



# Testes baseados na especificação - modelos de estado-

Criado: abril 2001  
Últ. atualiz.: set / 2010



# Referências

R.Binder. *Testing OO Systems*, 2000.

A.P.Mathur. *Foundations of Software Testing*. Pearson Education Editora, 2008, cap. 3.

H.Robinson. “Graph Theory in Model-based Testing”. Obtido em set/20010 em <http://www.harryrobinson.net/>  
[http://www.geocities.com/harry\\_robinson\\_testing/graph\\_theory.htm](http://www.geocities.com/harry_robinson_testing/graph_theory.htm)

M. Utting, B. Legeard. *Practical Model-Based Testing*.Morgan Kaufmann Publishers, 2007.

C.Nagle. “Test Automation Frameworks”, 2000. Obtido em set/2009 em:  
<http://testpro.com.au/whitepapers/Test-Automation-Frameworks-by-Carl-Nagle.pdf>



# Tópicos

- Testes baseados em modelos: conceito
- Modelo de estados: apresentação
- Características
- Propriedades
- Testes de transição de estados



# Testes caixa preta

Especificação:  
Requisitos  
Projeto

Independente de notação

Partição de equivalência  
Valores Limite  
Grafo causa-efeito  
Tabela de decisão

Dependente de notação

Baseada em modelo  
Baseada em linguagem de especificação



# Testes caixa preta

Especificação:  
Requisitos  
Projeto

Independente de notação

Partição de equivalência  
Valores Limite  
Grafo causa-efeito  
Tabela de decisão

Dependente de notação

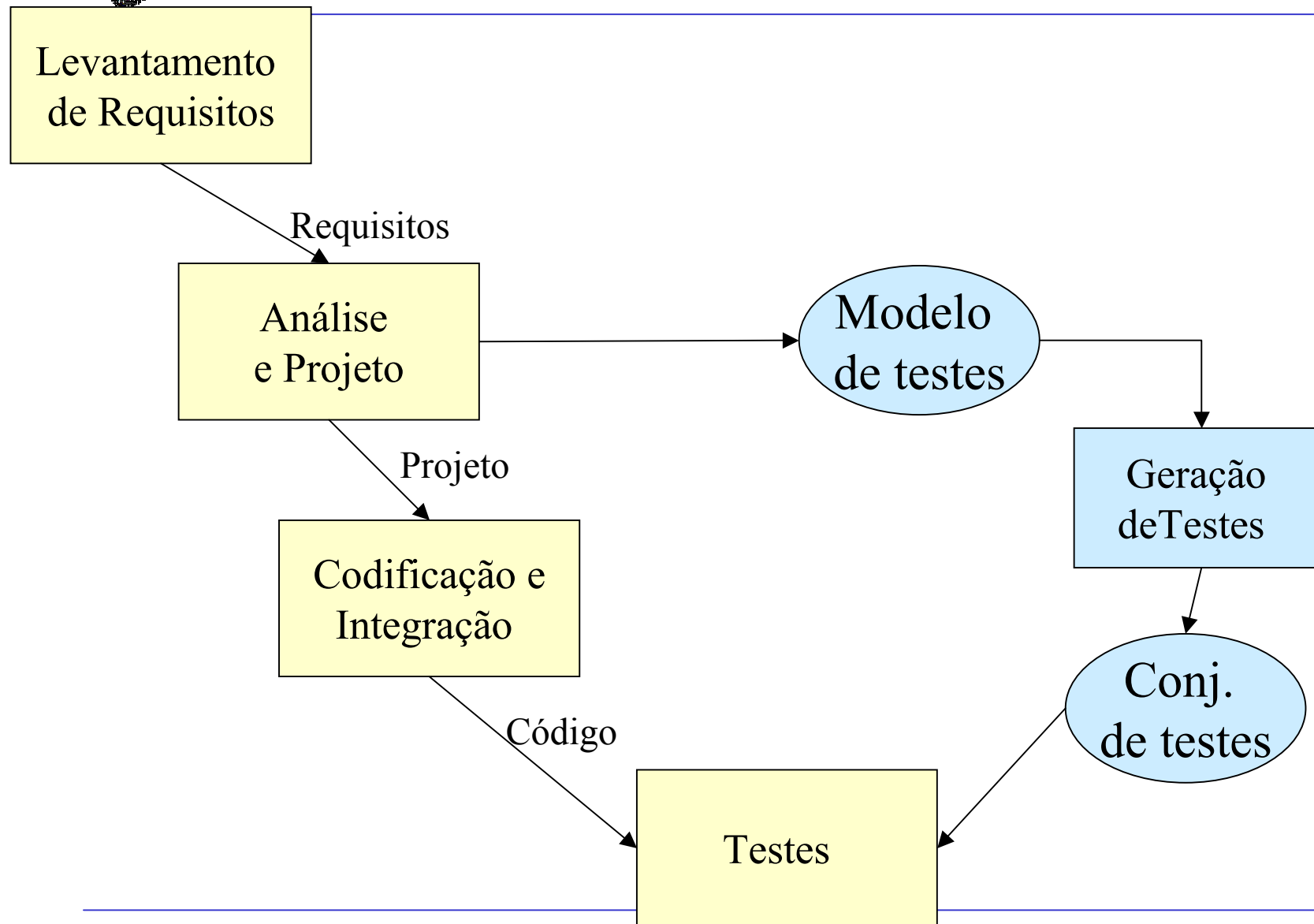
Baseada em modelo

Baseada em linguagem de especificação



## Testes baseados em modelos

- Nome dado a um conjunto de técnicas de testes baseados em **modelos de comportamento** do software.
- O modelo é obtido a partir dos requisitos.
- Vantagens:
  - Geração de testes começa cedo no ciclo de desenvolvimento
  - Possibilidade de automação da geração de testes



Testes baseados em modelos de estado



# Tipos de modelos

- **Tabelas/Árvores de decisão**: mostram conjuntos de condições de entrada e as ações resultantes
- **Máquinas Finitas de Estados**: mostram o comportamento do sistema em termos de estados e das transições entre estados
- **Gramáticas**: descrevem a sintaxe de entrada
- **Statecharts**: extensão das Máquinas Finitas de Estados contendo hierarquia, concorrência, entre outros.
- **Diagramas de casos de uso**: descrevem as funções realizadas por diferentes atores do sistema
- **Diagramas de Atividades**: mostram as atividades que o sistema pode realizar e a relação de ordem entre elas
- ...





## Modelo de testes

- O modelo deve atender aos objetivos dos testes
  - De acordo com o Plano de Testes
- O modelo deve ter uma sintaxe e semântica precisas → modelo testável
  - Tratável por ferramentas



# Modelo de estados



# O que é

- Máquinas Finitas de Estado (MFE)
    - aplicação na Engenharia (hw e sw) de modelo matemático denominado **autômato finito**
    - são usadas desde os anos 50 para modelar circuitos
    - usadas desde os anos 60 para modelar sw
      - protocolos de comunicação
      - analisadores de sintaxe
      - sistemas de controle
      - interfaces-usuário
      - comportamento de objetos
- reconhecedor de  
linguagens regulares

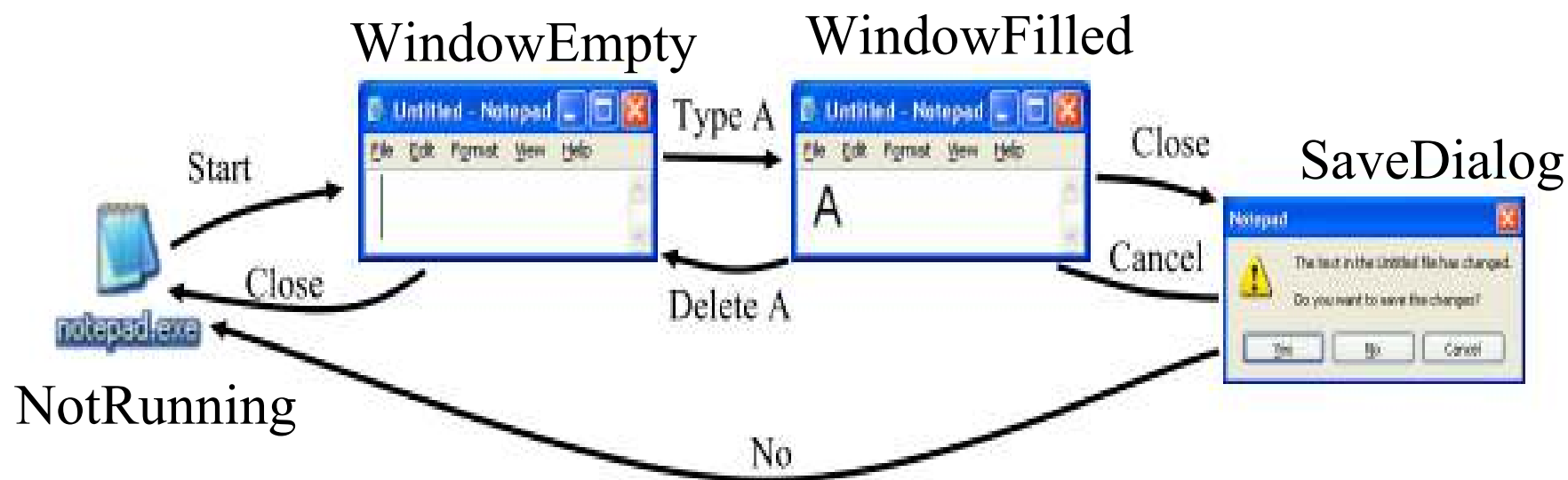


## O que é (cont.)

- Modelo de comportamento formado por estados, transições e ações
- O **estado** armazena informações sobre o **passado**
- As **transições** indicam **mudanças de estado**
- As **ações** representam **atividades** que podem ser realizadas em um determinado momento.



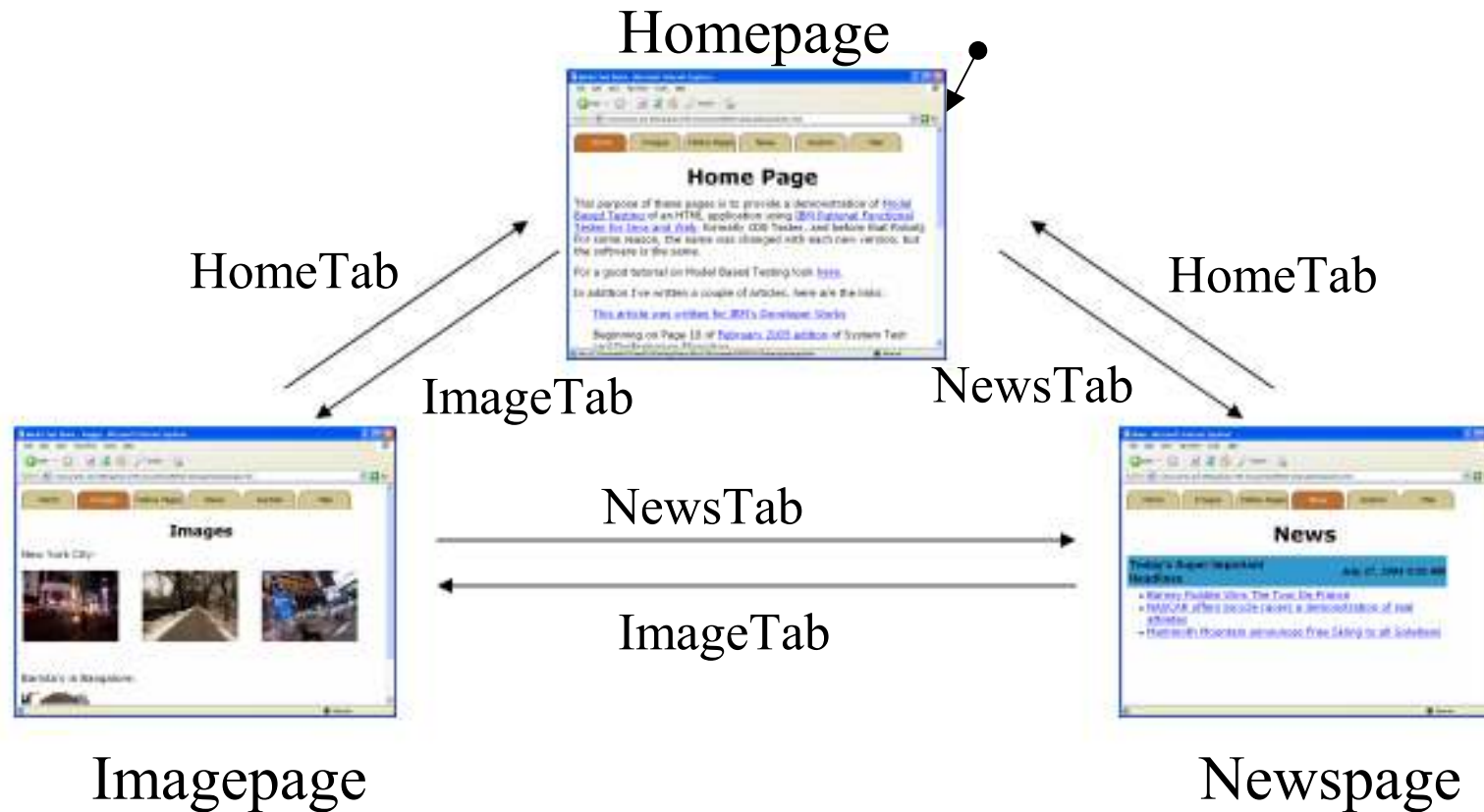
## Exemplo de modelo de estados: Notepad



