ENGENHARIA DE SOFTWARE
INTRODUÇÃO

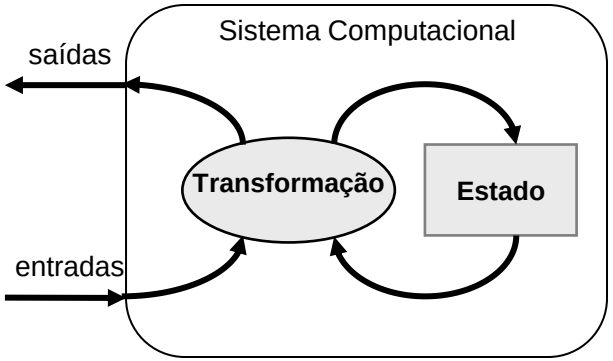
Luís Morgado

2024

MODELAÇÃO DE UM SISTEMA COMPUTACIONAL

- Os vários estados que um sistema pode assumir e a forma como eles evoluem ao longo do tempo para produzir o comportamento do sistema, constitui a **dinâmica** do sistema.
- O **comportamento** do sistema corresponde à forma como o sistema age, ou seja, activa as suas saídas (gera informação de saída), em função das suas entradas (informação de entrada, proveniente do exterior) e do seu estado interno.
- Um sistema computacional geral, pode assim ser caracterizado de forma abstrata, através de uma **representação de estado**, que define em cada momento a configuração interna do sistema, e de uma **função de transformação** que gera as **saídas** e o **próximo estado** em função das **entradas** e do **estado actual** do sistema.

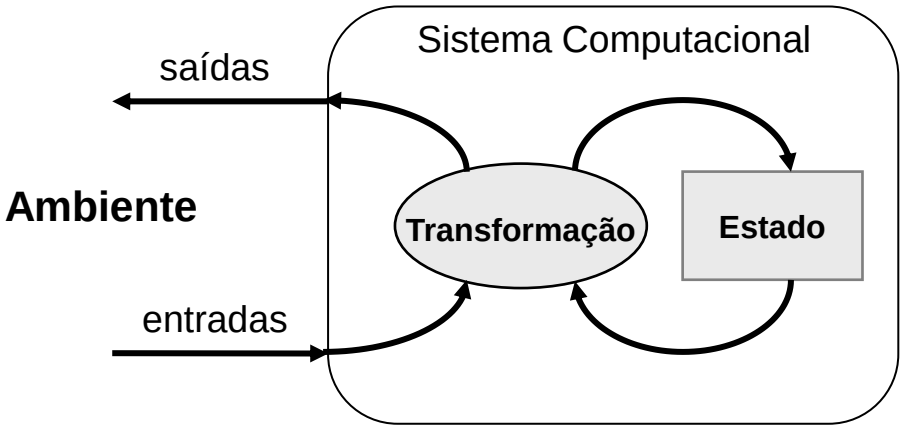
- **Dinâmica**
  - Evolução no **tempo**
  - Descreve os estados que um sistema pode assumir e a forma como eles evoluem ao longo do tempo, determinando o comportamento do sistema

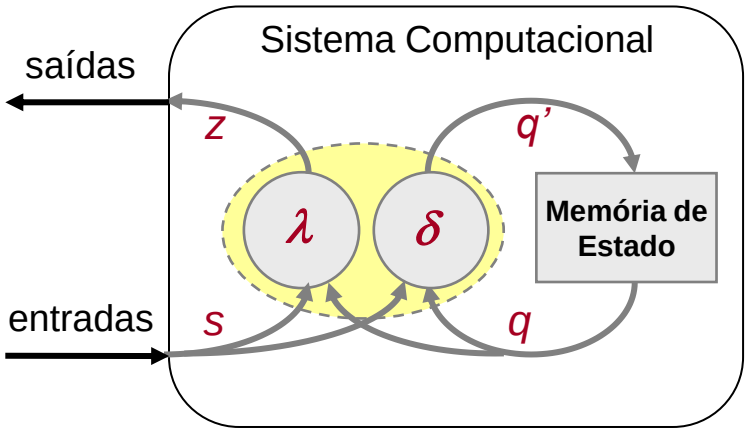


- **Comportamento**
  - Corresponde à forma como o sistema age (gera as saídas) perante a informação proveniente do exterior (entradas)
  - Expressa a estrutura e a dinâmica do sistema

MODELO DE DINÂMICA DE UM SISTEMA

A **dinâmica** pode ser expressa como uma ***função de transformação*** que, perante o estado actual e as entradas actuais, produz o estado seguinte e as saídas seguintes.

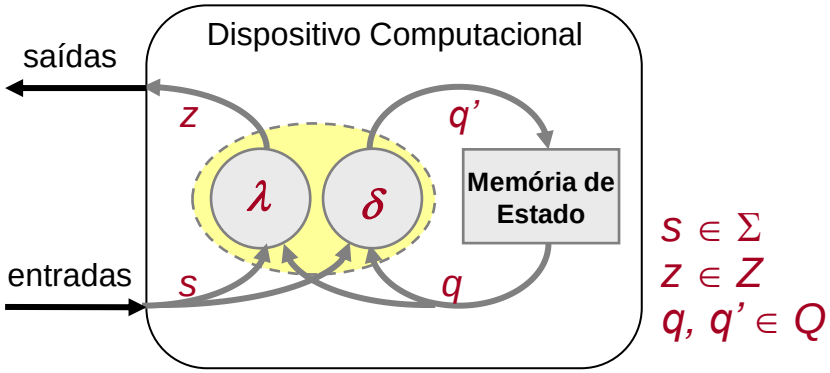




Esta caracterização de um sistema computacional é **independente da forma concreta como este possa ser implementado em termos físicos**. O suporte físico pode ser, por exemplo, mecânico, electrónico, químico.

