Princípios de Análise e Projeto de Sistemas com UML

2ª edição

Eduardo Bezerra

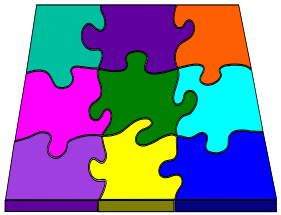
Editora Campus/Elsevier

Capítulo 9
Modelagem de estados

Todos os adultos um dia foram crianças, mas poucos se lembram disso.

--O Pequeno Príncipe, Antoine de Saint-Exupéry

Tópicos



- Introdução
- Diagrama de máquina de estados
- Já foi chamado de diagrama de transição de estados e diagrama de gráfico de estados
- Identificação dos elementos de um diagrama de estados
- Construção de diagramas de máquina de estados
- Modelagem de estados no processo de desenvolvimento

Introdução

- Objetos do mundo real se encontram em estados particulares a cada momento.
  - uma jarra está *cheia* de líquido
  - uma pessoa está *cansada*.
- Da mesma forma, cada objeto participante de um sistema de software orientado a objetos se encontra em um *estado* particular.
- Um objeto muda de estado quando acontece algum *evento* interno ou externo ao sistema.
- Durante a *transição* de um estado para outro, um objeto realiza determinadas *ações* dentro do sistema.
- Quando um objeto transita de um estado para outro, significa que o sistema no qual ele está inserido também está mudando de estado.

Diagrama de máquina de estado

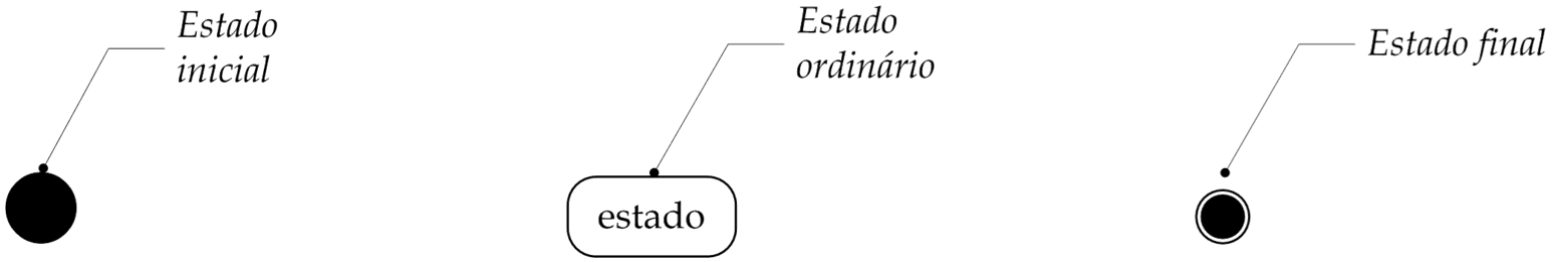
- Através da análise das *transições* entre *estados* dos objetos de um sistema de software, podem-se prever todas as possíveis *operações* realizadas, em função de *eventos* que possam ocorrer.
- O diagrama da UML que é utilizado para realizar esta análise é o ***diagrama de máquina de estado.***
- A UML tem um conjunto rico de notações para desenhar este diagrama.
  - Estados
  - *Transições*
  - *Evento*
  - *Ação*
  - *Atividade*
  - *Transições internas*
  - *Estados aninhados*
  - *Estados concorrentes*

Estado

- Situação na vida de um objeto em que ele satisfaz a alguma condição ou realiza alguma atividade. É função dos ***valores dos atributos*** e (ou) das ***ligações com outros objetos***.
  - O atributo *reservado* deste objeto livro tem valor *verdadeiro*.
  - Uma conta bancária passa para o *vermelho* quando o seu saldo fica *negativo*.
  - Um professor está *licenciado* quando não está ministrando curso algum durante o semestre.
  - Um tanque está *na reserva* quando nível de óleo está abaixo de 20%.
  - Um pedido está *atendido* quando todos os seus itens estão atendidos.
- Estados podem ser vistos como uma abstração dos atributos e associações de um objeto.

Estados inicial e final

- O estado inicial indica o estado de um objeto quando ele é criado. Só pode haver um estado inicial em um diagrama.
  - Essa restrição serve para definir a partir de que ponto um diagrama deve começar a ser lido.
- O estado final é representado como um círculo “eclipsado” e indica o fim do ciclo de vida de um objeto.
  - é opcional e pode haver mais de um estado final em um diagrama.
- Notação da UML para estados:



Transições

- Os estados estão associados a outros pelas transições.
- Uma transição é mostrada como uma linha conectando estados, com uma seta apontando para um dos estados.
- Quando uma transição entre estados ocorre, diz-se que a transição foi disparada.
- Uma transição pode ser rotulada com uma expressão da seguinte forma:

evento (lista-parâmetros) [guarda] / ação

Eventos

- Uma transição possui um evento associado.
- Um evento é algo que acontece em algum ponto no tempo e que pode modificar o estado de um objeto:
  - Pedido realizado
  - Fatura paga
  - Cheque devolvido
- Os eventos relevantes a um sistema de software podem ser classificados em nos seguintes tipos.
  1. **Evento de chamada**: recebimento de uma mensagem de outro objeto.
  2. **Evento de sinal**: recebimento de um sinal.
  3. **Evento temporal**: passagem de um intervalo de tempo predefinido.
  4. **Evento de mudança**: uma condição que se torna verdadeira.