Fonte: H. Robinson, StarWest 2006



# Exemplo de modelos de estados: Web site



Fonte: H. Robinson, StarWest 2006



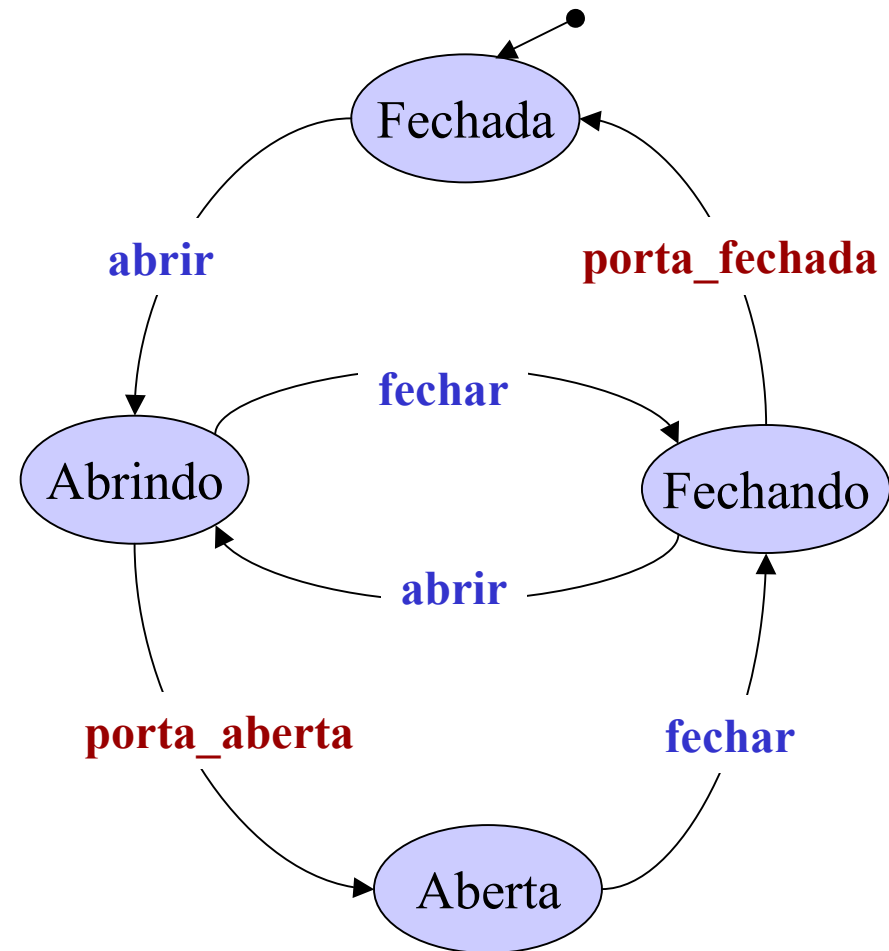
## Classificação

- Há dois grupos distintos de máquinas:
  - Modelo de Moore
  - Modelo de Mealy



## Modelo de Moore

- As saídas dependem unicamente dos estados
- As ações são executadas quando se entra no estado
- Ex.: sistema de controle de uma porta de garagem [Rumbaugh et al 91]

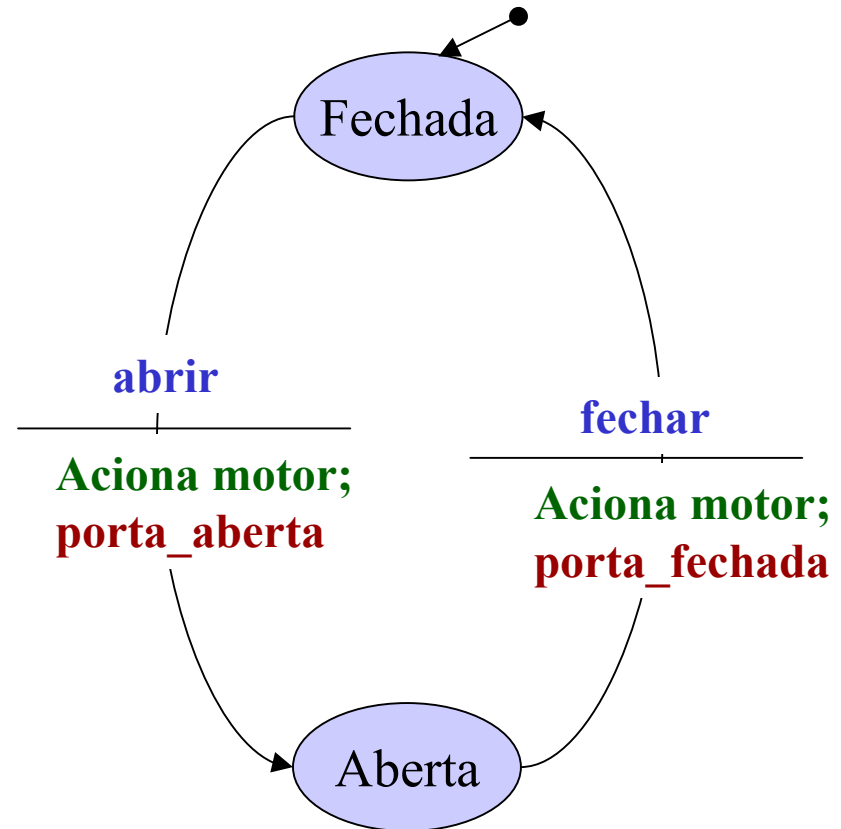






## Modelo de Mealy

- As saídas dependem das entradas e dos estados
- As ações são executadas conforme a entrada fornecida





# Elementos de uma MFE

- Estado
  - conjunto de valores dos dados do sistema em um determinado momento
- Transição
  - leva o sistema de um estado para outro devido à ocorrência de um evento
- Evento
  - entrada ou período de tempo
- Ação
  - Atividade a ser realizada em um determinado momento. No modelo de Mealy, elas ocorrem em resposta a uma entrada.



## Elementos de uma MFE (cont.)

- Estado inicial
  - estado do sistema (ou componente) em que o 1º evento é aceito
- Estado origem / estado destino
  - uma transição leva o sistema de um estado origem a um estado destino, os quais podem ser iguais
- Estado atual
  - estado corrente em que se encontra a execução do sistema
- Estado final
  - estado do sistema no qual eventos não são mais aceitos. O sistema pode ter 0 ou mais estados finais. No tipo transdutor, em geral, não existem estados finais.



# Tópicos

- Modelo de estados: apresentação
- **Características**
- Propriedades
- Testes de transição de estados



## Semântica (modelo de Mealy)

- A máquina inicia no estado inicial.
- A máquina espera por um evento durante um tempo indeterminado.
- A máquina recebe um evento.
- Se o evento não é aceito no estado corrente da máquina, ele é ignorado.
- Se o evento é aceito no estado atual: a transição correspondente é disparada, a ação associada é ativada e o estado designado como próximo torna-se o estado atual (pode ser o mesmo).
- Os passos anteriores se repetem até que a máquina chegue a um estado final ou volte ao estado inicial.

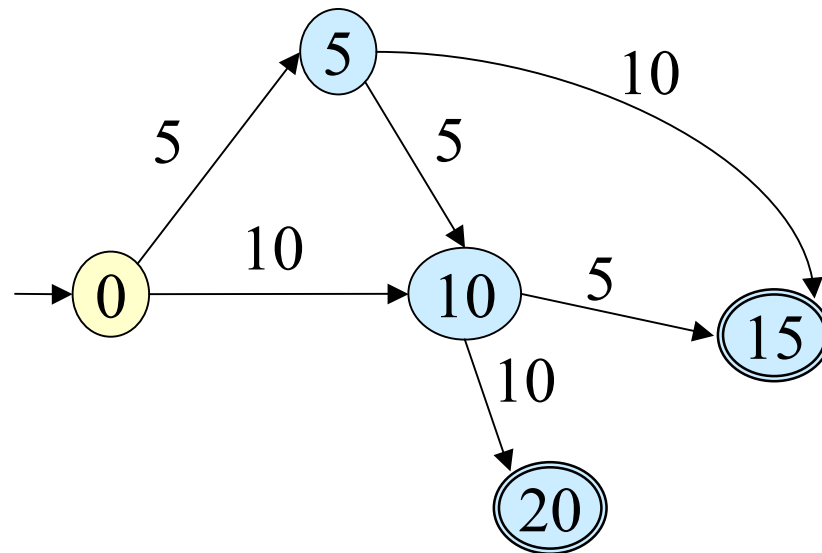


## Um modelo de estados ...

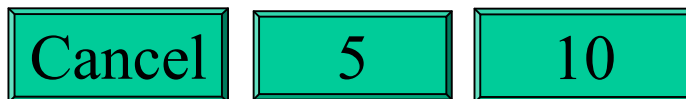
- Não leva em conta a maneira pela qual um evento é produzido
- Trata um evento por vez. Uma única transição pode ser disparada em um dado momento
- Não aceita nenhum evento além daqueles especificados
- Só pode estar em um único estado em um dado momento
- É estático: estados, eventos, transições e ações não podem ser criados nem removidos quando a máquina é executada
- Não descreve como uma ação produz uma saída
- Não tem intervalo de tempo associado a nenhum aspecto do modelo. O disparo de transições é considerado atômico, ie, não consome tempo



## Exemplo de funcionamento

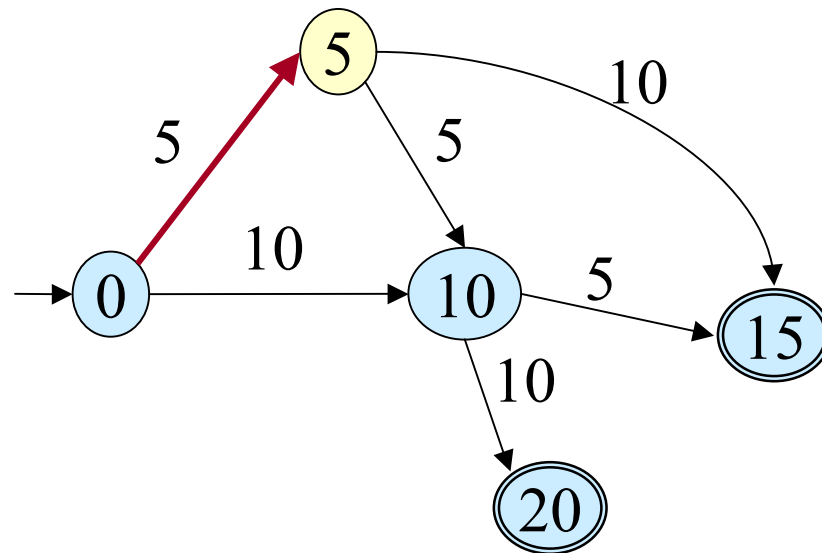


Máquina de refrigerante:  
- refrigerante custa 15 centavos  
- máquina aceita moedas de 5 e 10 centavos