MODELO FORMAL DE COMPUTAÇÃO

- Entradas e saídas abstraídas em termos dos conjuntos de símbolos que nelas podem ocorrer
  - Esses conjuntos de símbolos são designados ***alfabetos***
  - Consideremos um ***alfabeto de entrada*** Σ e um ***alfabeto de saída*** Z
- **Estado interno** do sistema descrito em termos de um **conjunto de estados possíveis**
  - Q
- **Função de transformação** do sistema descrita com base em duas funções distintas δ e λ
  - **Função de transição de estado**
    - $\delta : Q \times \Sigma \rightarrow Q$
  - **Função de saída**
    - $\lambda : Q \times \Sigma \rightarrow Z$



MODELO FORMAL DE COMPUTAÇÃO

Este tipo de modelo descreve um mecanismo computacional designado ***Máquina de Estados***

- A sua implementação física implica que o número de estados possíveis seja finito
- **Máquinas de Estados Finitos**

Duas formulações distintas da função de saída λ:

- ***Máquinas de Mealy***, nas quais a função de saída depende das entradas

$$\lambda : Q \times \Sigma \rightarrow Z$$

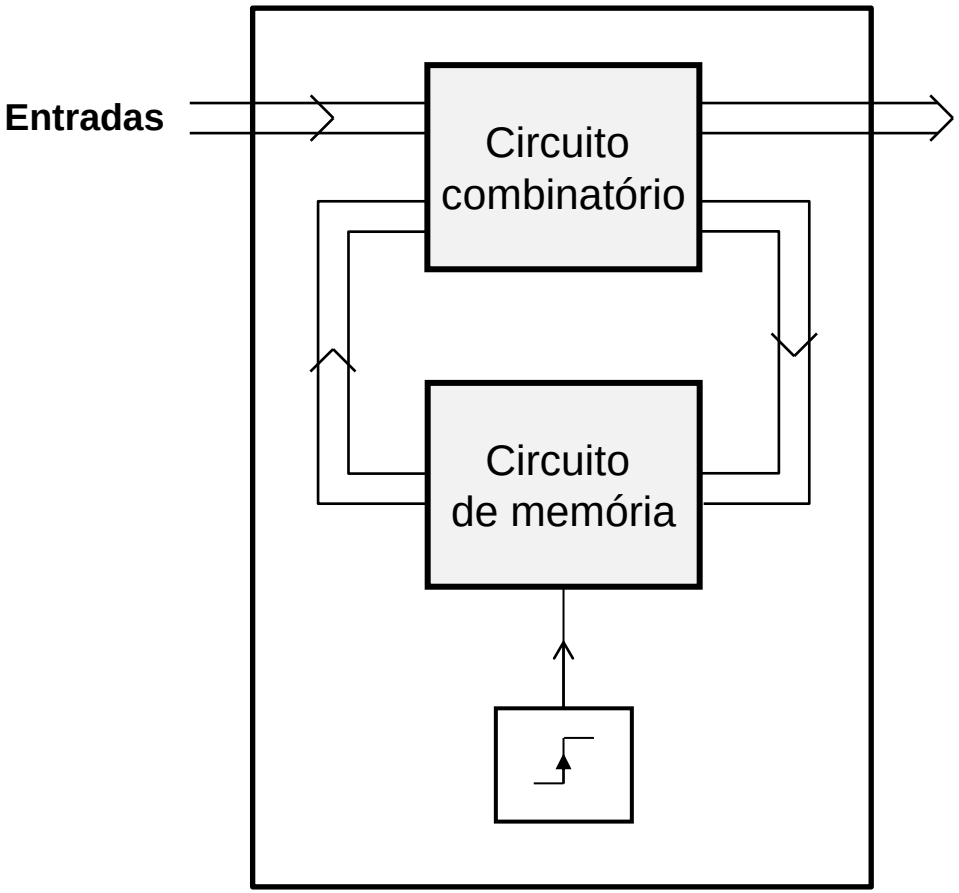
- ***Máquinas de Moore***, nas quais a função de saída não depende das entradas

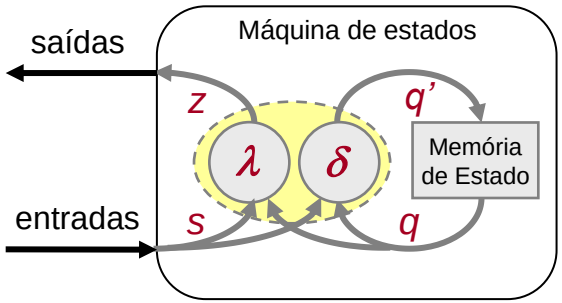
$$\lambda : Q \rightarrow Z$$

EXEMPLO: SISTEMAS DIGITAIS

Modelo geral de um Sistema Digital

[Peatman, 1981]





Um modelo geral de *sistema digital* é um exemplo de concretização do conceito de *máquina de estados*, onde o estado é mantido através de um circuito de memória (um circuito sequencial, com um sinal temporal de referência que determina o ritmo de evolução), e a função de transformação é implementada através de um circuito combinatório que recebe as entradas e a informação de estado, produzindo as saídas e o próximo estado.

Este modelo de circuito digital é a base de organização dos microprocessadores e dos computadores em geral

EXEMPLO: CASO PRÁTICO

Personagem virtual

Pretende-se implementar um jogo com uma personagem virtual que interage com um jogador humano.

O jogo consiste num ambiente onde a personagem tem por objectivo registar a presença de animais através de fotografias.

Quando o jogo se inicia a personagem fica numa situação de procura de animais. Quando detecta algum ruído aproxima-se e fica em inspecção da zona, procurando a fonte do ruído. Quando volta a haver silêncio a personagem volta a uma situação de procura de animais. Quando detecta um animal a personagem aproxima-se e fica em observação. Caso o animal continue presente, a personagem observa o animal e fica preparada para o registo, se ocorrer a fuga do animal a personagem fica em inspecção da zona, à procura de uma fonte de ruído. Na situação de registo, se o animal continuar presente fotografa-o, caso ocorra a fuga do animal ou a personagem tenha conseguido uma fotografia do animal, a personagem fica novamente numa situação de procura.

A interacção com o jogador é realizada em modo de texto.