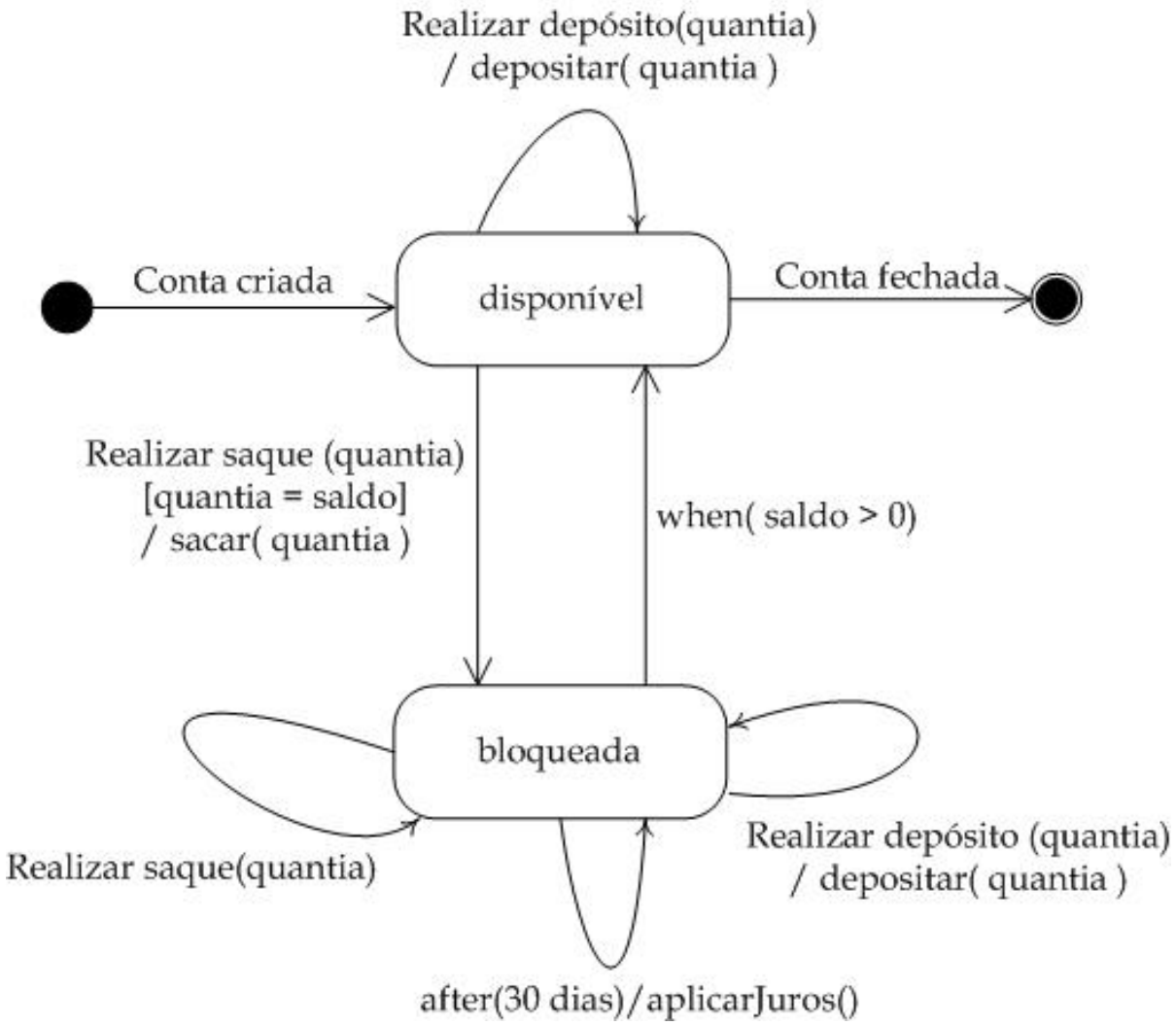
Tipos de Evento

- Evento de chamada
  - corresponde ao recebimento de uma mensagem de outro objeto.
  - Pode-se pensar neste tipo de evento como uma solicitação de serviço de um objeto a outro.
- Evento de sinal
  - Neste evento o objeto recebe um sinal de outro objeto que pode fazê-lo mudar de estado.
  - A diferença básica entre o evento de sinal e o evento de chamada é que neste último o objeto que envia a mensagem fica esperando a execução da mesma.
    - No evento de sinal, o objeto remetente continua o seu processamento após ter enviado o sinal.