## Exemplo de funcionamento



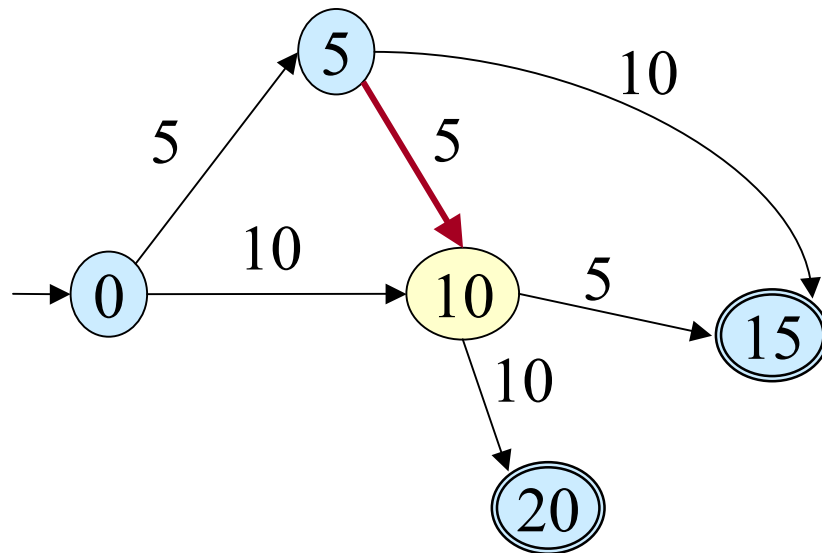
Máquina de refrigerante:  
-refrigerante custa 15 centavos  
- máquina aceita moedas de 5 e 10 centavos



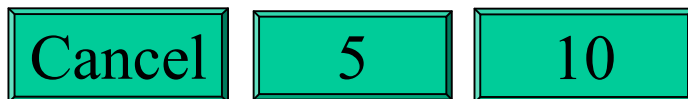




## Exemplo de funcionamento

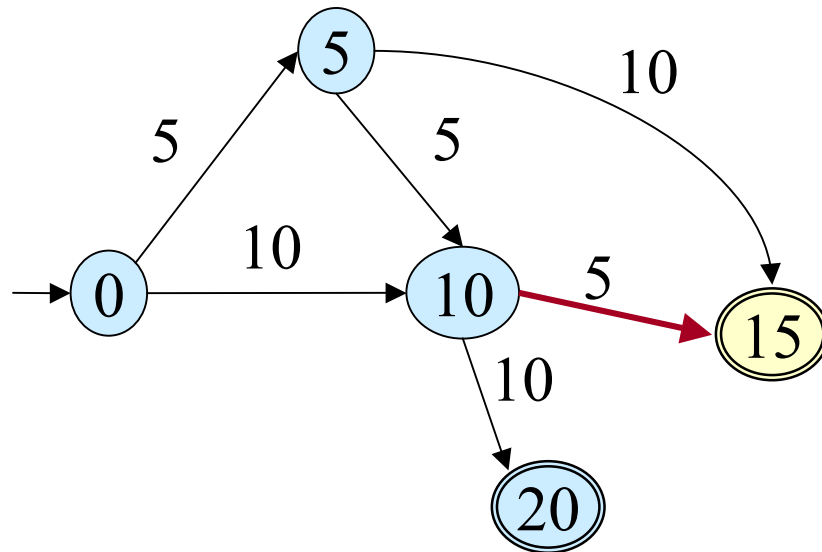


Máquina de refrigerante:  
-refrigerante custa 15 centavos  
- máquina aceita moedas de 5 e 10 centavos





## Exemplo de funcionamento

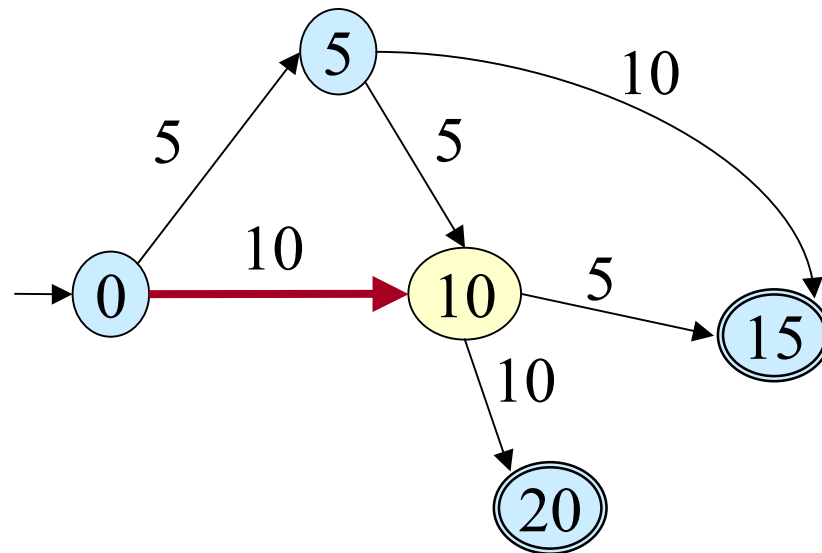


Máquina de refrigerante:  
- refrigerante custa 15 centavos  
- máquina aceita moedas de 5 e 10 centavos





## Exemplo de funcionamento

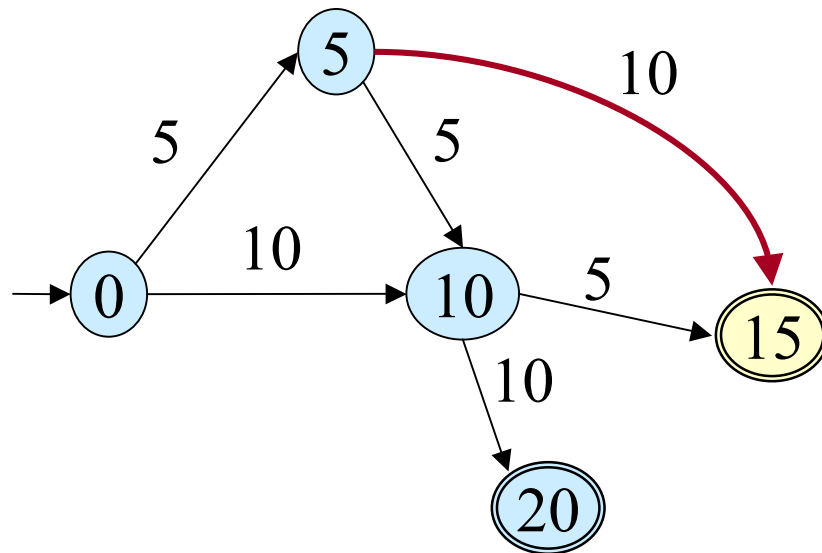


Máquina de refrigerante:  
- refrigerante custa 15 centavos  
- máquina aceita moedas de 5 e 10 centavos

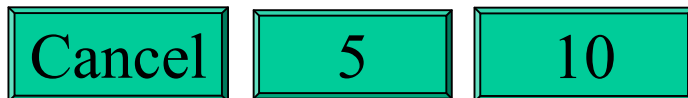




## Exemplo de funcionamento

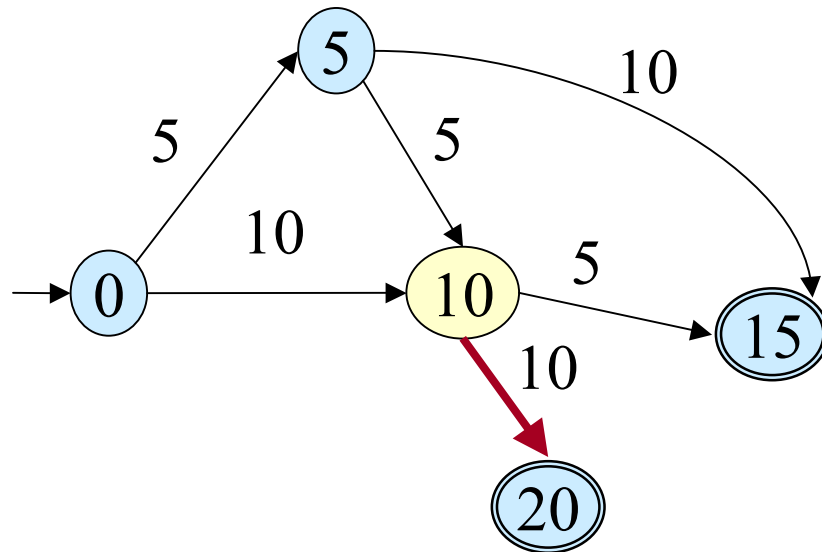


Máquina de refrigerante:  
- refrigerante custa 15 centavos  
- máquina aceita moedas de 5 e 10 centavos

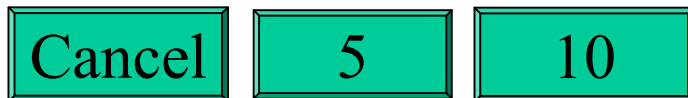




## Exemplo de funcionamento

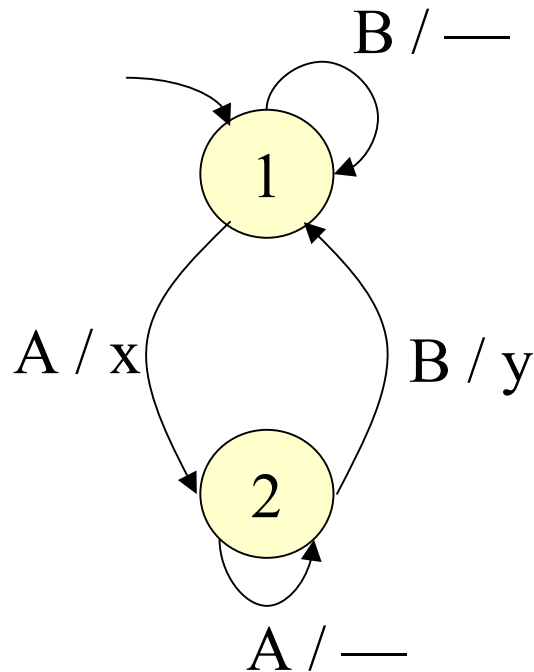


Máquina de refrigerante:  
- refrigerante custa 15 centavos  
- máquina aceita moedas de 5 e 10 centavos



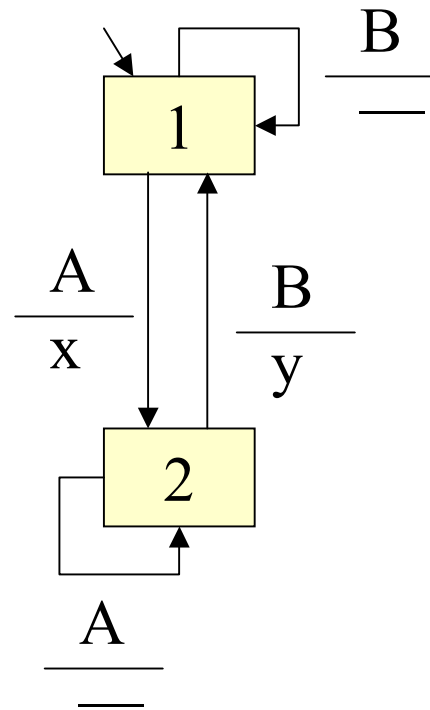


# Notação 1: diagrama de estado

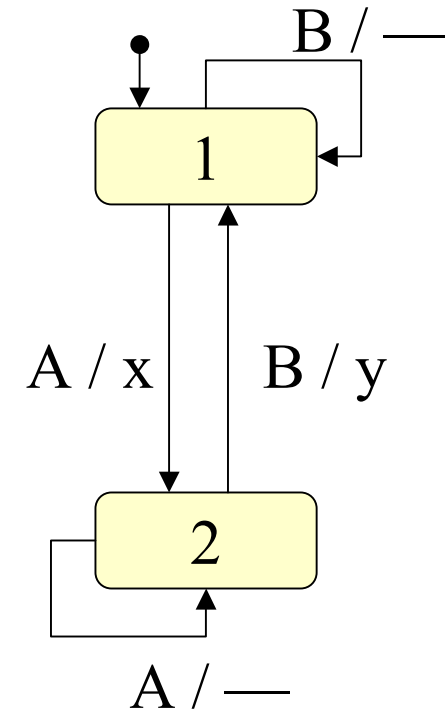


**clássico**

[Binder 2000]