EXEMPLO: CASO PRÁTICO

Personagem virtual

Eventos
- Silencio
- Ruido
- Animal
- Fuga
- Fotografia
- Terminar

Acções
- Procurar
- Aproximar
- Observar
- Fotografar

Estados
- Procura
- Inspecção
- Observação
- Registo

Conjunto de símbolos de entrada (o **alfabeto de entrada**):

- Σ = { Silencio, Ruido, Animal, Fuga, Fotografia, Terminar }

Conjunto de símbolos de saída (o **alfabeto de saída**):

- Z = { Procurar, Aproximar, Observar, Fotografar }

Conjunto de estados que caracterizam a personagem:

- Q = { Procura, Inspecção, Observação, Registo }

DINÂMICA DA PERSONAGEM

Quando o jogo se inicia a personagem fica numa situação de **procura** de animais. Quando detecta algum **ruído aproxima-se** e fica em **inspecção** da zona, **procurando** a fonte do **ruído**. Quando volta a haver **silêncio** a personagem volta a uma situação de **procura** de animais. Quando detecta um **animal** a personagem **aproxima-se** e fica em **observação.** Caso o **animal** continue presente, a personagem **observa** o animal e fica preparada para o **registo**, se ocorrer a **fuga** do animal a personagem fica em **inspecção** da zona, à procura de uma fonte de ruído. Na situação de **registo**, se o **animal** continuar presente **fotografa-o**, caso ocorra a **fuga** do animal ou a personagem tenha conseguido uma **fotografia** do animal, a personagem fica novamente numa situação de **procura**.

Função de transição de estado

$\delta : Q \times \Sigma \rightarrow Q$

Estado	Evento	Novo estado
Procura	Animal	Observação
Procura	Ruido	Inspecção
Procura	Silencio	Procura
Inspecção	Animal	Observação
Inspecção	Ruido	Inspecção
Inspecção	Silencio	Procura
Observação	Fuga	Inspecção
Observação	Animal	Registo
Registo	Animal	Registo
Registo	Fuga	Procura
Registo	Fotografia	Procura

Tabela de transição de estado

Função de saída

$\lambda : Q \times \Sigma \rightarrow Z$

Estado	Evento	Acção
Procura	Animal	Aproximar
Procura	Ruido	Aproximar
Procura	Silencio	Procurar
Inspecção	Animal	Aproximar
Inspecção	Ruido	Procurar
Inspecção	Silencio	
Observação	Fuga	
Observação	Animal	Observar
Registo	Animal	Fotografar
Registo	Fuga	
Registo	Fotografia	

Tabela de acção de estado

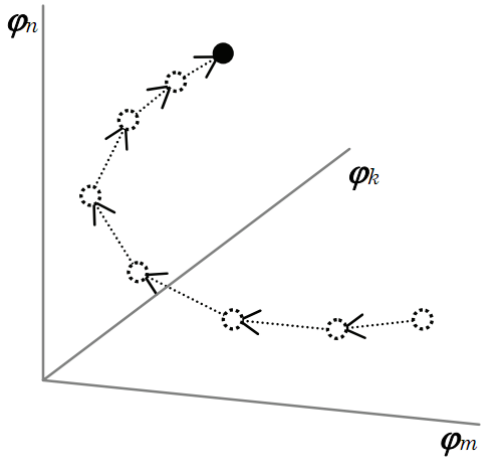
MODELO DE DINÂMICA DE UM SISTEMA

O conjunto de estados de um sistema constituem um *espaço de estados*, com dimensões φ_n que correspondem aos domínios de valores do estado, cada estado corresponde a uma posição nesse espaço.

A evolução do estado de um sistema, pode ser representada geometricamente num espaço de estados sob a forma de trajetórias de evolução de estado, que descrevem padrões de comportamento do sistema independentes do tempo, constituindo uma forma gráfica de descrição do comportamento de um sistema.

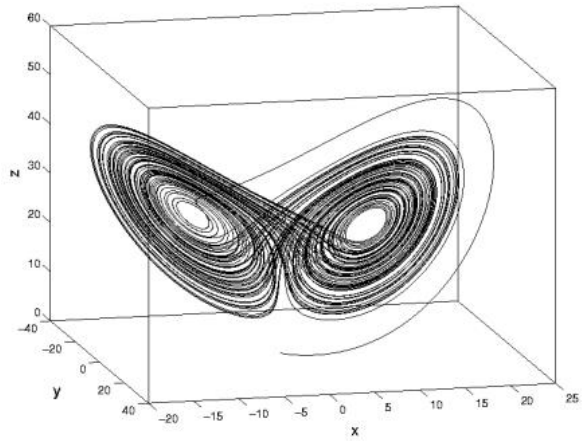
Se os valores de estado forem do domínio real, o espaço de estados é contínuo, sendo nesse caso designado um *espaço de fase*, pois o que é possível discriminar são as fases de evolução do estado do sistema.

Domínios de valores discretos



Espaço de Estados

Domínios de valores contínuos



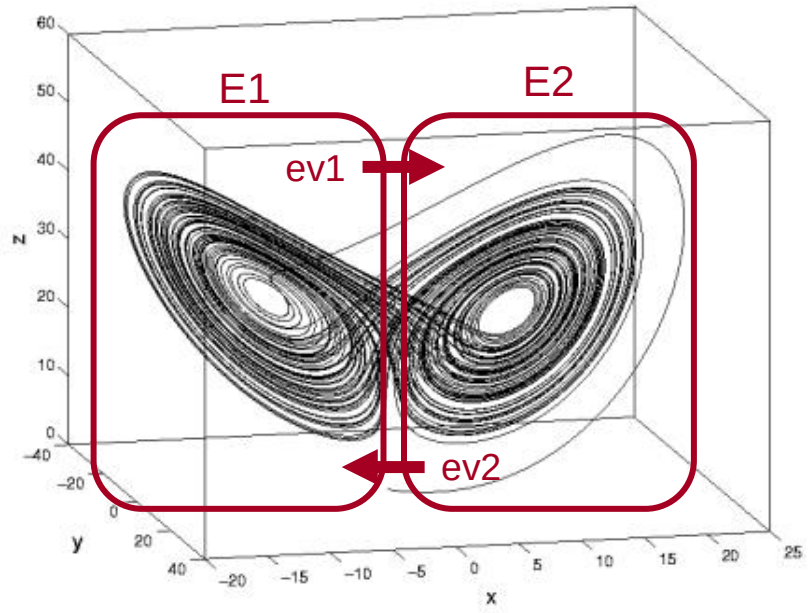
Espaço de Fase

Suporte de descrição do
comportamento de um sistema

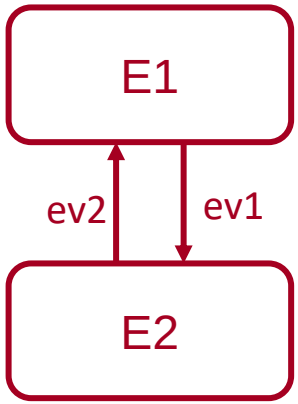
MODELO DE DINÂMICA DE UM SISTEMA

No caso de um sistema caracterizado por um *espaço de fase* (contínuo), é possível modelar os padrões gerais de evolução de estado, abstraindo conjuntos de trajetórias que correspondem ao mesmo tipo de comportamento como **estados** abstractos, por exemplo, num espaço de fase de um modelo atmosférico podem ser identificados estados abstractos como *céu limpo* ou *céu nublado*.