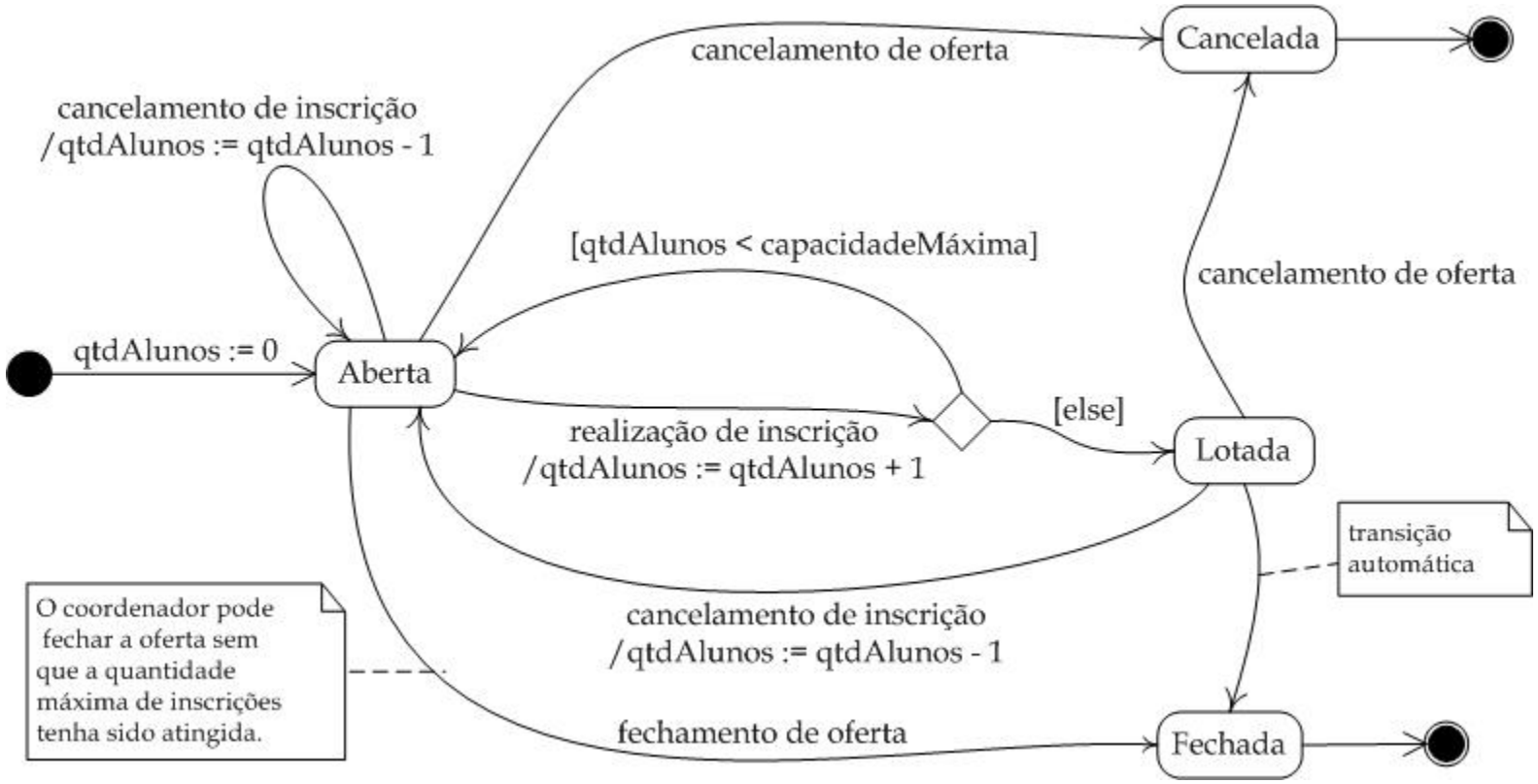
Tipos de Evento (cont.)

- Evento de temporal
  - Corresponde à passagem de um intervalo de tempo predefinido.
    - O objeto pode interpretar a passagem de um certo intervalo de tempo como sendo um evento.
  - É especificado com a cláusula **after** seguida de um parâmetro que especifica um intervalo de tempo.
    - **after(30 segundos)**: indica que a transição será disparada 30 segundos após o objeto ter entrado no estado atual.
- Evento de mudança
  - Corresponde a uma condição que se torna verdadeira.
  - É representado por uma expressão de valor lógico (verdadeiro ou falso) e é especificado utilizando-se a cláusula **when**.
    - **when(saldo > 0)**: significa que a transição é disparada quando o valor do atributo saldo for positivo.
  - Eventos temporais também podem ser definidos utilizando-se a cláusula **when**.
    - **when(data = 13/07/2002)**
    - **when(horário = 00:00h)**

Exemplo (ContaBancária)



Exemplo (OfertaDisciplina)



Condição de guarda

- É uma expressão de valor lógico que condiciona o disparo de uma transição.
- A transição correspondente é disparada se e somente se o evento associado ocorre *e* a condição de guarda é verdadeira.
  - Uma transição que não possui condição de guarda é sempre disparada quando o evento ocorre.
- A condição de guarda pode ser definida utilizando-se parâmetros passados no evento e também atributos e referências a ligações da classe em questão.

Ações

- Ao transitar de um estado para outro, um objeto pode realizar uma ou mais **ações**.
- Uma ação é uma expressão definida em termo dos atributos, operações, associações da classe ou dos parâmetros do evento também podem ser utilizados.
- A ação associada a uma transição é executada se e somente se a transição for disparada.

Atividades

- Semelhantes a ações, atividades são algo que deve ser executado.
- No entanto, uma atividade pode ser *interrompida* (uma ação não pode).
  - Por exemplo, enquanto a atividade estiver em execução, pode acontecer um evento que a interrompa.
- Outra diferença: uma atividade sempre está associada a um estado (ao contrário, uma ação está associada a uma transição).