**Análise  
Estruturada**



**UML**



## Notação 2: tabela de transições

Estados	Eventos	
	A	B
1	2/x	1/–
2	2/–	1/y

estado  
origem

estado  
destino

ação



# Tópicos

- Modelo de estados: apresentação
- Características
- **Propriedades**
- Testes de transição de estados

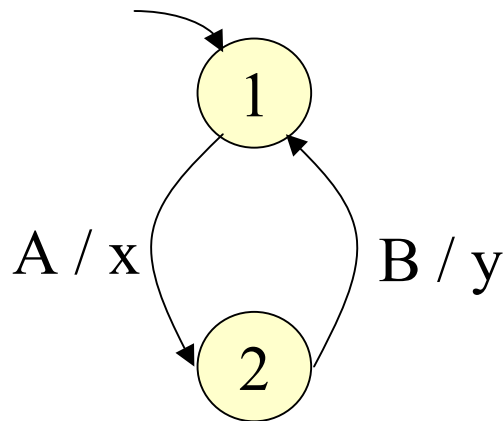




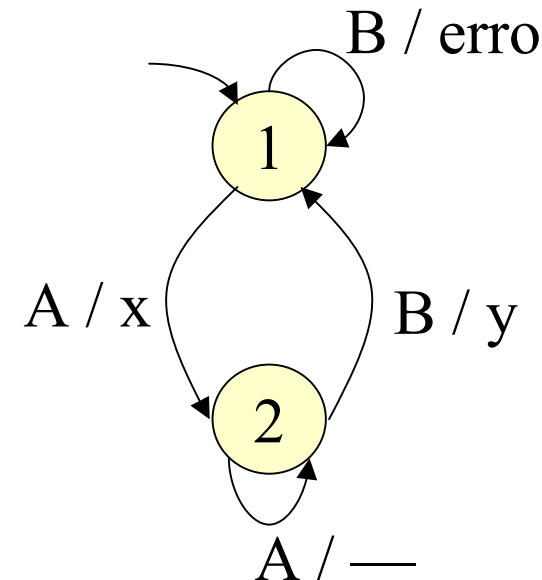
# Propriedades

- **Completa**: quando a cada par evento-estado está associada uma transição.

Caso contrário a MFE é dita **incompleta** ou **parcialmente especificada**.



**incompleta**



**completa**

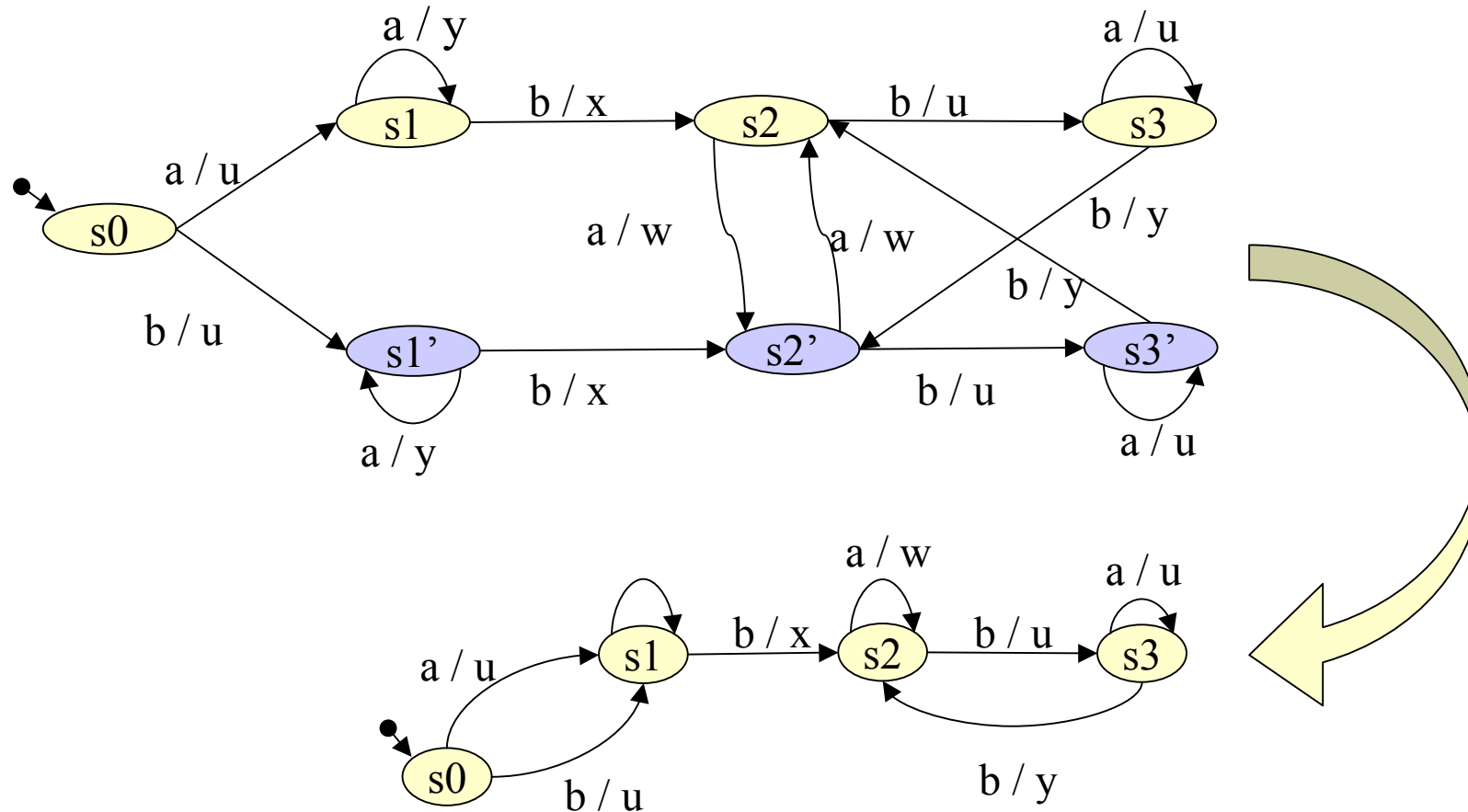


# Propriedades

- **Mínima**: quando não possui **estados equivalentes**
  - Dois estados **s** e **s'** são **equivalentes** se toda sequência de entrada começando em **s** produz exatamente as mesmas saídas quando começam em **s'**
  - Existem algoritmos para detectar se uma MFE é mínima.
  - Existem algoritmos que minimizam uma MFE.



## Estados equivalentes - exemplo



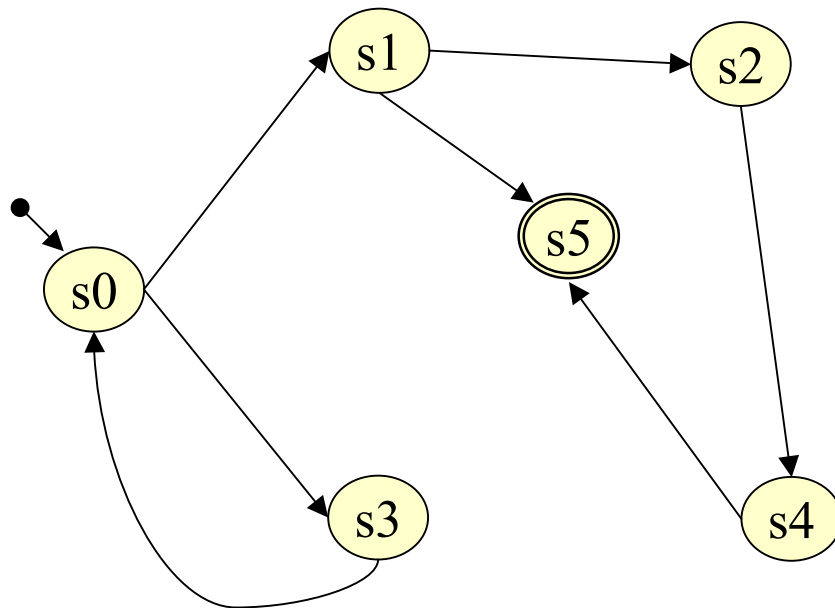


# Propriedades

- **Fortemente conexa:** todo estado é **alcançável** a partir de um outro estado
  - Um estado  $s'$  é alcançável a partir de um estado  $s$  se existe um caminho de  $s$  a  $s'$ .
  - Em outros termos: se existe uma seqüência válida de eventos que leve a MFE de  $s$  a  $s'$ .
  - A MFE é dita **inicialmente conexa** se todo estado  $s$  é alcançável a partir do estado inicial  $s_0$ .



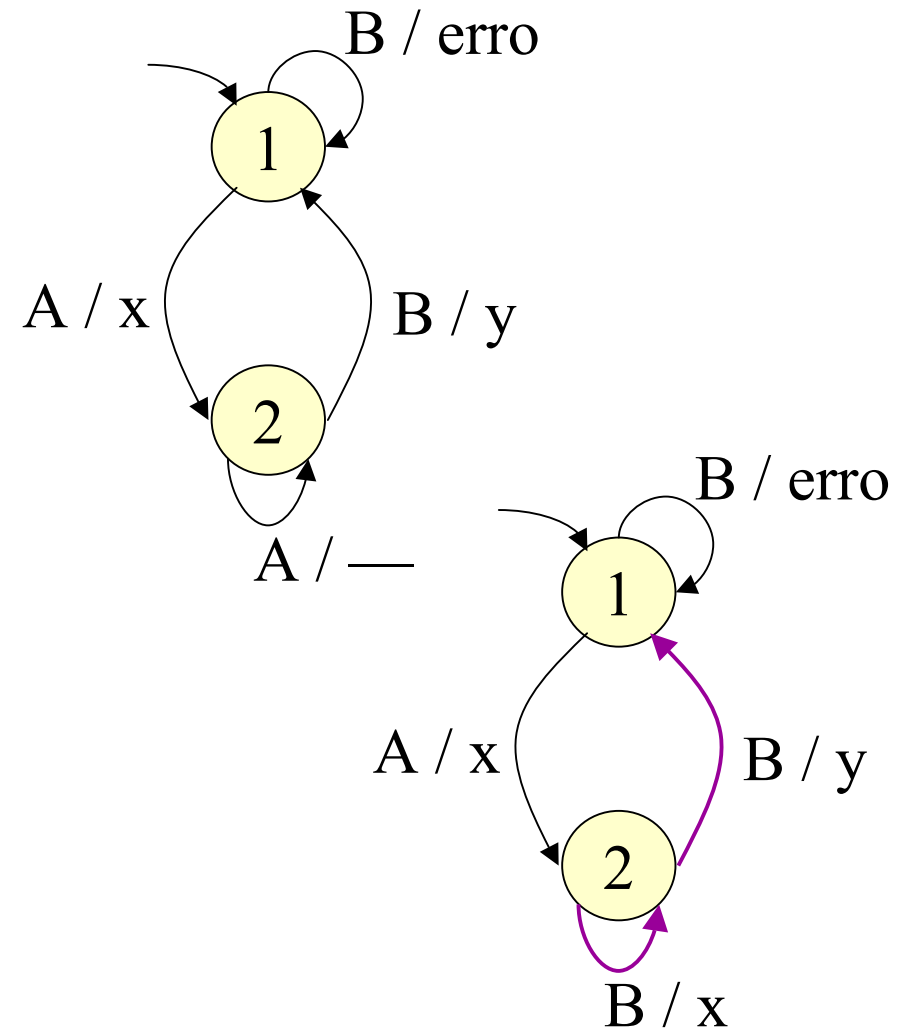
## Conectividade - exemplo





# Propriedades

- **Determinismo**: para cada estado, existe uma e somente uma transição para cada entrada aceita no estado. Nesse caso diz-se que a MFE é **determinista**. Caso contrário, a MFE é **indeterminista**, ou seja, pode ter mais de uma transição para um dado estado e uma dada entrada.
- Existem algoritmos que transformam máquinas indeterministas em deterministas, mas com restrições (domínio de valores de dados, etc).





