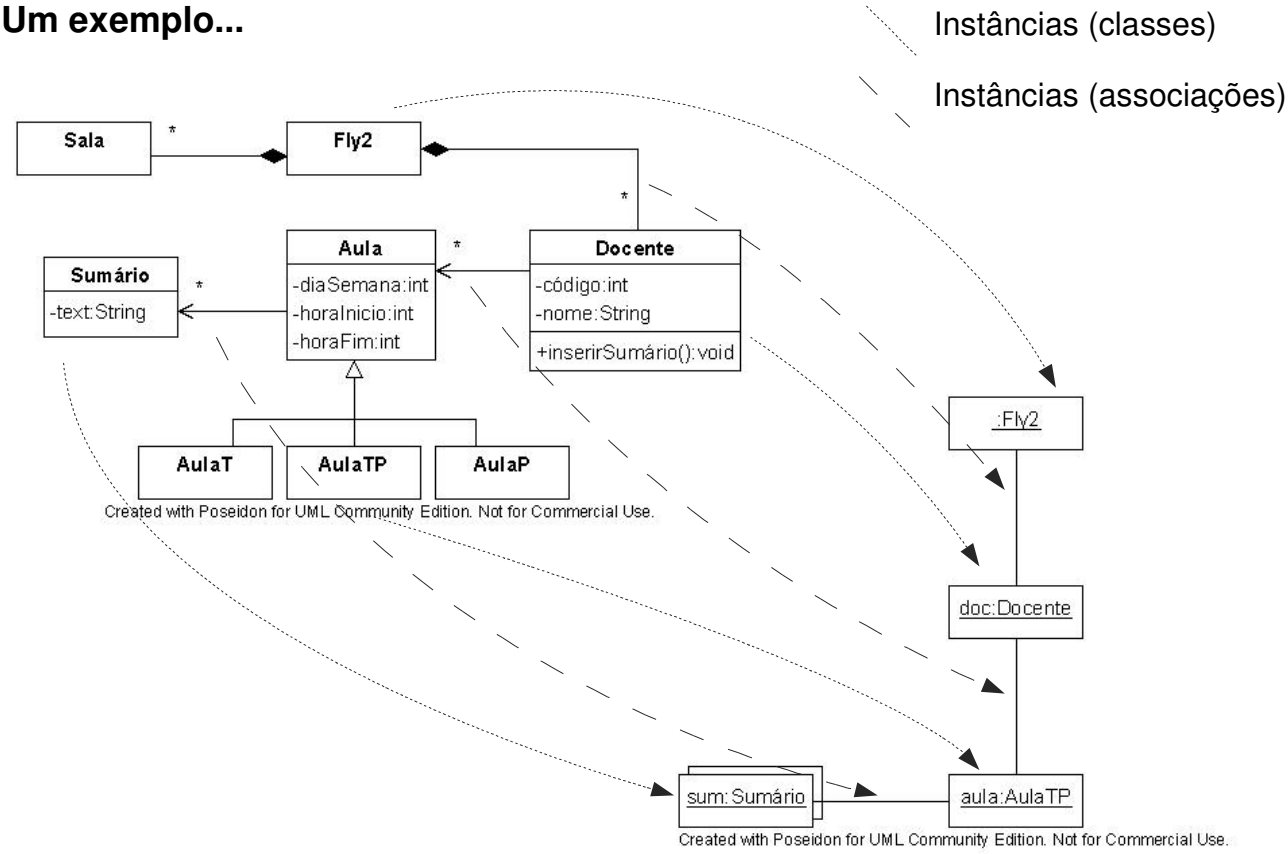
Diagramas de Objectos

- Apresentam uma configuração particular de objectos no sistema.
- Modelam a visão estática do sistema, do ponto de vista de instâncias concretas.
- Permitem representar objectos e ligações entre os objectos:
  - objectos — são instâncias das classes do modelo (isto é, do diagrama de classes);
  - ligações — são instâncias das associações entre as classes.



- Úteis, por exemplo, para *debugging*.

Um exemplo...