As transições entre estados podem ser abstraídas por **eventos**, os quais correspondem a ocorrências que determinam a transição entre estados, por exemplo, a ocorrência de um determinado *nível de pressão atmosférica*.



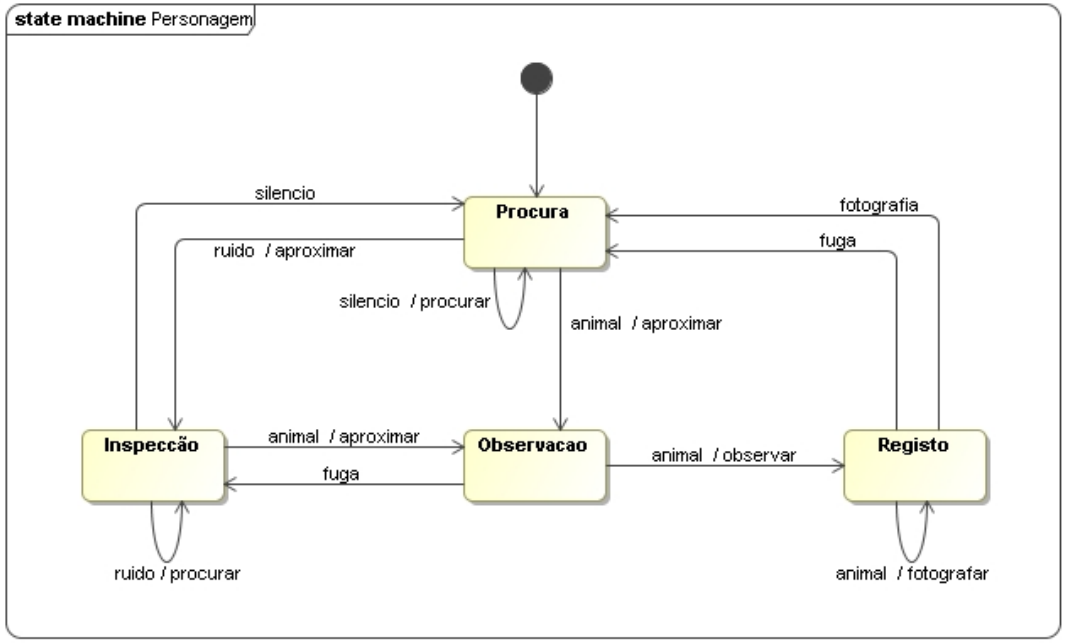
Representação abstracta



DIAGRAMAS DE TRANSIÇÃO DE ESTADO

A **linguagem UML** suporta a representação gráfica da dinâmica de um sistema sob a forma de *diagramas de transição de estado*

A representação gráfica facilita a descrição e compreensão da dinâmica e comportamento de um sistema



Conversão para uma máquina de estados

Função de transição de estado

$\delta : Q \times \Sigma \rightarrow Q$

Estado	Evento	Novo estado
Procura	Animal	Observação
Procura	Ruido	Inspecção
Procura	Silencio	Procura
Inspecção	Animal	Observação
Inspecção	Ruido	Inspecção
Inspecção	Silencio	Procura
Observação	Fuga	Inspecção
Observação	Animal	Registo
Registo	Animal	Registo
Registo	Fuga	Procura
Registo	Fotografia	Procura

Função de saída

$\delta : Q \times \Sigma \rightarrow Z$

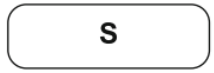
Estado	Evento	Acção
Procura	Animal	Aproximar
Procura	Ruido	Aproximar
Procura	Silencio	Procurar
Inspecção	Animal	Aproximar
Inspecção	Ruido	Procurar
Inspecção	Silencio	
Observação	Fuga	
Observação	Animal	Observar
Registo	Animal	Fotografar
Registo	Fuga	
Registo	Fotografia	

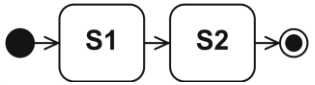
DIAGRAMAS DE TRANSIÇÃO DE ESTADO

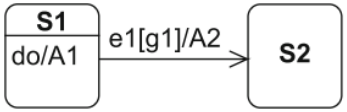
REPRESENTAÇÃO DE COMPORTAMENTO

Modelo de dinâmica

- **Estado**
  - Representação da situação de evolução de um sistema ou parte de um sistema
- **Pseudo-estado**
  - Símbolos utilizados com significado específico para definição de semântica adicional
    - **Início**: Representa a fonte de transição inicial da máquina
    - **Fim**: Representa o destino para a transição final da máquina
- **Transição**
  - Acontecimento através do qual o sistema evolui do estado actual para um novo estado
  - **Evento**
    - Ocorrência no tempo e no espaço com significado para a evolução de estado
  - **Guarda**
    - Condição que inibe ou permite transições ou acções
- **Acção**
  - Define comportamento
    - Associado a **transição**
    - Associado a **estado**







[Seidl, 2012]