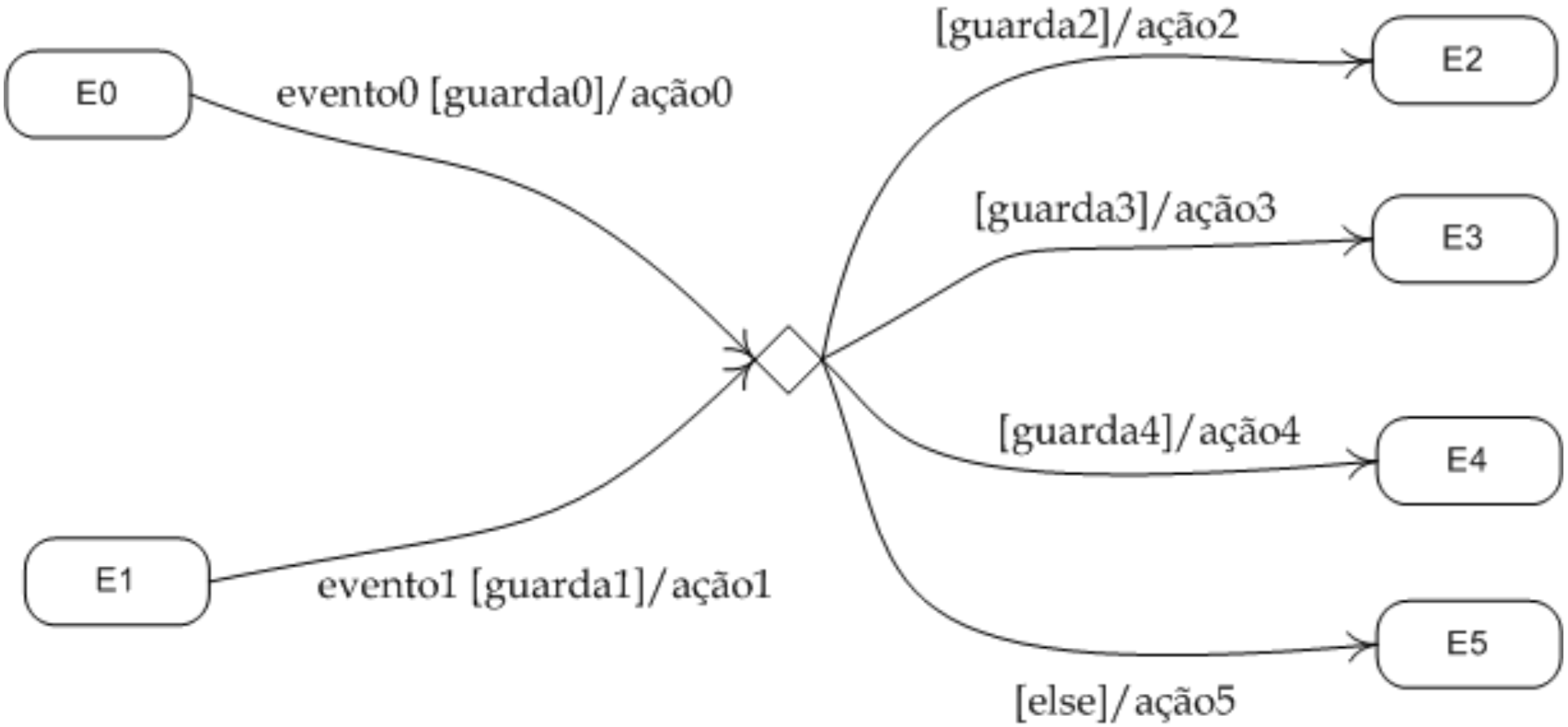
Ponto de junção

- Pode ser que o próximo estado de um objeto varie de acordo com uma condição.
  - Se o valor da condição for verdadeiro, o objeto vai para um estado E1; se o valor for falso, o objeto vai para outro estado E2.
  - É como se a transição tivesse bifurcações, e cada transição de saída da bifurcação tivesse uma condição de guarda.
- Essa situação pode ser representada em um DTE através de um ***ponto de junção***
- Pontos de junção permitem que duas ou mais transições compartilhem uma "trajetória de transições".

Ponto de junção

- De uma forma geral, pode haver um número ilimitado de transições saindo de um ponto de junção.
- Pode haver também uma transição de saída que esteja rotulada com a cláusula **else**.
  - Se as outras condições forem falsas, a transição da clausula **else** é disparada.
- Pontos de junção permitem que duas ou mais transições compartilhem uma “trajetória de transições”.
- De uma forma geral, pode haver um número ilimitado de transições saindo de um ponto de junção.
- Pode haver também uma transição de saída que esteja rotulada com a cláusula **else**.
  - Se as outras condições forem falsas, a transição da clausula **else** é disparada.