# Propriedades

Uma MFE testável deve ser:

- **Completa:** a cada par evento-estado deve ser associada uma transição (alguns métodos aceitam MFE incompleta)
- **Mínima:** não possui estados equivalentes
- **Inicialmente conexa:** todo estado é alcançável a partir do estado inicial
- **Determinista:** a cada par evento-estado está associada uma e somente uma transição



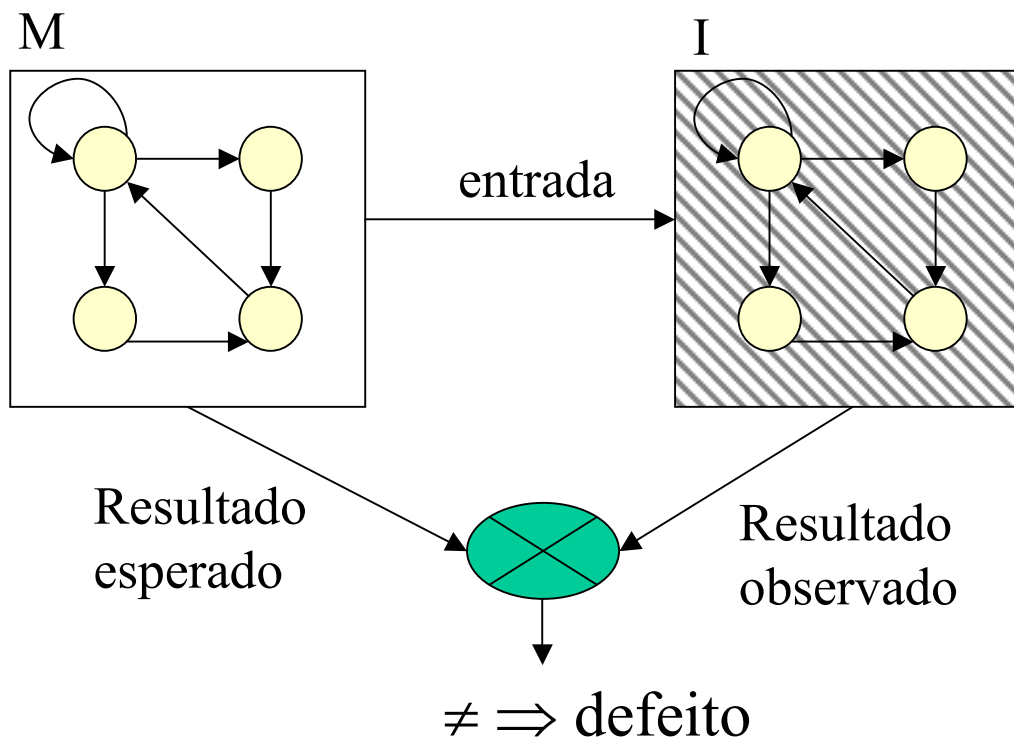
# Tópicos

- Modelo de estados: apresentação
- Características
- Propriedades
- Testes de transição de estados





# Testes de transições de estado: princípio





# Passos para os testes de transição de estados

- ① Construir o modelo de estados
- ② Revisar o modelo construído  $\Rightarrow$  possui propriedades desejadas?
- ③ Gerar os casos de testes obtendo as entradas e saídas esperadas
- ④ Executar os testes gerados
- ⑤ Analisar os resultados:
  - saída observada = saída esperada ?
  - estado final correto ?

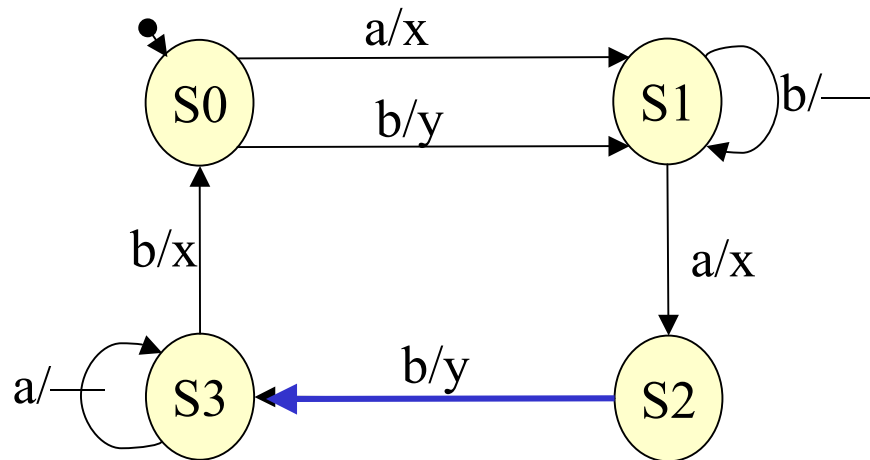



## Como chegar a um estado específico

- Usar uma **seqüência de sincronização**, que leva a MFE de um estado  $s$  a um estado  $s'$ 
  - ☹ Nem sempre existe essa seqüência
- Usar uma **transição de reset**, que leva a MFE de um estado  $s$  ao estado inicial  $s_0$  ...
  - ☹ Precisa ser especificada e implementada
- ... e usar seqüência de entradas que leve de  $s_0$  a  $s'$ 
  - ☺ Sempre existe pois a MFE deve ser determinista e inicialmente conexa



## Princípio - Exemplo



- Para chegar a s2, aplique a **seqüência de sincronização:**  
**a / x – a / x**
- Aplique a entrada **b**
- Verifique se a saída é **y**
- Verifique se a máquina está no estado **s3** 



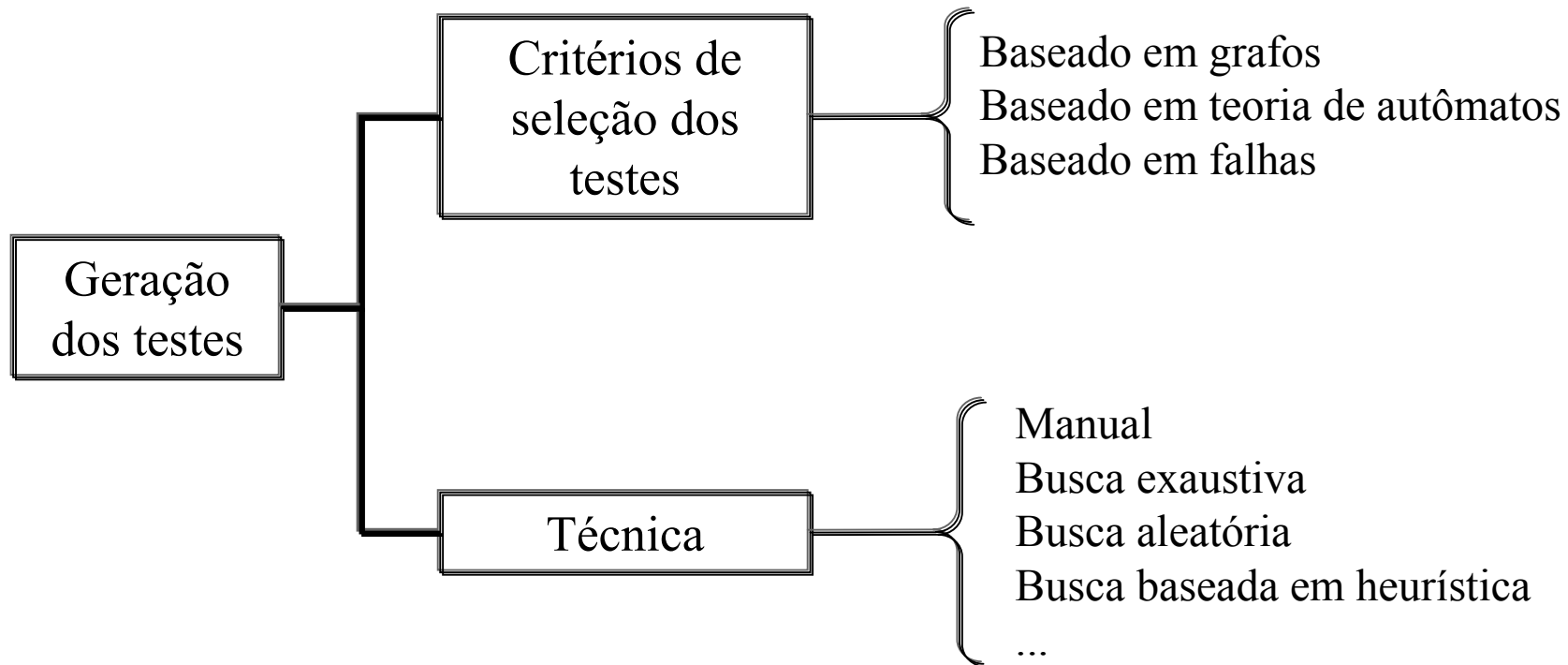
## Quais falhas se pode revelar?

- **Omissão de transição:** a implementação não responde a um par evento/estado especificado
- **Transição incorreta:** o estado resultante é válido, mas incorreto (falha de transferência)
- **Omissão de ação:** a implementação não responde a um evento válido
- **Ação incorreta:** a implementação executa a ação errada em resposta a um evento
- **Caminho furtivo:** a implementação aceita um evento não especificado para o estado (suposição de completeza incorreta)
- **Corrupção de estado:** a implementação faz transição para um estado inválido (não especificado)
- **Omissão/acréscimo de estado:** a implementação apresenta comportamento imprevisível
- **Alçapão:** a implementação aceita evento não especificado

[Binder00, 7.4]



# Taxonomia dos testes



Utting, A.Pretschner, B.Legeard. A Taxonomy of Model-Based Testing. Working Paper Series. Univ. of Waikato, Apr/2006.



## Testes baseados em cobertura

- Exercitar caminhos no modelo  
(análogo aos testes de caminhos caixa branca)
- Caminho = seqüência de entradas + saídas esperadas
- Como nos testes caixa branca: n° de caminhos pode ser muito alto ...
- O quê fazer nesse caso?





## Geração de testes - critérios

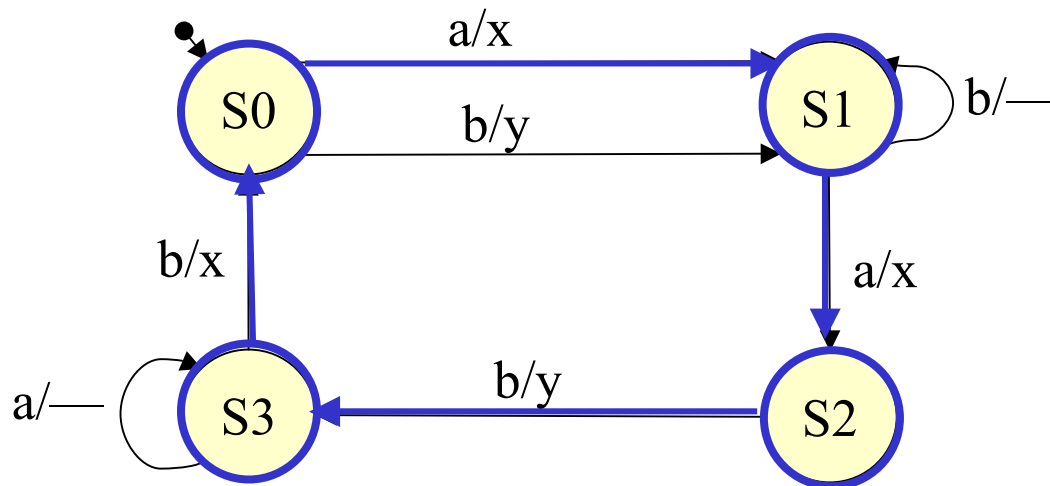
- Diferentes critérios podem ser aplicados, como nos testes caixa branca:
  - Cobertura de estados do modelo
  - Cobertura de transições do modelo
  - Cobertura de seqüências de transições (“Switch cover”)
  - Cobertura borda-interior (boundary-interior)
  - ...