Diagramas de Interacção

Sumário:

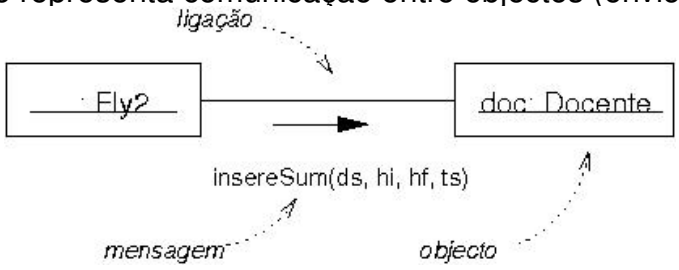
- Tipos de Diagramas de Interacção
- Interacções
- Diagramas de Colaboração – conceitos base
- Diagramas de Sequência – conceitos base
- Diagramas de Colaboração – conceitos avançados
- Diagramas de Sequência – conceitos avançados

Tipos de Diagrama de Interacção

- Os Diagramas de Classe modelam a arquitectura de objectos do sistema
- Os Diagramas de Interacção modelam o diálogo entre os objectos que compõem o sistema.
- O UML define dois tipos de diagramas de interacção:
  - Diagramas de Colaboração
  - Diagramas de Sequência
- Os dois tipos de diagrama têm o mesmo poder expressivo.
- Muda a forma de apresentação:
  - Diagramas de Colaboração – ênfase na estrutura;
  - Diagramas de Sequência – ênfase na sequência temporal dos eventos.
- Componentes base dos diagramas de interacção:
  - Objectos (ver diagramas de objecto);
  - Interacções entre objectos.

Interacções

- Uma Interacção representa comunicação entre objectos (envio de mensagens).

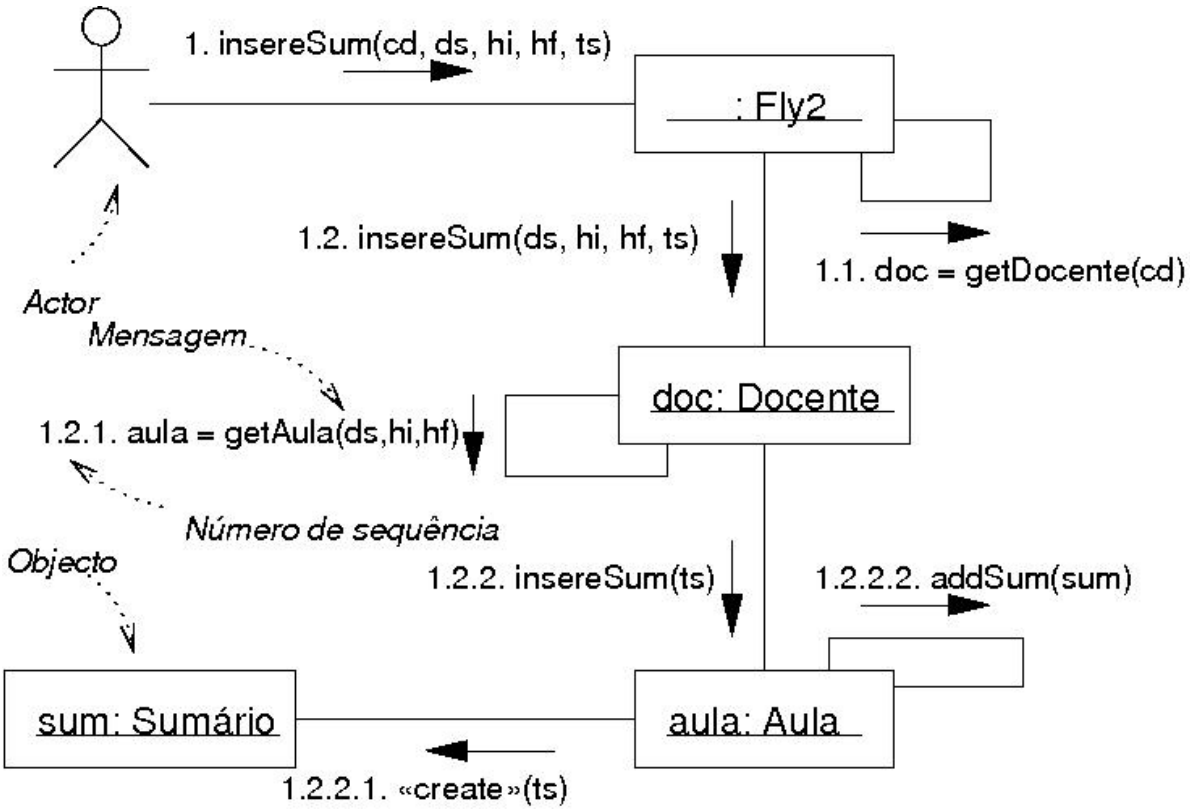


- Ligação:
  - Instância de uma associação
  - Decorações:
    - Association – a instância é conhecida por intermédio de uma associação
    - Self – a instância refere-se a si própria
    - global/local/parameter – *scope* da instância num dado contexto
  - Mensagens:
    - call – invocação síncrona (⟶)
    - send – invocação assíncrona (⟶)
    - Return – (<- - - -)
    - create – criar objecto
    - destroy – destruir objecto