Exemplo de ponto de junção



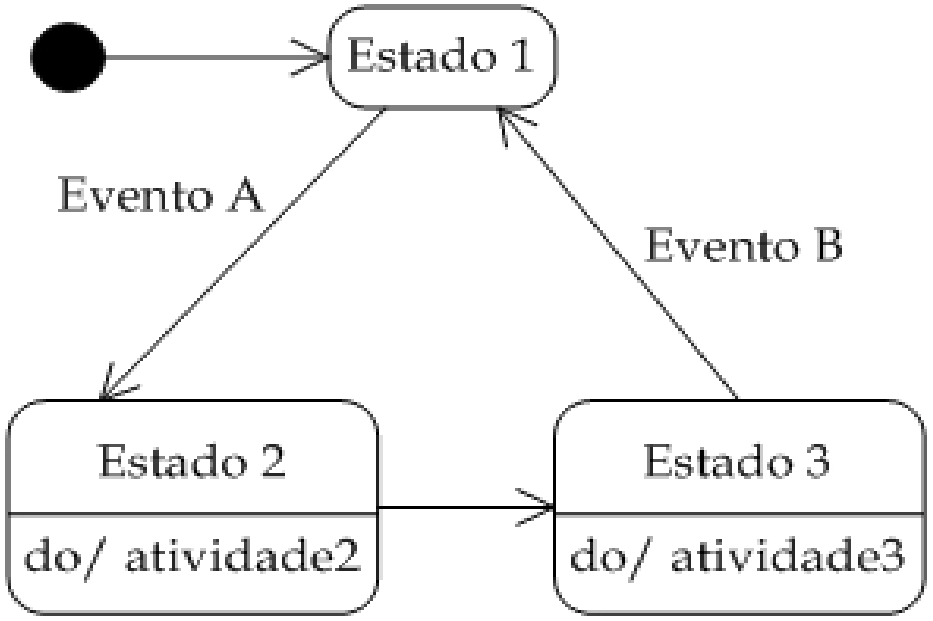
Cláusulas

- No compartimento adicional de um retângulo de estado podem-se especificar ações ou atividades a serem executadas.
- Sintaxe geral: **evento / [ação | atividade]**
- Há três cláusulas predefinidas: e*ntry,exit,do*
- Cláusula **exit**
  - Pode ser usada para especificar uma ação a ser realizada no momento em que o objeto entra em um estado.
  - A ação desta cláusula é sempre executada, independentemente do estado do qual o objeto veio.
    - É como se a ação especificada estivesse associada a todas as transições de entrada no estado.

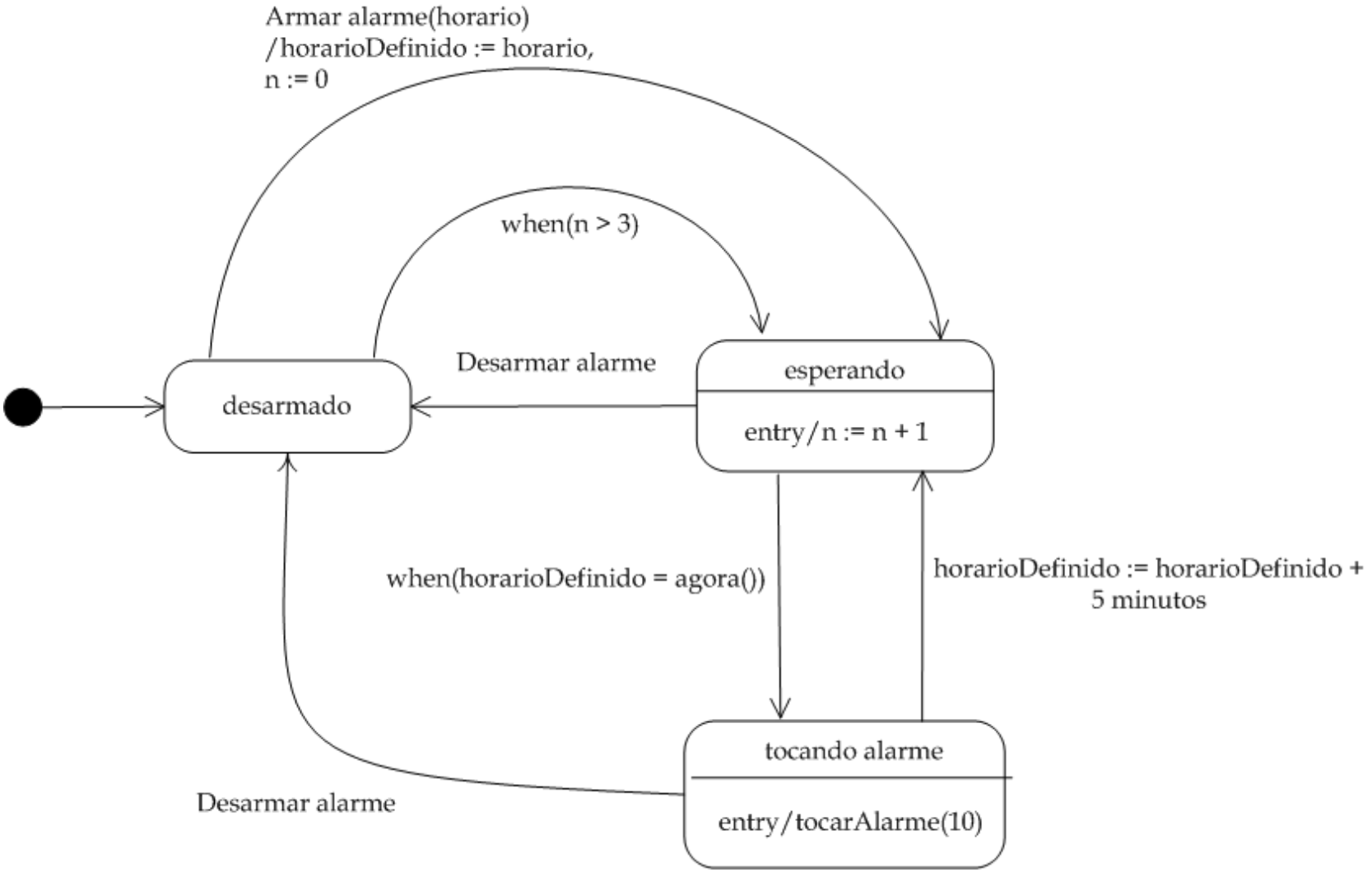
Cláusulas

- Cláusula **exit**
  - Serve para declarar ações que são executadas sempre que o objeto sai de um estado.
  - É sempre executada, independentemente do estado para o qual o objeto vai.
    - É como se a ação especificada estivesse associada a todas as transições de saída do estado.
- Cláusula **do**
  - Usada para definir alguma atividade a ser executada quando o objeto passa para um determinado estado.
  - Ao contrário da cláusula entry, serve para especificar uma atividade, em vez de uma ação.