## Cobertura de estados

Critério: visitar cada estado do modelo pelo menos uma vez

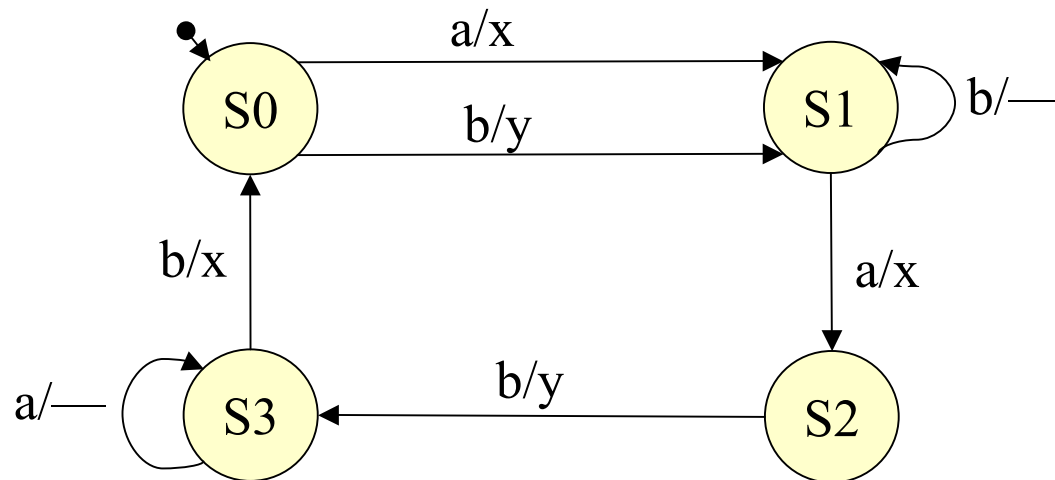


**Caso de teste:  $a / x - a / x - b / y - b / x$**



## Cobertura de transições ( método T )

Critério: visitar cada transição do modelo pelo menos uma vez



Estado	S0	S1	S1	S2	S3	S3	S0	S1	S2	S3	S0
Entrada	a	b	a	b	a	b	b	b	a	b	b
Saída esperada	x	—	x	y	—	x	y	x	y	y	x



## Cobertura de transições

- Algoritmos do tipo Carteiro Chinês são utilizados para percorrer as arestas do grafo
- Para evitar caminhos muito longos pode-se ter um caso de teste para cada transição
- Passos:
  - Aplicar seqüência de entradas que levem a  $s_2$
  - Aplicar entrada  $b$
  - Verificar se saída obtida é  $y$



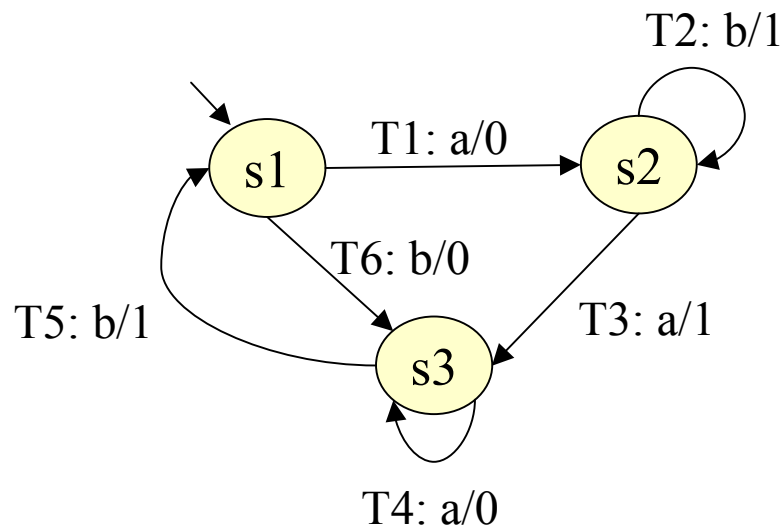
## Switch-cover (SC)

- Switch-cover (SC) = combinações de arestas (transições)
  - Ex.: 1-SC = combinações de pares de transições
  -
- Em Teoria de Grafos: algoritmo de de Bruijn para gerar combinações de seqüências de arestas



## Switch-cover (SC)

**Critério1-SC:** cada par de transições adjacentes do modelo deve ser coberto pelo menos uma vez

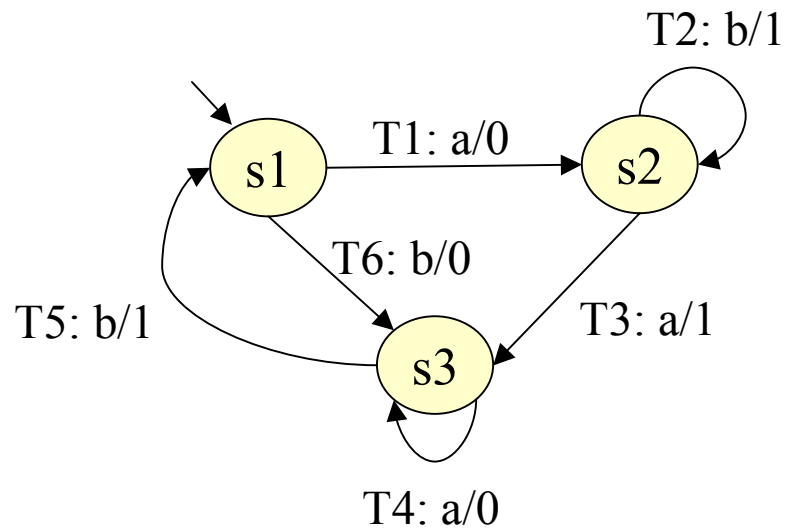


Requisitos de teste para o modelo:

(T1, T2), (T1, T3), (T2, T2),  
(T2, T3), (T3, T4), (T3, T5),  
(T4, T4), (T4, T5), (T5, T6),  
(T5, T1), (T6, T4), (T6, T5)



## Switch-cover (1-SC) - testes

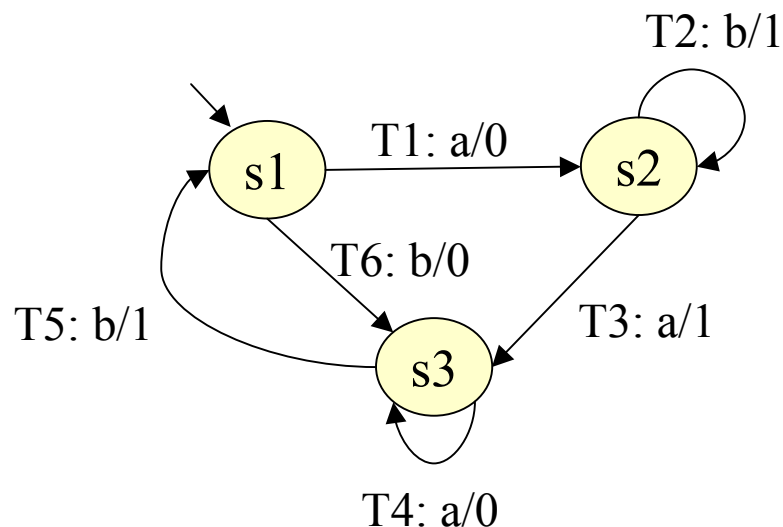


Seq. de testes	Saídas esperadas	Pares cobertos
abbaaab	0111001	(T1, T2), (T2, T2), (T2, T3), (T3, T4), (T4, T4), (T4, T5)
aaba	0110	(T1, T3), (T3, T5), (T5, T1)
aabb	0110	(T1, T3), (T3, T5), (T5, T6)
baab	0001	(T6, T4), (T4, T4), (T4, T5), (T6, T5)
bb	01	(T6, T5)



## Cobertura borda-interior

- Um conjunto de testes é adequado para este critério se os laços do modelo são visitados 0 (passa pela borda, i.e, pula o laço) e pelo menos 1 vez (passa pelo interior do laço)



Seq. de testes	Saídas esperadas	Transições cobertas
aab	011	(T1, T3), (T3, T5)
abaabb	00001	(T1, T2), (T3, T4), (T5, T6)



## Percurso aleatório (Random Walk)

- Princípio:
    - A partir do estado corrente, escolher aleatoriamente uma transição
    - Repetir o mesmo processo para o próximo estado
- |  |   |
|--|---|
|  | <ul style="list-style-type: none"><li>☺ Fácil de implementar</li><li>☺ Bom para testar cenários inesperados ou pouco comuns</li><li>☹ Demora a cobrir modelos grandes</li></ul> |
|--|---|





## Percurso aleatório guiado

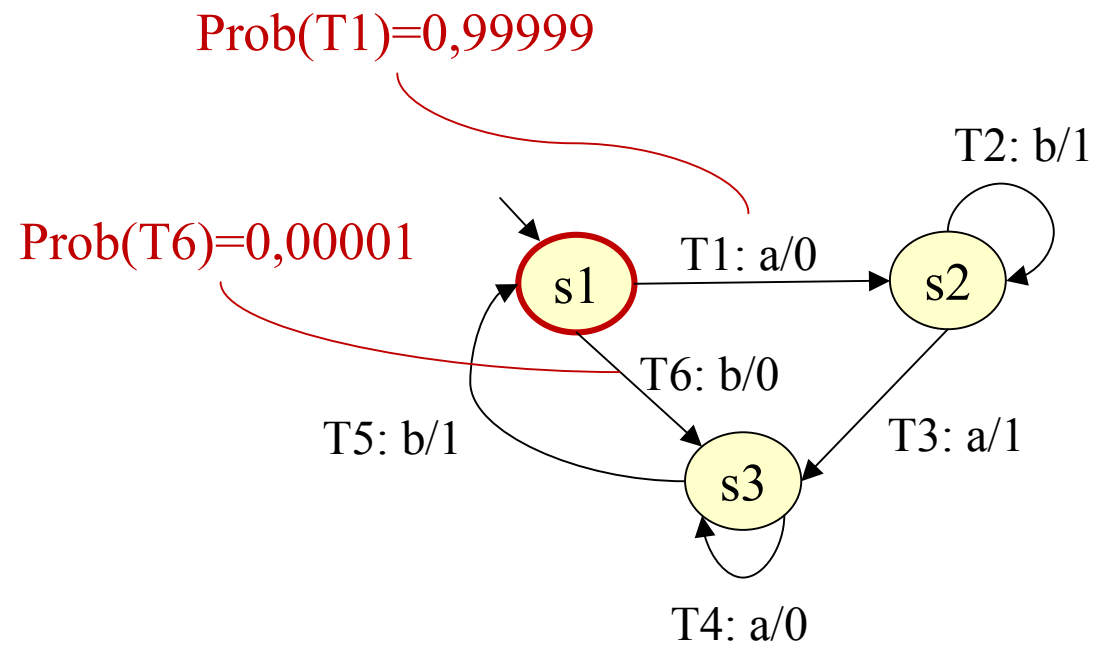
- Guia o percurso de forma a cobrir áreas de interesse no modelo
- Modelo com probabilidades associadas às transições:
  - Ex.: distribuição uniforme

$$\text{Prob}(s_i, s_j) = \begin{cases} 1/g(s_i) & \text{se } (s_i, s_j) \in T \\ 0 & \text{senão} \end{cases} \quad g(s_i) - \text{grau de saída de } s_i$$

- Transições com maior probabilidade têm mais chance de serem visitadas
- ☹ Cobertura de transições com baixa probabilidade vai requerer grande número de testes