Diagramas de Colaboração

- Apresentam uma configuração específica do sistema que suporta um dado *Use Case* (que objectos deverão existir em tempo de execução – *runtime*).
- Para essa configuração, mostra como os objectos comunicam entre si (colaboram) para satisfazer um dado pedido.
- A ênfase está na organização estrutural dos objectos:
  - Permite analisar se a configuração necessária é suportada pela estrutura de classes idealizada nos diagramas de classe.
- Ordenação temporal das mensagens não é claramente representada.

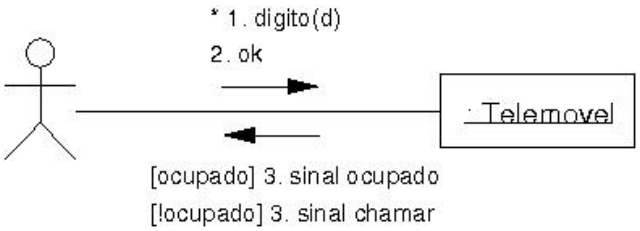
Um exemplo...



Diagramas de Colaboração – Lógica Condicional

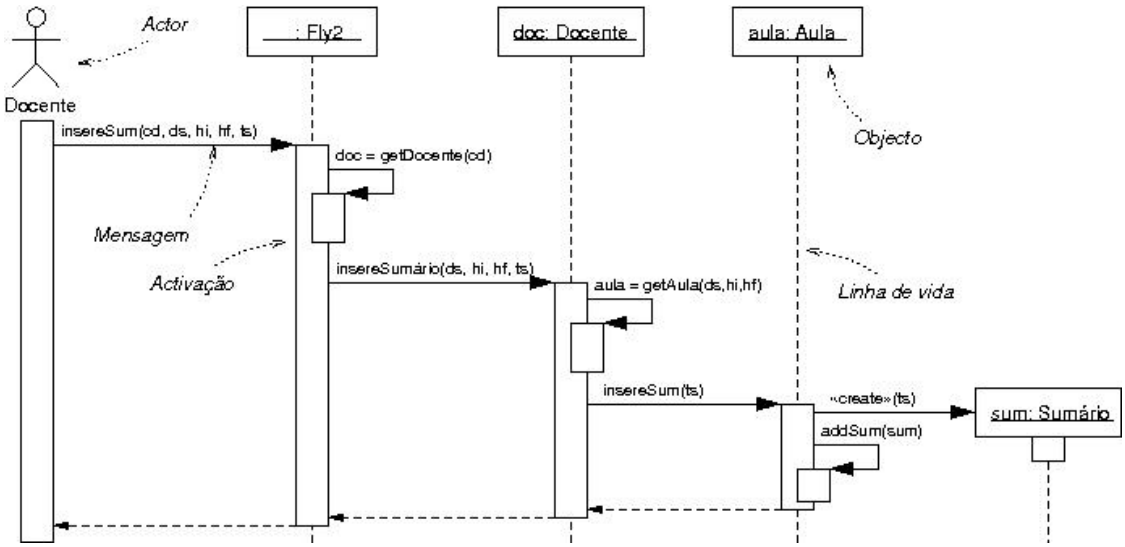
- A notação suporta a modelação de estruturas condicionais e repetitivas:
  - Condição: [*guarda*] *mensagem*
  - Repetição: *[*guarda*] mensagem

- Em alternativa podemos utilizar mais que um diagrama para modelar o comportamento.