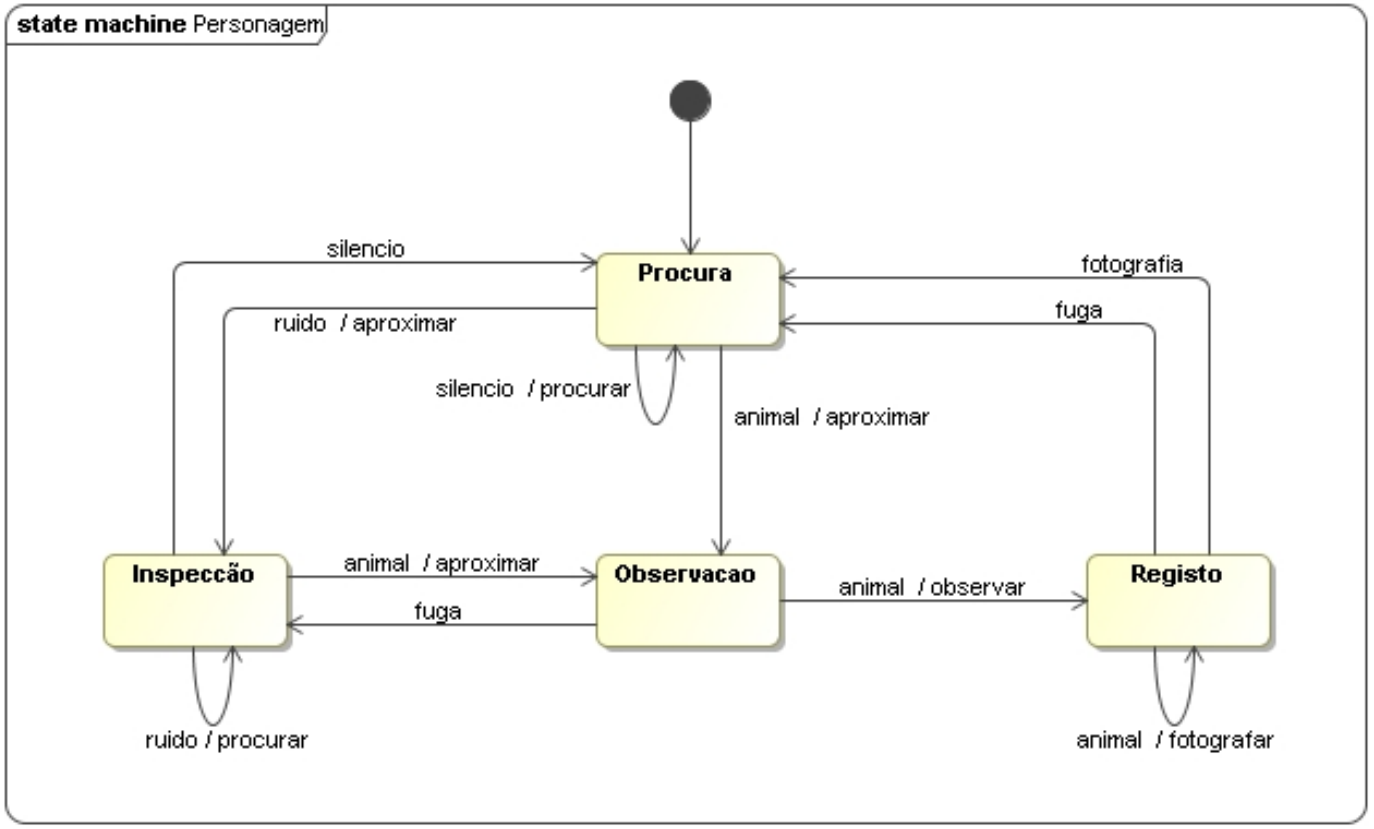
EXEMPLO: COMPORTAMENTO DA PERSONAGEM

Quando o jogo se inicia a personagem fica numa situação de **procura** de animais. Quando detecta algum **ruído aproxima-se** e fica em **inspecção** da zona, **procurando** a fonte do **ruído**. Quando volta a haver **silêncio** a personagem volta a uma situação de **procura** de animais. Quando detecta um **animal** a personagem **aproxima-se** e fica em **observação.** Caso o **animal** continue presente, a personagem **observa** o animal e fica preparada para o **registo**, se ocorrer a **fuga** do animal a personagem fica em **inspecção** da zona, à procura de uma fonte de ruído. Na situação de **registo**, se o **animal** continuar presente **fotografa-o**, caso ocorra a **fuga** do animal ou a personagem tenha conseguido uma **fotografia** do animal, a personagem fica novamente numa situação de **procura**.

Eventos do ambiente
- Silencio
- Ruido
- Animal
- Fuga
- Fotografia
- Terminar

Acções da personagem
- Procurar
- Aproximar
- Observar
- Fotografar

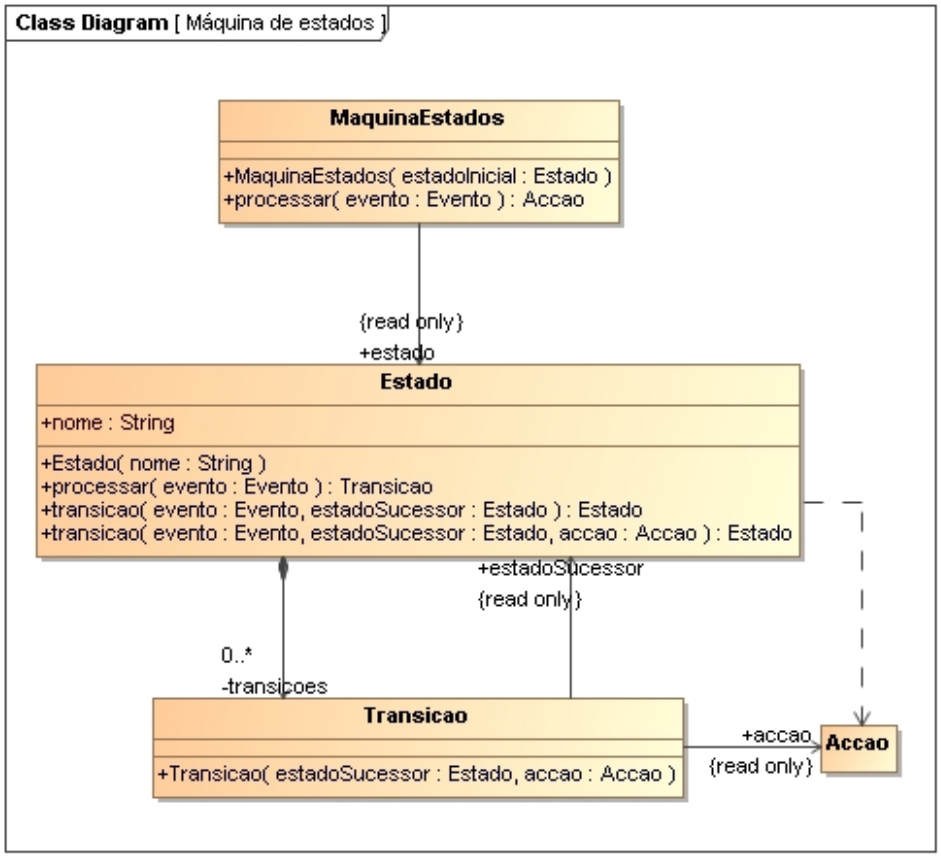
Estados da personagem
- Procura
- Inspecção
- Observação
- Registo