# Exemplo





## Testes baseados em teoria de autômatos

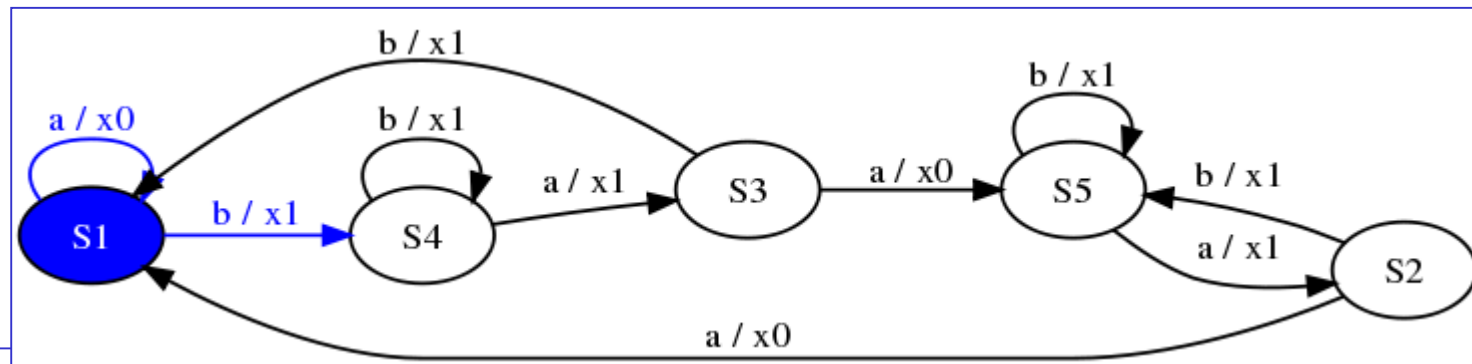
- Testes formais, visam mostrar a **equivalência** entre I (implementação) e M (modelo)
  - Equivalência de entrada e saída: I, quando submetida a entradas produzidas a partir de M, produz as mesmas saídas especificadas em M?
- Baseiam-se em seqüências de verificação de estado



## Verificação do estado

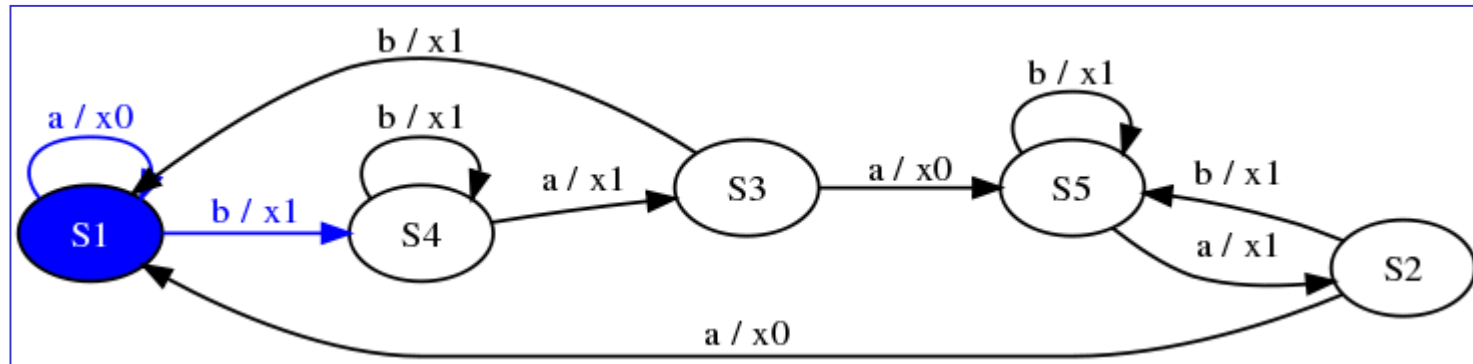
- Consiste em aplicar seqüências de entradas ao estado corrente da MFE de forma que, a partir das saídas obtidas se possa identificar o estado origem, isto é, o estado em que a máquina estava quando a seqüência foi aplicada.
- Exemplo de modelo:

Gerado com a Plavis: <http://www2.dem.inpe.br/plavisFSM/main.php>





# Tabela de transições



Estado atual	Saída		Próximo estado	
	a	b	a	b
s1	x0	x1	s1	s4
s2	x0	x1	s1	s5
s3	x0	x1	s5	s1
s4	x1	x1	s3	s4
s5	x1	x1	s2	s5

Inspirado em [Mathur2008, pg214]

estes baseados em modelos de estado



## Elementos de uma MFE - Formalização

- Uma MFE pode ser definida como uma tupla:  
 $(X, Y, S, s_0, \delta, O)$ 
  - $X$ : conjunto finito de entradas (alfabeto de entrada)
  - $Y$ : conjunto finito de saídas (alfabeto de saída)
  - $S$ : conjunto finito de estados
  - $s_0$ : estado inicial
  - $\delta : X \times S \rightarrow S$  - função de transição -- no máximo um próximo estado (modelo determinista)
  - $O : X \times S \rightarrow Y$  - função de saída
  - $F \subset S$ : conjunto de estados finais (opcional)



## Verificação do estado

- Consiste em aplicar seqüências de entradas ao estado corrente da MFE de forma que, a partir das saídas obtidas se possa identificar o estado origem, isto é, o estado em que a máquina estava quando a seqüência foi aplicada.
- Existem 3 métodos principais (e diversas variantes):
  - **UIO**: obtém seqüências únicas de entrada e saída por estado
  - **DS**: obtém uma seqüência de distinção para a MFE que permite distinguir um estado do outro
  - **W**: obtém um conjunto de seqüências de caracterização



## Método W

- Conjunto de caracterização (W)
  - Contém seqüências que podem distinguir cada par de estados
$$\forall s, s' \in S, s \neq s', \exists x \in W: O(s, x) \neq O(s', x) \text{ para } x \in X^+$$
  - W está associado à máquina
  - A máquina precisa ser:
    - Completa
    - Fortemente conexa
    - Mínima
  - W sempre existe se a máquina é mínima



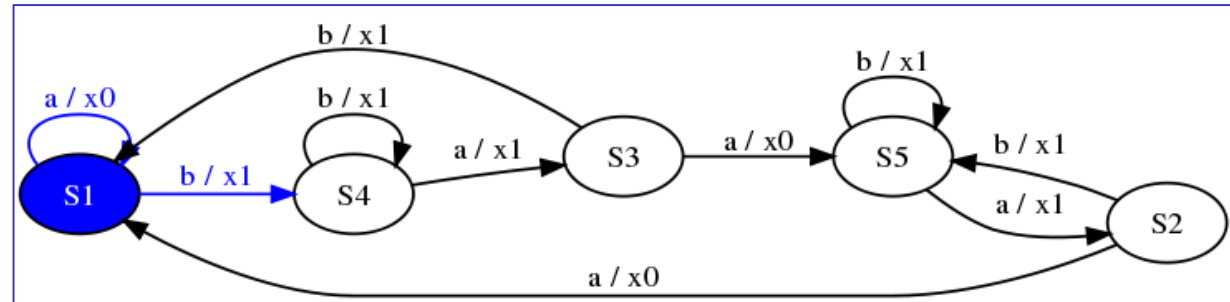


## Método W – Geração de testes

- Geração de testes:
  - Estimar se M e I têm aproximadamente o mesmo nro. de estados
    - Assume-se que nro estados (I)  $\approx$  nro estados (M)
  - Obter W
  - Obter a árvore de transições para dela extrair o conjunto de cobertura de transições, P
  - Construir o conjunto Z
  - O conjunto de testes é dado por:  $P \bullet Z$



# O conjunto W



$S_i$	$S_j$	$x$	$O(S_i, x)$	$O(S_j, x)$
S1	S2	<b>baaa</b>	x1x1x0x1	x1x1x0x0
S1	S3	<b>aa</b>	x0x0	x0x1
S1	S4	<b>a</b>	x0	x1
S1	S5	a	x0	x1
S2	S3	aa	x0x0	x0x1
S2	S4	a	x0	x1
S2	S5	a	x0	x1
S3	S4	a	x0	x1
S3	S5	a	x0	x1
S4	S5	<b>aaa</b>	x1x0x1	x1x0x0

tado



# Árvore de Alcançabilidade

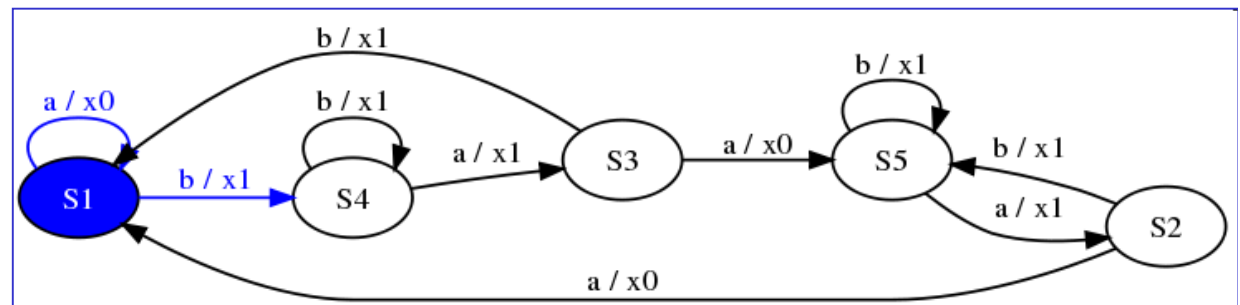
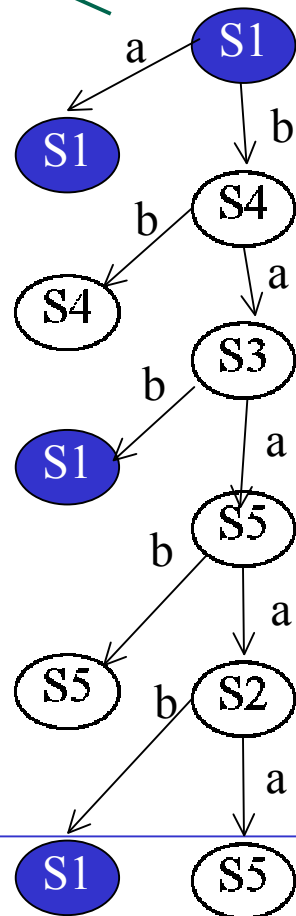
- Para obter P, é preciso construir a árvore de alcançabilidade, da seguinte forma:
  - ① O estado inicial é a raiz da árvore.
  - ② Examine cada estado não-terminal e cada transição que sai desse estado. No mínimo uma nova aresta é criada para cada transição. Essa aresta liga o estado atual ao próximo estado
  - ③ Para cada nova aresta e cada novo nó adicionado no passo 2:
    - se o novo nó corresponde a um estado já representado por um outro nó da árvore ou a um estado final, marque esse nó como terminal (nenhuma aresta será criada a partir desse nó)
  - ④ Repita os passos ② e ③ até que todos os nós tenham sido marcados como terminais.

[Binder00, 7.4]



# Conjunto de cobertura de transições

Árvore de alcançabilidade

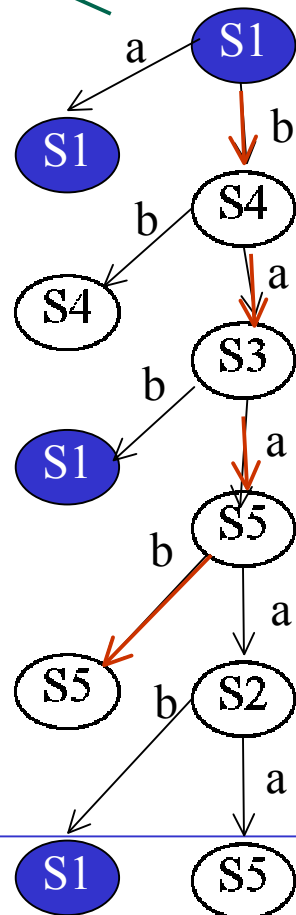


Testes baseados em modelos de estado



# Conjunto de cobertura de transições

Árvore de alcançabilidade



$P = \{\epsilon, a, b, bb, ba, bab, baa, baab, baaa, baaab, baaaa\}$

- Conjunto (de caminhos parciais) que cobre todas as transições.
- Cada caminho começa no estado inicial.
- $\epsilon$  é a seq nula.