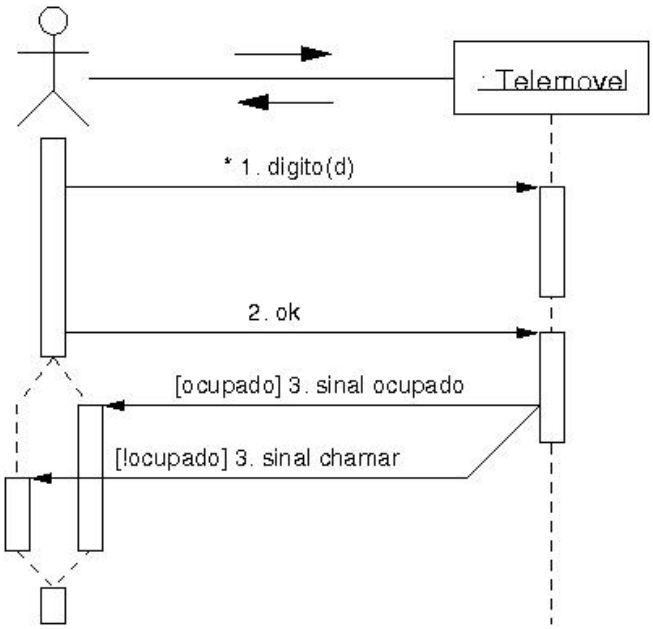
Diagramas de Sequência

- Representam o mesmo que os Diagramas de Colaboração, mas a ênfase é colocada na ordenação temporal das mensagens.
- Permitem analisar a distribuição de "responsabilidade" pelas diferentes classes (analisar onde está a ser efectuado o processamento)



Diagramas de Sequência – Lógica Condicional

- Nos Diagramas de Sequência a modelação de lógica condicional/estruturas repetitivas torna-se mais evidente.



Diagramas de Estado (*Statecharts*)

Sumário

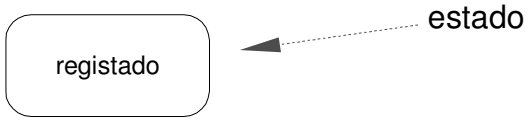
- Introdução aos Diagramas de Estado — Aplicação
- Notação base
  - Estado
  - Estado inicial
  - Estado final
- Estados e super-estados
- Mais sobre transições
- Estados com História
- Sub-estados concorrentes

Introdução aos Diagramas de Estado — Aplicação

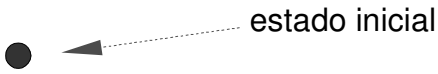
- Com os diagramas de classe podemos descrever a arquitectura de classes do sistema. Nada é dito, no entanto, sobre a forma como os objectos se comportam.
- Os Diagramas de Estado permitem modelar o comportamento de um dado objecto/sistema de forma global.
- A ênfase é colocada no estado do objecto/sistema — 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.
- Devem utilizar-se para classes em que se torne necessário compreender o comportamento do objecto de forma global ao sistema.
- Nem todas as classes vão necessitar de diagramas de estado.

Notação base

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

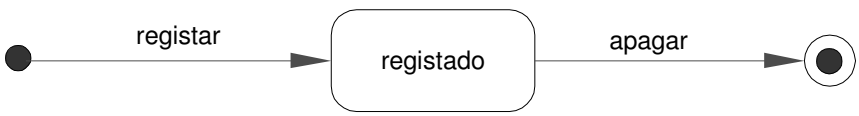


  - Estado inicial — estado do objecto quando é criado



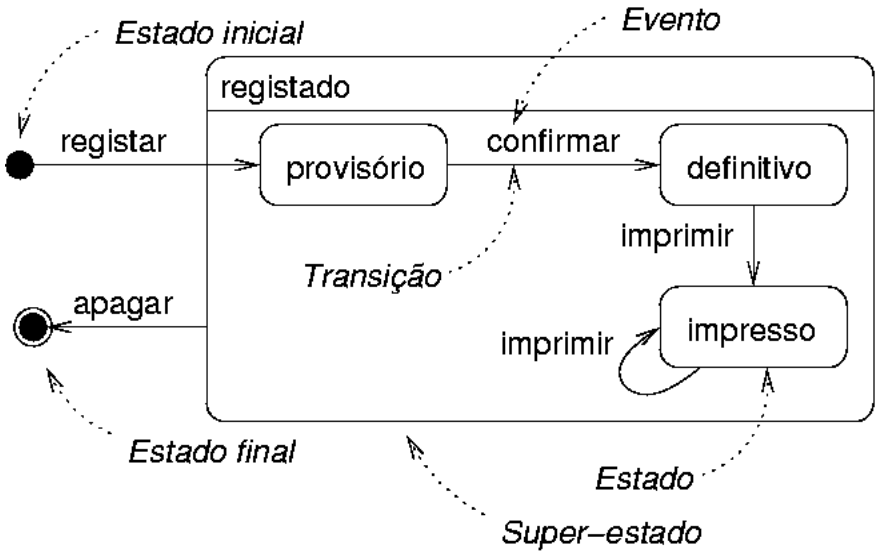
  - Estado final — destruição do objecto