IMPLEMENTAÇÃO: MÁQUINA DE ESTADOS

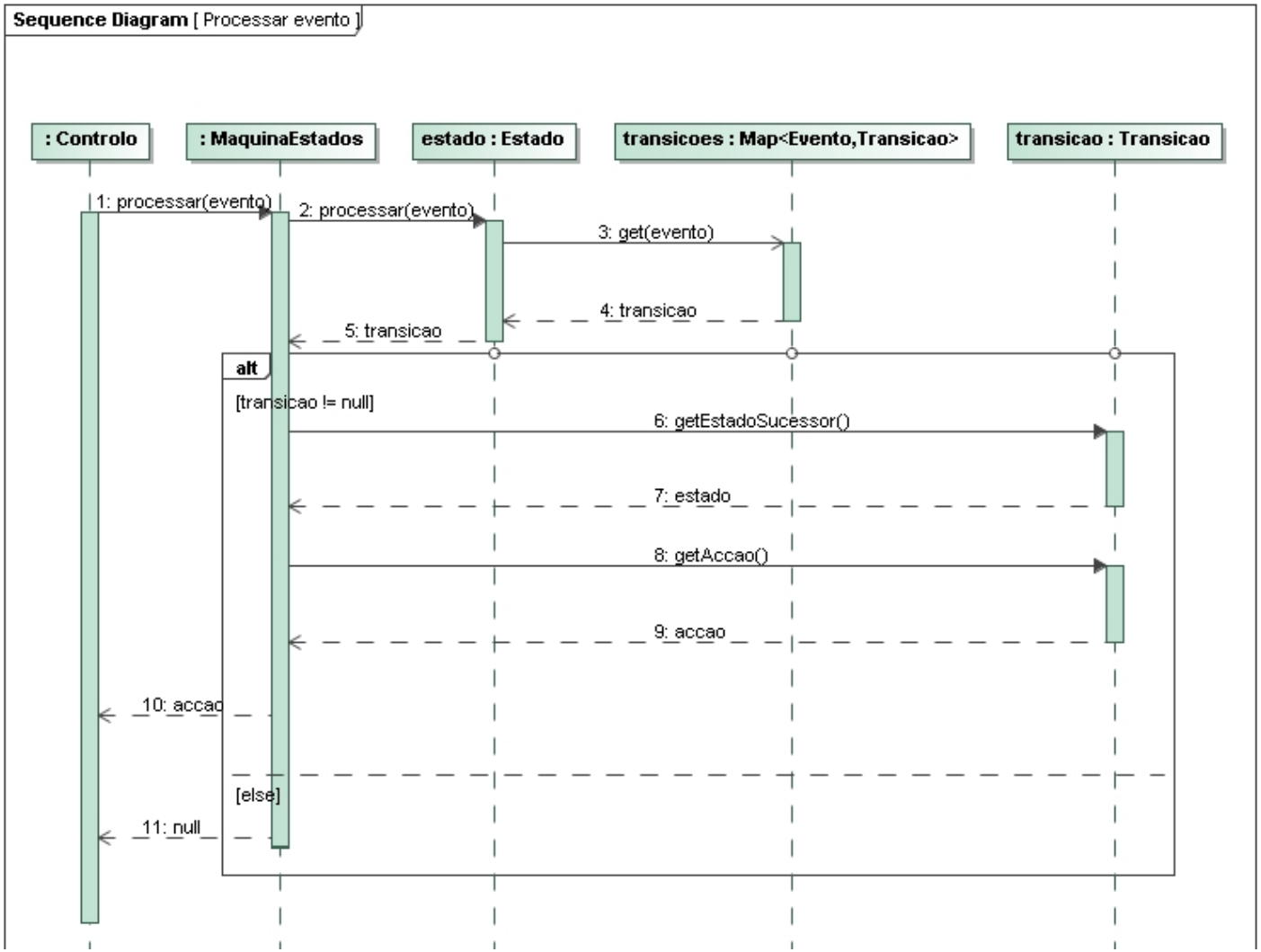
- **Conceitos principais**
  - **Máquina de estados**
    - Estado (actual)
    - Processar evento, gerando uma acção
  - **Estado**
    - Transições associadas a cada evento
    - Definir transição
    - Processar evento, gerando uma transição
  - **Transição**
    - Estado sucessor
    - Acção (de transição)

Modelo de estrutura

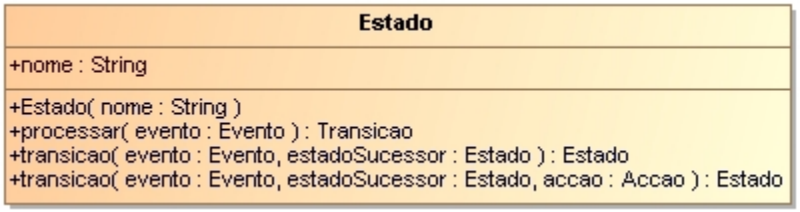


IMPLEMENTAÇÃO: MÁQUINA DE ESTADOS

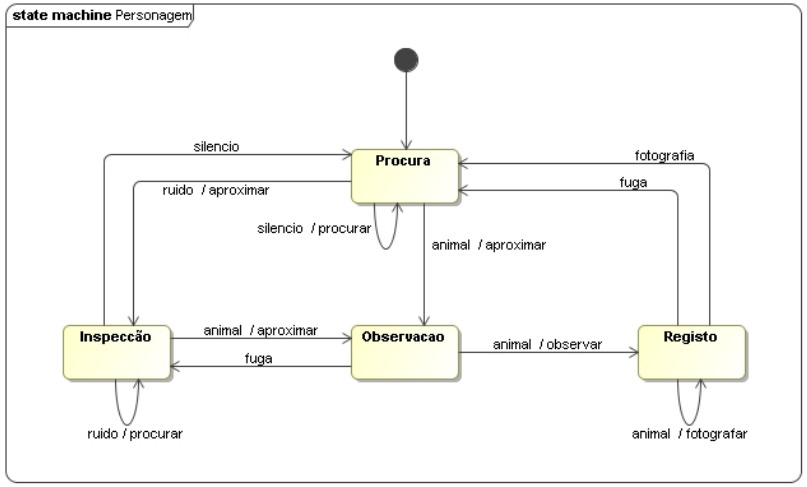
Modelo de interacção (definição de comportamento)