## Conjunto Z

- O conjunto Z é obtido como:
  - $Z = (X^0 \bullet W) \cup (X^1 \bullet W) \dots \cup (X^{m-n} \bullet W)$ 
    - m: n° de estados de I
    - n: n° de estados de M
    - $m > n$
    - $X^0 = \{\epsilon\}$ ,  $X^1 = X$ ,  $X^2 = X \bullet X$ , ...  
em que  $\bullet$  indica concatenação



## Conjunto de testes

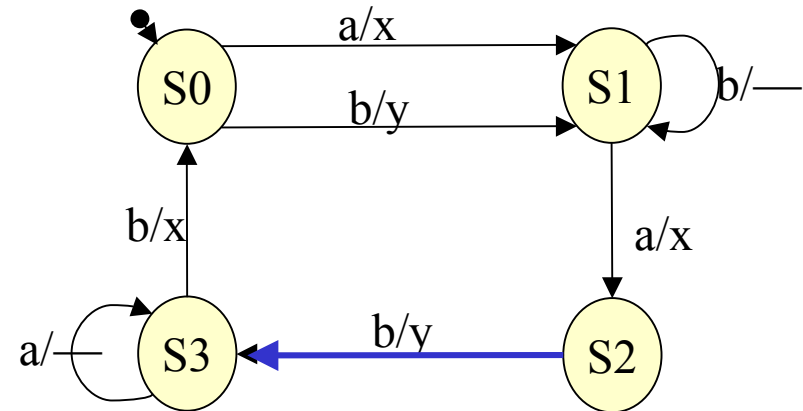
- No exemplo, assumindo que  $m = n$  tem-se:
  - $Z = X^0 \bullet W = \{a, aa, aaa, baaa\}$
- Concatenando-se  $P$  e  $Z$ :
  - $T = \{\epsilon, a, b, bb, ba, bab, baa, baab, baaa, baaab, baaaa\}$ 
    - $\{a, aa, aaa, baaa\}$
  - $\{a, aa, aaa, baaa,$   
 $aa, aaa, aaaa, abaaa,$   
 $ba, baa, baaa, bbaaa,$   
 $bba, bbaa, bbaaa, bbbaaa,$   
 $\dots\}$

**Plavis: 66 casos de teste**



# Seqüência Única de Entrada e Saída

- Método U ou UIO (do inglês Unique I/O Sequences)
  - Seqüência  $x$  que distingue cada estado  $s$  dos demais:  
 $\forall s' \in S, s \neq s', O(s, x) \neq O(s', x)$
  - Cada estado tem sua UIO
  - A máquina precisa ser:
    - Fortemente conexa
    - Mínima
  - UIOs podem não existir



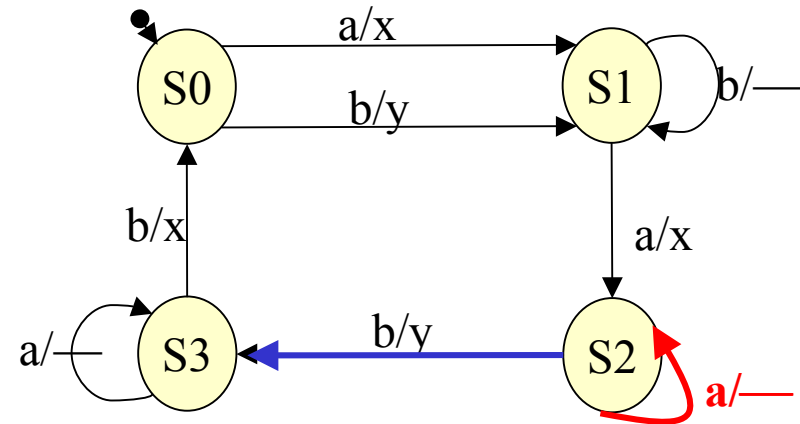
Estado	UIO
S0	ab
S1	b
S2	?
S3	a





# Seqüência Única de Entrada e Saída

- Método U ou UIO (do inglês Unique I/O Sequences)
  - Seqüência  $x$  que distingue cada estado  $s$  dos demais:  
 $\forall s' \in S, s \neq s', O(s, x) \neq O(s', x)$
  - Cada estado tem sua UIO
  - A máquina precisa ser:
    - Fortemente conexa
    - Mínima
  - UIOs podem não existir

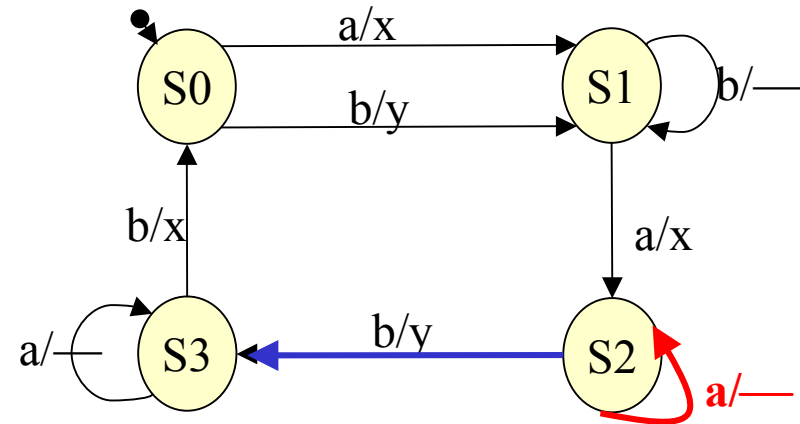


Estado	UIO
S0	ab
S1	b
S2	<b>ab</b>
S3	<b>ba</b>



## Seqüência de Distinção

- Método D ou DS (do inglês Distinguishing Sequence)
  - Seqüência  $x$  que produz uma saída diferente para cada estado  
 $\forall s, s' \in S, s \neq s', O(s, x) \neq O(s', x)$
  - A DS está associada à máquina
  - A máquina precisa ser:
    - Completa
    - Fortemente conexa
    - Mínima
  - A DS pode não existir



DS =ab	Estado	saida
	S0	x —
	S1	x y
	S2	— y
	S3	— x



# Sumário

- Testes caixa preta são baseados na especificação do sistema, ignorando seu código fonte
- Partições de equivalência são úteis para testar entradas isoladamente
- Análise de valores-limite testa entradas isoladamente, nos limites, sendo portanto indispensáveis
- Grafo causa-efeito/tabela de decisões são úteis para testar combinações de entradas
- Máquinas de estado são úteis para testar seqüências válidas de entradas. Devem ser verificadas.
- Abordagem recomendada: combinar técnicas.



## Exercícios

- Gere casos de testes para os problemas a seguir, considerando os critérios:
  - Cobertura de estados
  - Cobertura de transições



# Exercício1 – Controle de microondas

Considere o sistema de controle de um microondas, com botões para: escolha da potência, ajuste do relógio e iniciar. O funcionamento do forno se dá da seguinte forma:

1. Selecione o nível de potência (máxima ou média)
2. Informe o tempo de cozimento
3. Pressione Iniciar e o alimento será cozido durante o tempo especificado

Por razões de segurança, o microondas não pode operar enquanto a porta estiver aberta.

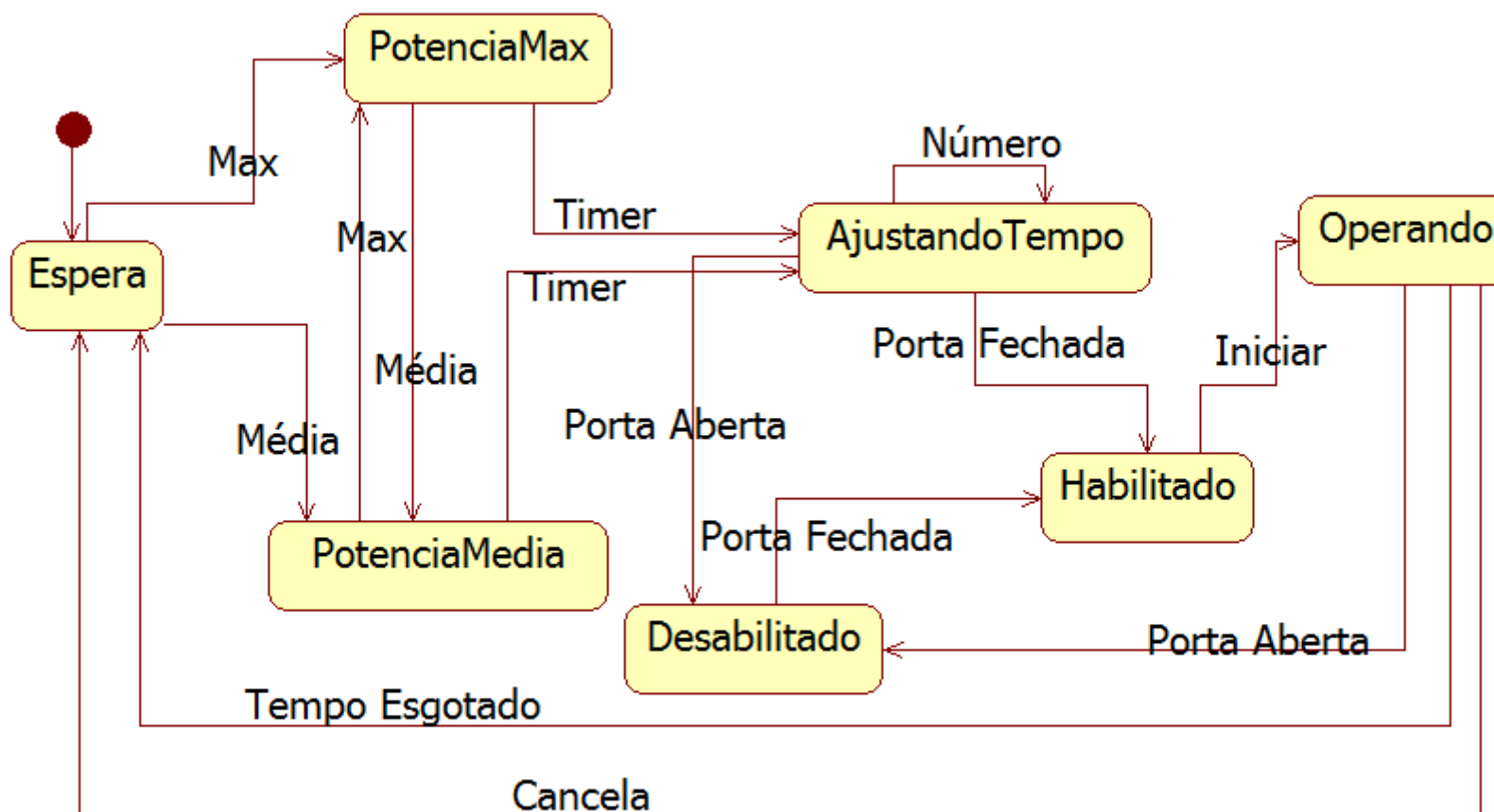
Ao completar o cozimento, um alarme é acionado.

O microondas possui também um pequeno visor alfanumérico que permite exibir várias mensagens de avisos e de alerta.

Baseado em [Sommerville 2003, cap 7]



# Diagrama de estados





## Exercício 2

- Forneça a especificação na forma de modelo de estados para um sistema de controle de aquecimento de uma casa, composto de um aquecedor, um termostato e um ventilador. O controle da temperatura é distribuído, ie, cada cômodo tem um controlador de temperatura. Se a temperatura em um cômodo cai abaixo de  $t_a - 2$ , onde  $t_a$  é a temperatura ambiente desejada, o aquecedor é ligado. Quando a temperatura do aquecedor atinge um limite máximo  $T$ , este é desligado e o ventilador é acionado para espalhar o ar quente. Enquanto isso o termostato monitora e registra a temperatura ambiente. Quando esta atinge  $t_a + 2$ , o aquecedor é desligado. O ventilador continua ligado até que a temperatura do aquecedor chegue a  $T - 5$ . Assuma que  $t_a + 2 > T - 5$ .

[Alagar e Periyasamy 98]