Cláusula do - exemplo



Exemplo (Despertador)



Identificação de elementos do DTE

- Um bom ponto de partida para identificar estados é analisar os possíveis valores de seus atributos e as ligações que ele pode realizar com outros objetos.
- No entanto, a existência de atributos ou ligações não é suficiente para justificar a criação de um diagrama.
  - O comportamento de objetos dessa classe deve depender de tais atributos ou ligações.

Identificação de elementos do DTE

- Já que transições dependem de eventos para ocorrer, devem-se identificar estes eventos primeiramente.
- Além disso, deve-se examinar também se há algum fator que condicione o disparo da transição.
  - Se existir, este fator deve ser modelado como uma condição de guarda da transição.
- Um bom ponto de partida para identificar eventos é a descrição dos casos de uso.
- Os eventos encontrados na descrição dos casos de uso são externos ao sistema.
- Contudo, uma transição pode também ser disparada por um evento *interno* ao sistema.

Identificação de elementos do DTE

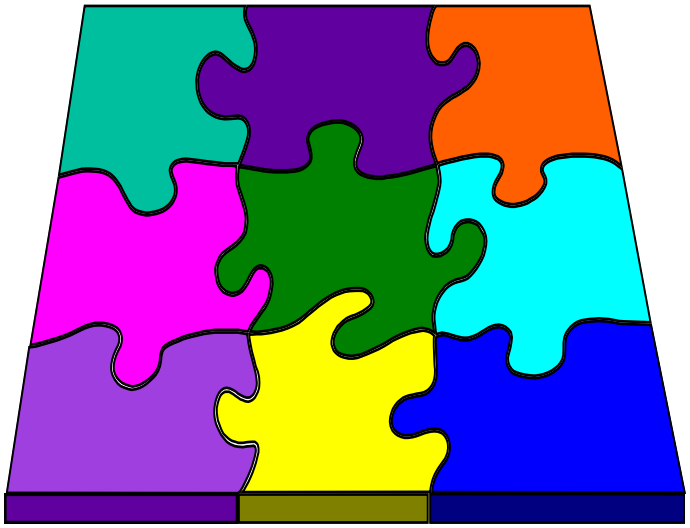
- De uma forma geral, cada operação com visibilidade pública de uma classe pode ser vista como um evento em potencial.
- Uma outra fonte para identificação de eventos associados a transições é analisar as *regras de negócio*.
  - *"Um cliente do banco não pode retirar mais de R$ 1.000 por dia de sua conta*".
  - *"Os pedidos para um cliente não especial devem ser pagos antecipadamente"*.
  - *"O número máximo de alunos por curso é igual a 30"*.

Um DME para uma classe

- Os diagramas de estados são desenhados por classe.
  - Desvantagem: dificuldade na visualização do estado do sistema como um todo.
  - Essa desvantagem é parcialmente compensada pelos diagramas de interação.
- Nem todas as classes de um sistema precisam de um DME.
  - Somente classes que exibem um comportamento dinâmico relevante.
  - Objetos cujo histórico precisa ser rastreado pelo sistema são típicos para se construir um diagrama de estados.

Procedimento para construção

1. Identifique os estados relevantes para a classe.
2. Identifique os eventos relevantes. Para cada evento, identifique qual a transição que ele ocasiona.
3. Para cada estado: identifique as transições possíveis quando um evento ocorre.
4. Para cada estado, identifique os eventos internos e ações correspondentes.
5. Para cada transição, verifique se há fatores que influenciam no seu disparo. (definição de condições de guarda e ações).
6. Para cada condição de guarda e para cada ação, identifique os atributos e ligações que estão envolvidos.
7. Defina o estado inicial e os eventuais estados finais.
8. Desenhe o DME.