COMPORTAMENTO DA PERSONAGEM



Exemplo de definição da dinâmica do comportamento da personagem



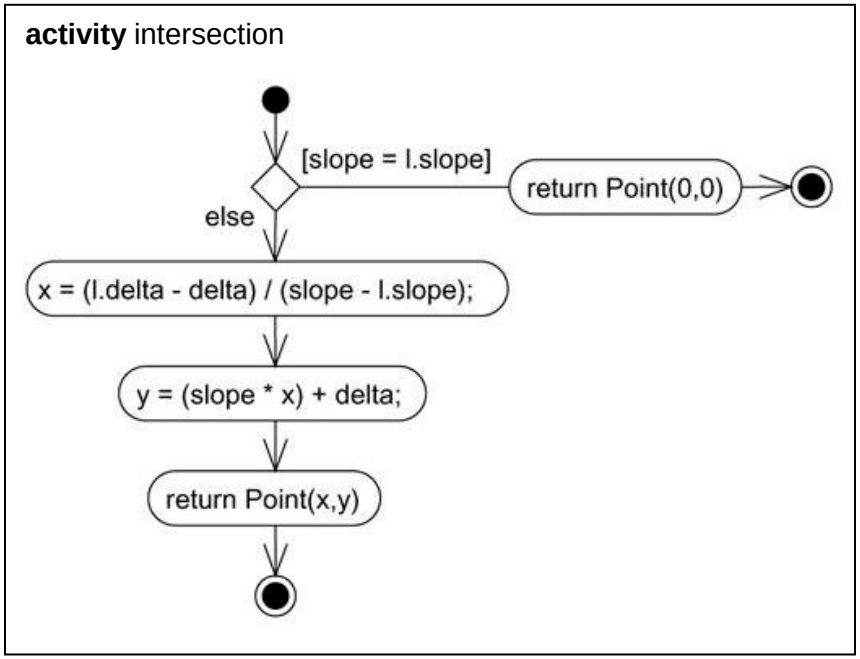

```
// Definir estados
Estado procura = new Estado("Procura");
Estado inspeccao = new Estado("Inspecção");
Estado observacao = new Estado("Observação");
Estado registo = new Estado("Registo");

// Definir transições
procura
    .transicao(ANIMAL, observacao, APROXIMAR)
    .transicao(RUIDO, inspeccao, APROXIMAR)
    .transicao(SILENCIO, procura, PROCURAR);

inspeccao
    .transicao(ANIMAL, observacao, APROXIMAR)
    .transicao(RUIDO, inspeccao, PROCURAR)
    .transicao(SILENCIO, procura);
```

DIAGRAMAS DE ACTIVIDADE

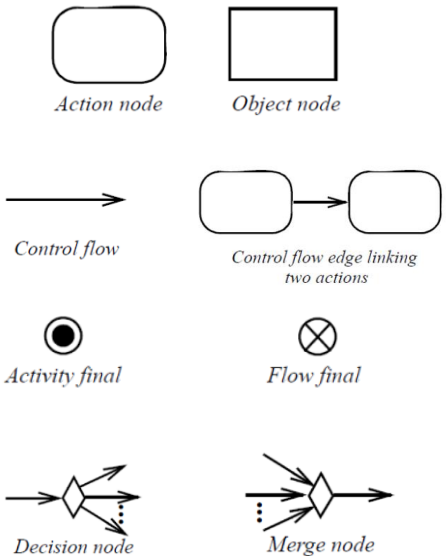
REPRESENTAÇÃO DE COMPORTAMENTO

Representação de comportamento através da descrição do **fluxo de controlo**, ou seja, através da descrição da **sequências de acções e condições** associadas para realizar uma determinada função

Servem para definir o *modelo de dinâmica* de um sistema



Principais elementos



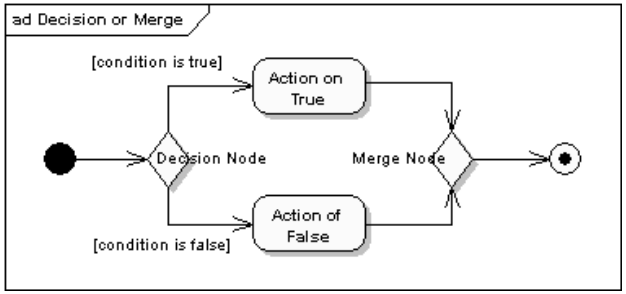
Conversão para código

```
Point intersection(Line l) {
    if(slope == l.slope)
          return Point(0,0);
    float x = (l.delta - delta) / (slope - l.slope);
    float y = (slope * x) + delta;
    return new Point(x, y);
}
```

DIAGRAMAS DE ACTIVIDADE

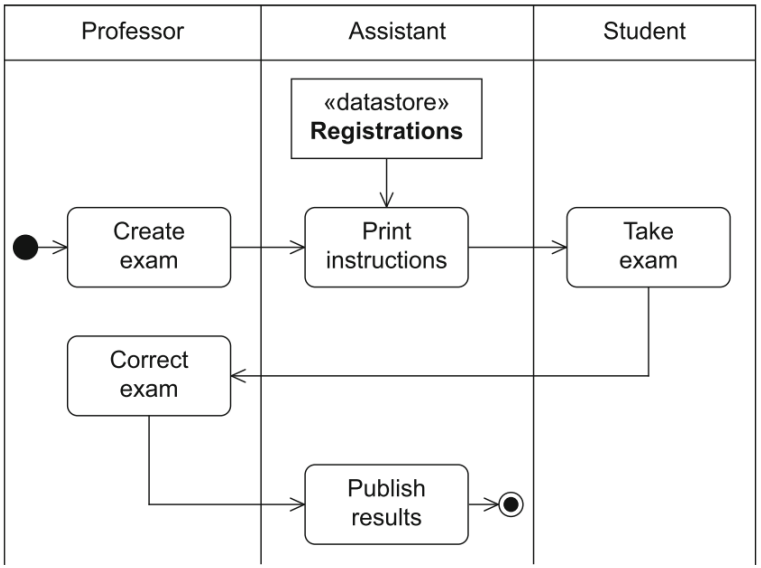
DECISÕES / JUNÇÕES

- **Decisão**: nó de bifurcação do fluxo de controlo de execução
  - **Guarda**: condição que determina o fluxo a executar
- **Junção**: nó de junção de fluxos de controlo de execução



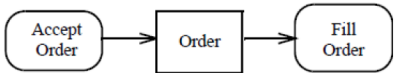
PARTIÇÕES (*Swimlanes*)

- **Partição**: representa uma área de responsabilidade específica de uma actividade
- Pode ser implementada por uma ou mais classes.