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



Estados e Super-estados

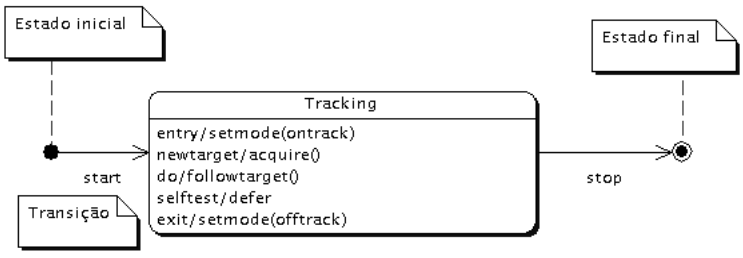
- Super-estado — permitem estruturar os modelos



Mais sobre transições

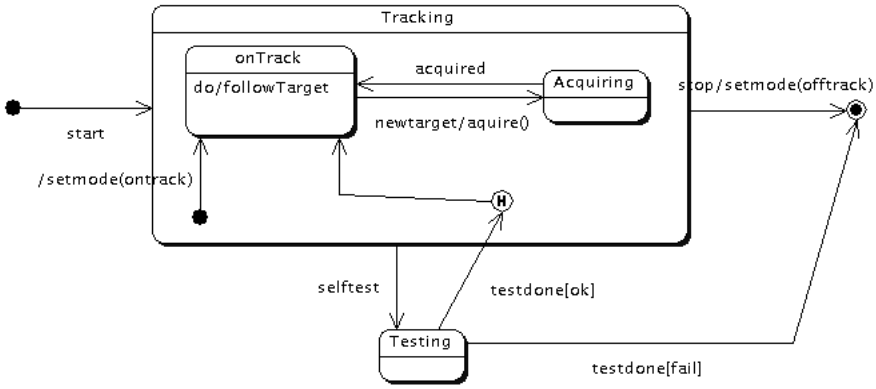
É possível definir comportamento dentro do estado sem associação explícita a transições:

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



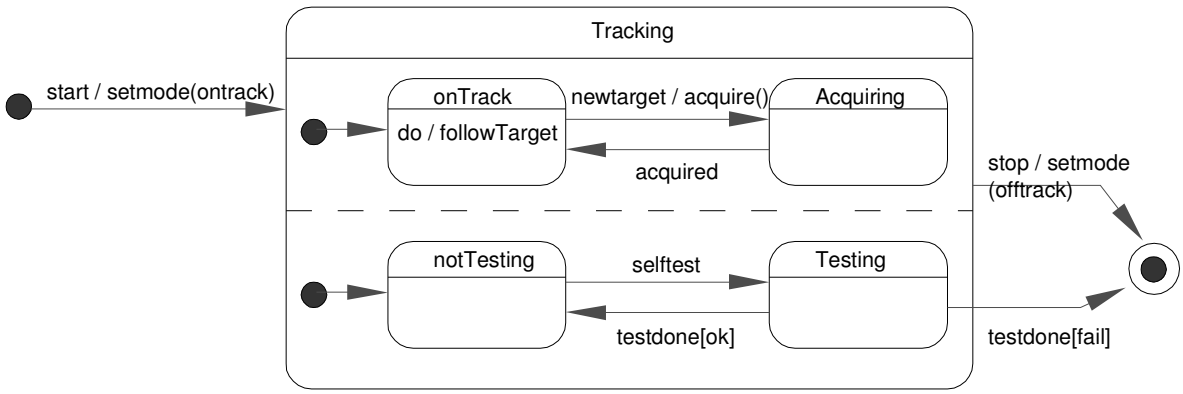
Estados com História

- Permitem modelar interrupções — objecto volta para o estado em que estava anteriormente (H vs. H*)