Modelagem de estados no processo de desenvolvimento

Modelagem de estados no PDS

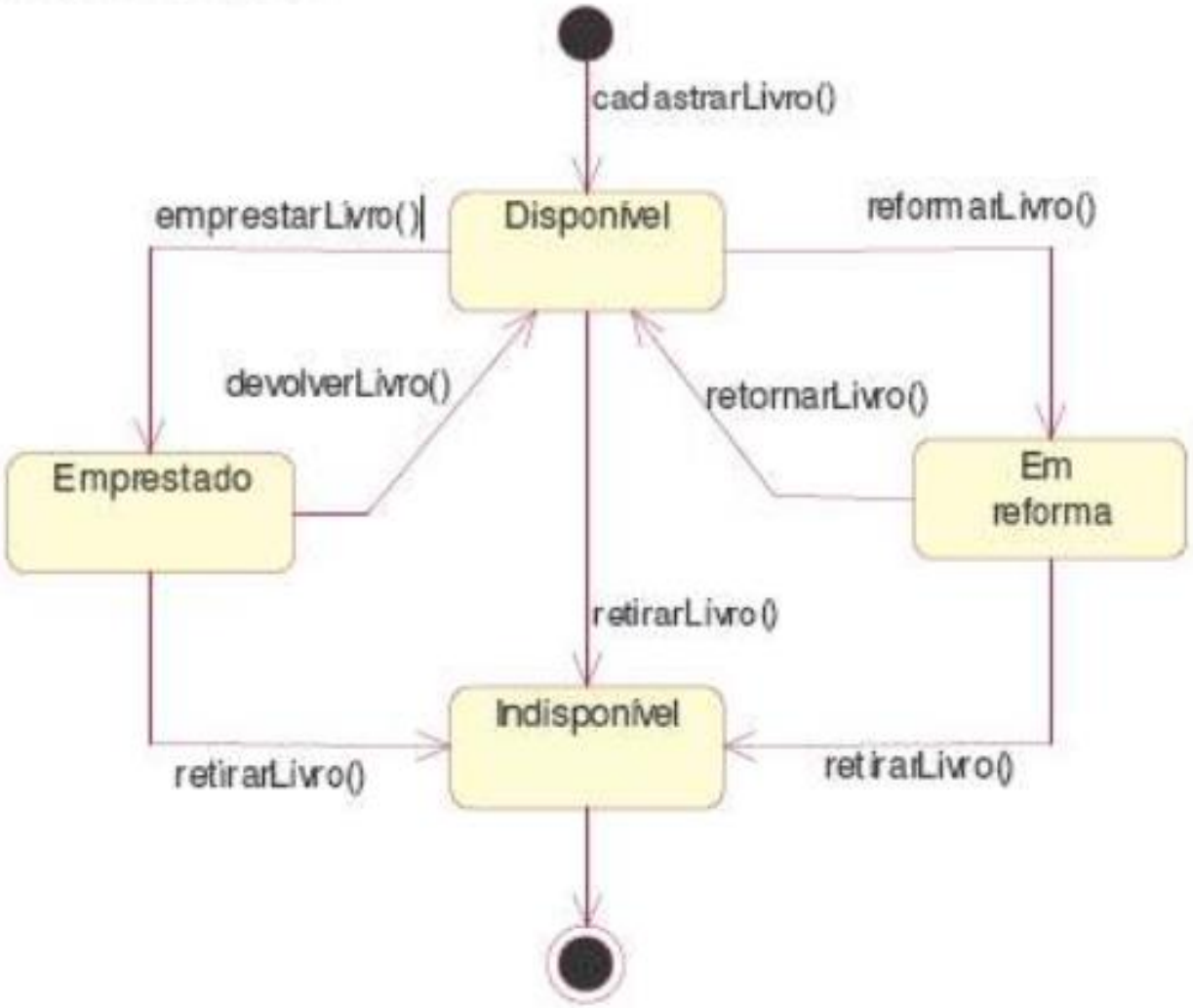
- Os DMEs podem ser construídos com base nos diagramas de interação e nos diagramas de classes.
- Durante a construção do DME para uma classe, novos atributos e operações podem surgir.
  - Essas novas propriedades devem ser adicionadas ao modelo de classes.
- A construção de um DME freqüentemente leva à descoberta de novos atributos para uma classe
  - principalmente atributos para servirem de abstrações para estados.
- Além disso, este processo de construção permite identificar novas operações na classe
  - pois os objetos precisam reagir aos eventos que eles recebem.

Modelagem de estados no PDS

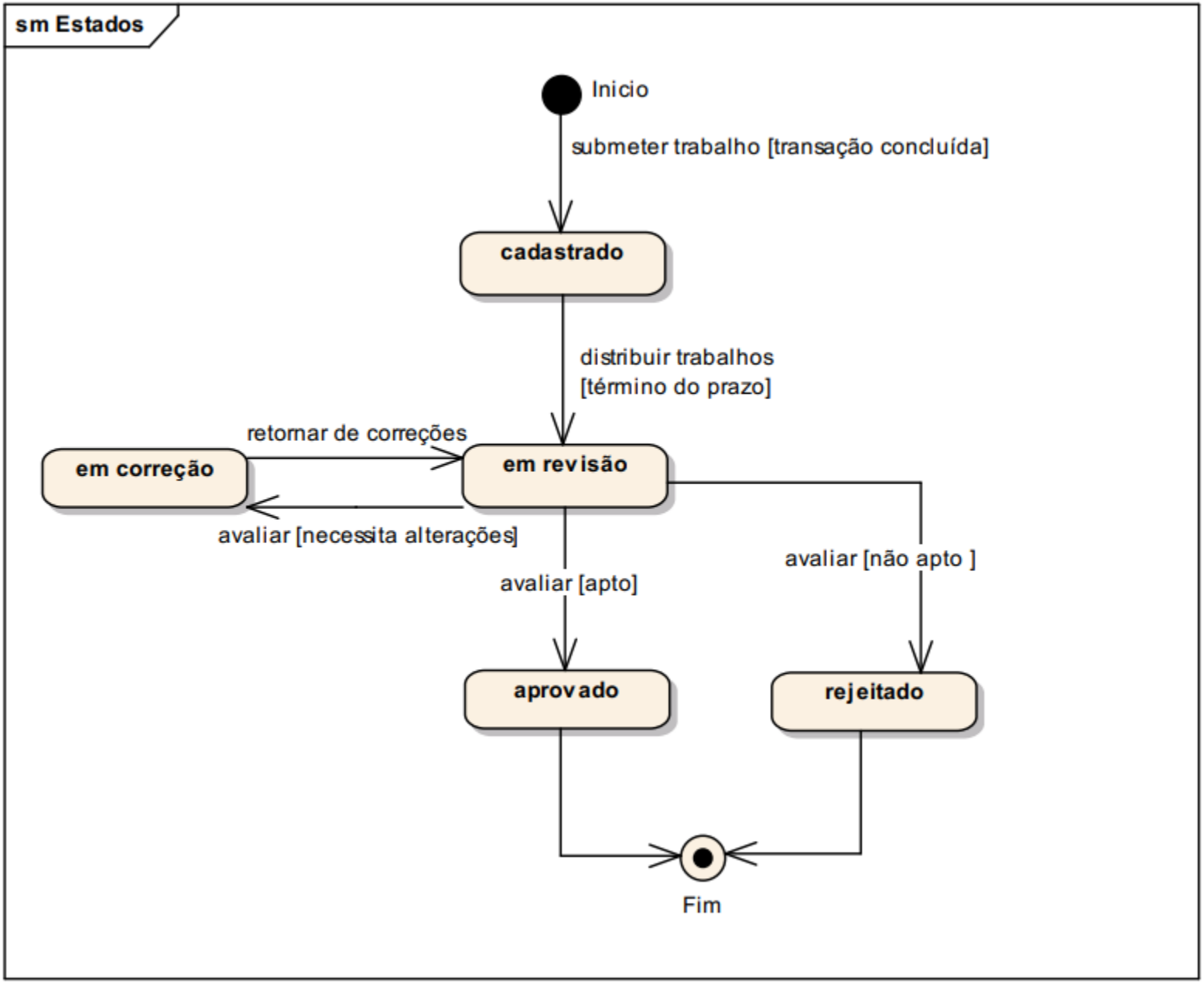
- O comportamento de um objeto varia em função do estado no qual ele se encontra.
- Pode ser necessária a atualização de uma ou mais operações de uma classe para refletir o comportamento do objetos em cada estado.
- Por exemplo, o comportamento da operação **sacar()** da classe **ContaBancária** varia em função do estado no qual esta classe se encontra
  - saques não podem ser realizados em uma conta que esteja no estado **bloqueada**.