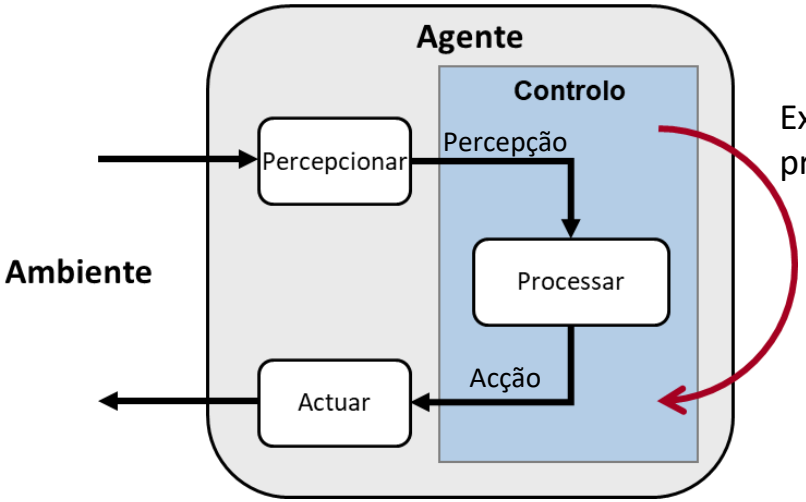
FLUXO DE OBJECTOS

- Transferência de informação entre actividades



[Seidl, 2012]

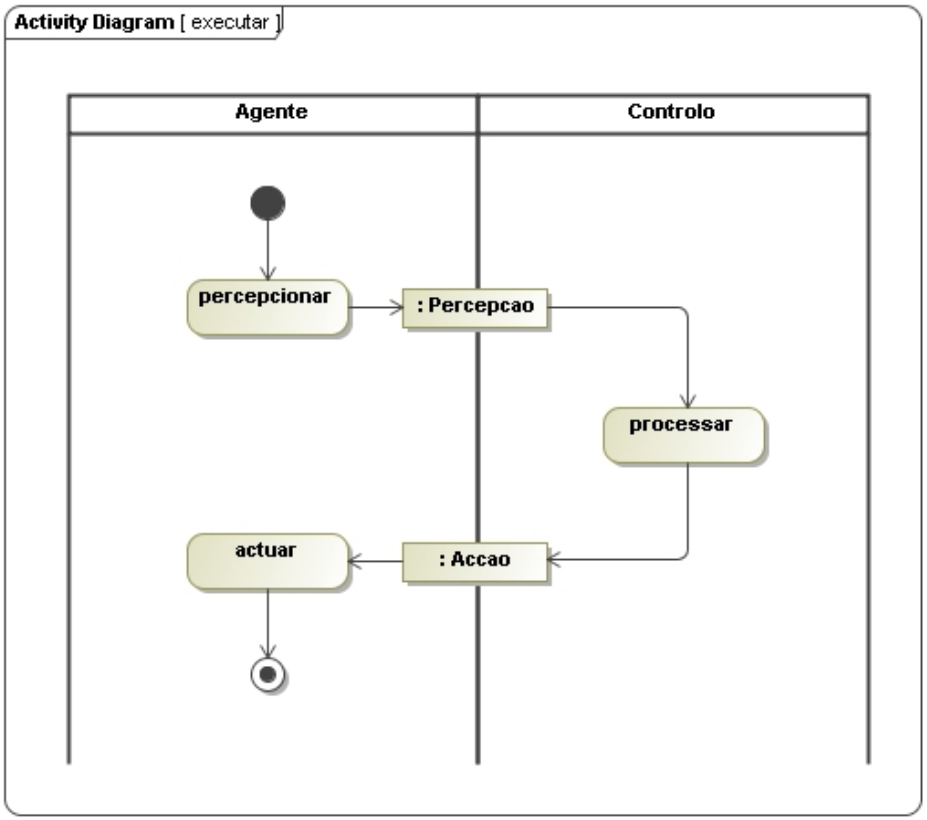
EXEMPLO: MODELO DE AGENTE



Executar passo de processamento

Modelo de dinâmica

(Diagrama de actividade)



Modelo de dinâmica com transferência de informação entre actividades (**Fluxo de objectos**)

BIBLIOGRAFIA

[Pressman, 2003]

R. Pressman, *Software Engineering: a Practitioner's Approach,* McGraw-Hill, 2003.

[Peatman, 1981]

J. Peatman, *The Design of Digital Systems*, McGraw-Hill, 1981.

[Booch *et al*., 1998]

G. Booch, J. Rumbaugh, I. Jacobson, *The Unified Modeling Language User Guide*, Addison Wesley, 1998.

[Miles & Hamilton, 2006]

R. Miles, K. Hamilton, *Learning UML 2.0*, O'Reilly, 2006.

[Eriksson et al., 2004]

H. Eriksson, M. Penker, B. Lyons, D. Fado, *UML 2 Toolkit*, Wiley, 2004.

[Seidl, 2012]

UML Classroom: An Introduction to Object-Oriented Modeling, M. Seidl *et al.,* Springer, 2012

[Douglass, 2009]

B. Douglass, *Real-Time Agility: The Harmony/ESW Method for Real-Time and Embedded Systems Development*, Addison-Wesley, 2009.

[Martin, 2003]

J. Martin, *Introduction to Languages and the Theory of Computation*, McGraw-Hill, 2003.

[Sipser, 2005]

M. Sipser, *Introduction to the Theory of Computation*, Thomson, 2005.