Sub-estados concorrentes

- **É também possível modelar comportamentos concorrentes**



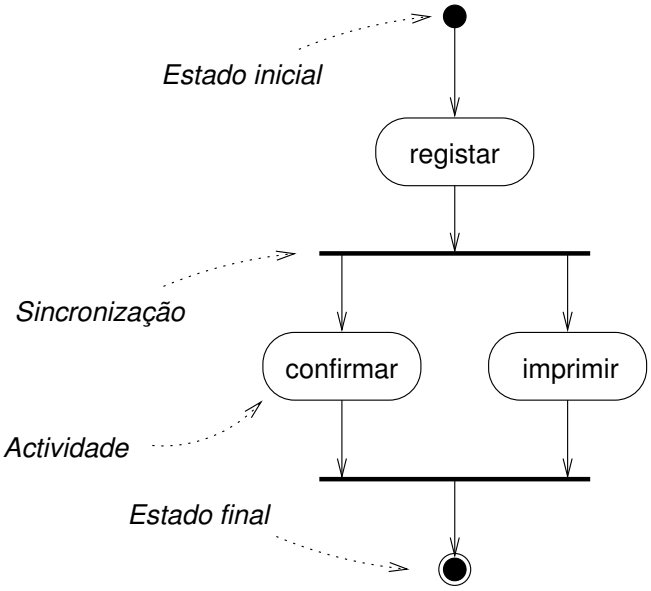
Diagramas de Actividade

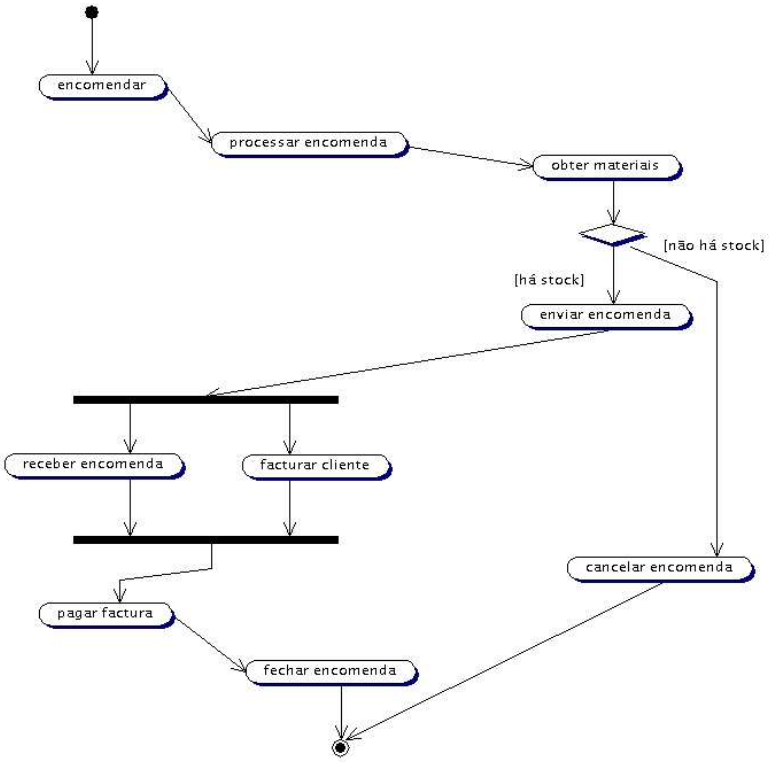
Sumário

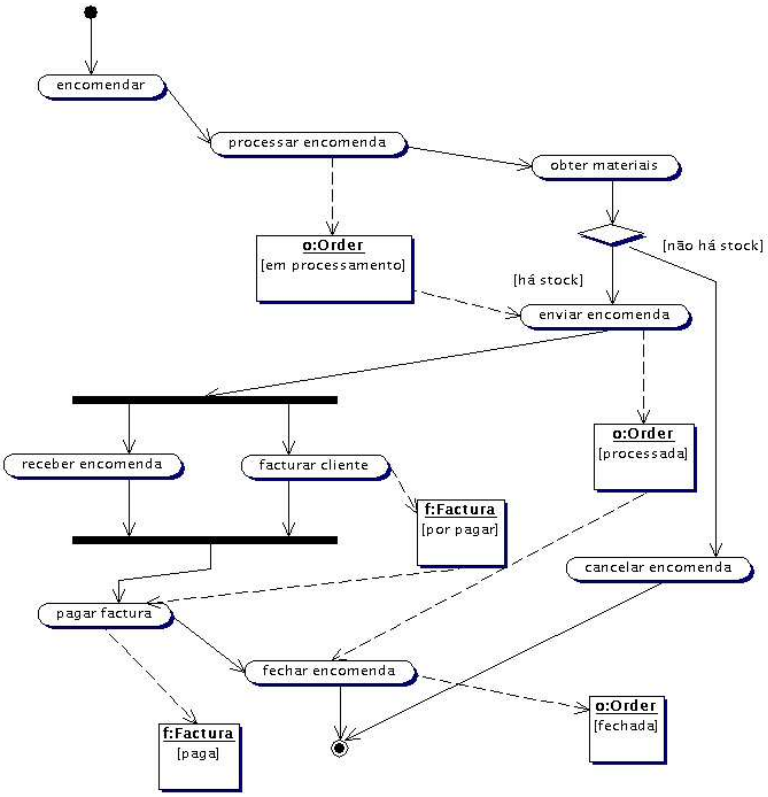
- Aplicação
- Exemplos

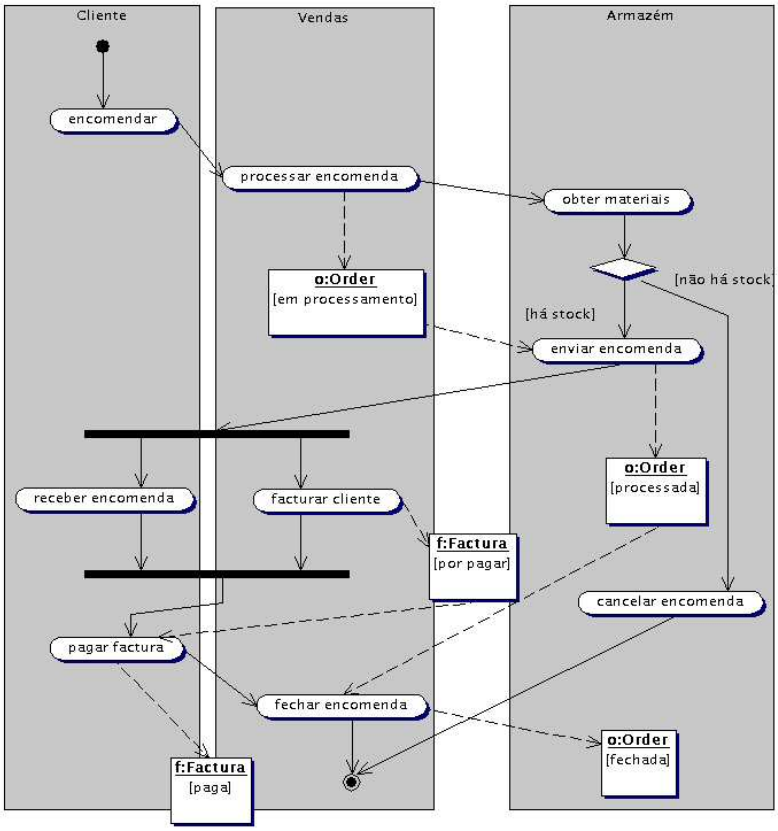
Aplicação

- Diagramas de estado modelam comportamento dando ênfase aos estados que os objectos atravessam.
- Diagramas de actividade são uma variação dos diagramas de estado em que os estados representam actividades.
- Diagramas de actividade modelam comportamento dando ênfase às actividades realizadas.
- Úteis para modelar o comportamento associado a operações de um objecto/sistema.
- Permitem modelar o fluxo de objectos entre as actividades.
- Permitem modelar que é responsável por que actividade.


Estado inicial
registar
Sincronização
confirmar
imprimir
Actividade
Estado final


encomendar
processar encomenda
obter materiais
[não há stock]
[há stock]
enviar encomenda
receber encomenda
facturar cliente
cancelar encomenda
pagar factura
fechar encomenda


encomendar
processar encomenda
obter materiais
o:Order
[em processamento]
[não há stock]
[há stock]
enviar encomenda
o:Order
[processada]
receber encomenda
facturar cliente
f:Factura
[por pagar]
cancelar encomenda
pagar factura
fechar encomenda
o:Order
[fechada]
f:Factura
[paga]


Cliente
Vendas
Armazém
encomendar
processar encomenda
obter materiais
o:Order
[em processamento]
[não há stock]
[há stock]
enviar encomenda
o:Order
[processada]
receber encomenda
facturar cliente
f:Factura
[por pagar]
cancelar encomenda
pagar factura
fechar encomenda
o:Order
[fechada]
f:Factura
[paga]