1) Faça um **Diagrama de Estados** para a Classe **Livro**: Um sistema de gestão de biblioteca permite que seus usuários emprestem livros, mediante um cadastramento prévio. Quando o usuário deseja emprestar um livro, ele verifica inicialmente se o livro está disponível. Somente livros disponíveis podem ser emprestados. Ao efetuar o empréstimo, é feito o registro do mesmo e o livro é entregue para o usuário, juntamente com a informação de quando o livro deverá ser devolvido. Quando o usuário devolve o livro emprestado, é dada a baixa no empréstimo e o livro torna-se disponível novamente para empréstimo. De tempos em tempos, a bibliotecária retira os livros que não apresentam boas condições de uso para reforma. Durante o tempo em que o livro fica em reforma, esta informação é dada para o usuário que deseja emprestar o livro, juntamente com a data de previsão em que ele retornará da reforma. Quando a reforma do livro estiver concluída, o livro retorna à biblioteca, ficando novamente disponível para novos empréstimos. A bibliotecária registra também a retirada definitiva de um livro (quando este é roubado na biblioteca ou durante um empréstimo, ou ainda quando não é bem sucedido durante uma reforma), tornando-o assim, indisponível para empréstimos.

Diagrama de Estados



2) Faça o **Diagrama de Estados** para a Classe **Trabalho Científico**. Um congresso recebe a submissão de trabalhos científicos para serem apresentados em suas sessões. Um trabalho científico pode ser um artigo completo, um artigo resumido ou um tutorial. Ao submeter seu trabalho, se concluída a transação, o autor recebe a notificação de que o mesmo encontra-se cadastrado. Terminado o prazo de submissão de trabalhos, os trabalhos são distribuídos aos revisores. Neste momento, se o autor quiser saber a situação de seu trabalho, o sistema irá identificá-lo como estando em revisão. Após a revisão, o trabalho, se julgado apto para apresentação, é considerado aprovado. Caso não seja julgado apto para apresentação, então é considerado como rejeitado. Mas caso haja necessidades de alterações, o revisor faz comentários e devolve o trabalho para o autor, ficando, o trabalho, em correção até que o autor retorne com as modificações. Quando isto acontece, o trabalho volta ao revisor para nova análise, podendo-se repetir este ciclo várias vezes até o trabalho ser considerado definitivamente aprovado ou rejeitado.


sm Estados
Inicio
submeter trabalho [transação concluída]
cadastrado
distribuir trabalhos
[término do prazo]
retornar de correções
em correção
em revisão
avaliar [necessita alterações]
avaliar [não apto]
avaliar [apto]
aprovado
rejeitado
Fim

3) Faça um **Diagrama de Estados** para a **Classe Cliente**: Uma empresa de consultoria em melhoria de processos de software resolveu controlar os diversos estágios de seus clientes em virtude de alta demanda de serviços e para manter um histórico dos trabalhos realizados. Inicialmente um cliente quando solicita orçamento é identificado como em negociação. A partir do momento que o orçamento é aprovado e uma data é definida para avaliação do cliente este passa a ser agendado. Na semana combinada, a equipe de consultores realiza o processo de avaliação, passando o cliente para o estado em avaliação. Ao final deste processo é realizado um diagnóstico. Sendo aprovado, o cliente passa a ser certificado. Em caso de reprovação, o cliente passa por um processo de melhoria. O cliente faz os ajustes necessários de acordo com as orientações da consultoria e submete-se a uma nova avaliação. Se for aprovado é certificado, caso contrário continua em processo de melhoria. A qualquer momento durante a avaliação ou melhoria o cliente pode desistir do processo de consultoria e passa a ser identificado como desistente. Depois de três anos um cliente certificado passa a ser expirado.