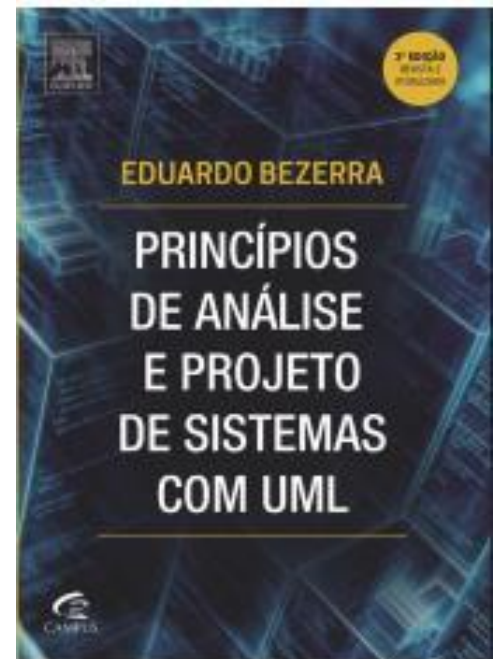


# **Princípios de Análise e Projeto de Sistemas com UML**

**3ª edição (2015)**

Eduardo Bezerra  
Editora Campus/Elsevier



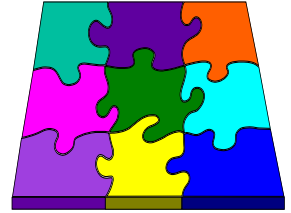
# Capítulo 10

## Modelagem de atividades

*Qualquer um pode escrever código que um computador pode entender. Bons programadores escrevem código que seres humanos podem entender.*

-- Martin Fowler

# Tópicos



- Diagrama de atividade
- Diagrama de atividade no processo de desenvolvimento iterativo

# Diagrama de atividade

- Há diversos diagramas da UML que descrevem os aspectos dinâmicos de um sistema.
  - diagramas de estados, diagramas de seqüência e de comunicação e **diagrama de atividade**
- O diagrama de atividade é um tipo especial de diagrama de estados, onde são representados os estados de uma atividade.
- Um diagrama de atividade exhibe passos de uma computação.
  - Cada atividade é um passo da computação.
  - É orientado a fluxos de controle (ao contrário dos DTEs que são orientados a eventos).
- São um tipo de *fluxograma estendido*..., pois permitem representar ações concorrentes e sua sincronização.

# Diagrama de atividade

- Elementos podem ser divididos em dois grupos: controle seqüencial e controle paralelo.
- Elementos utilizados em fluxos seqüenciais:
  - Estado ação
  - Estado atividade
  - Estados inicial e final, e condição de guarda
  - Transição de término
  - Pontos de ramificação e de união
- Elementos utilizados em fluxos paralelos:
  - Barras de sincronização
    - Barra de bifurcação (fork)
    - Barra de junção (join)

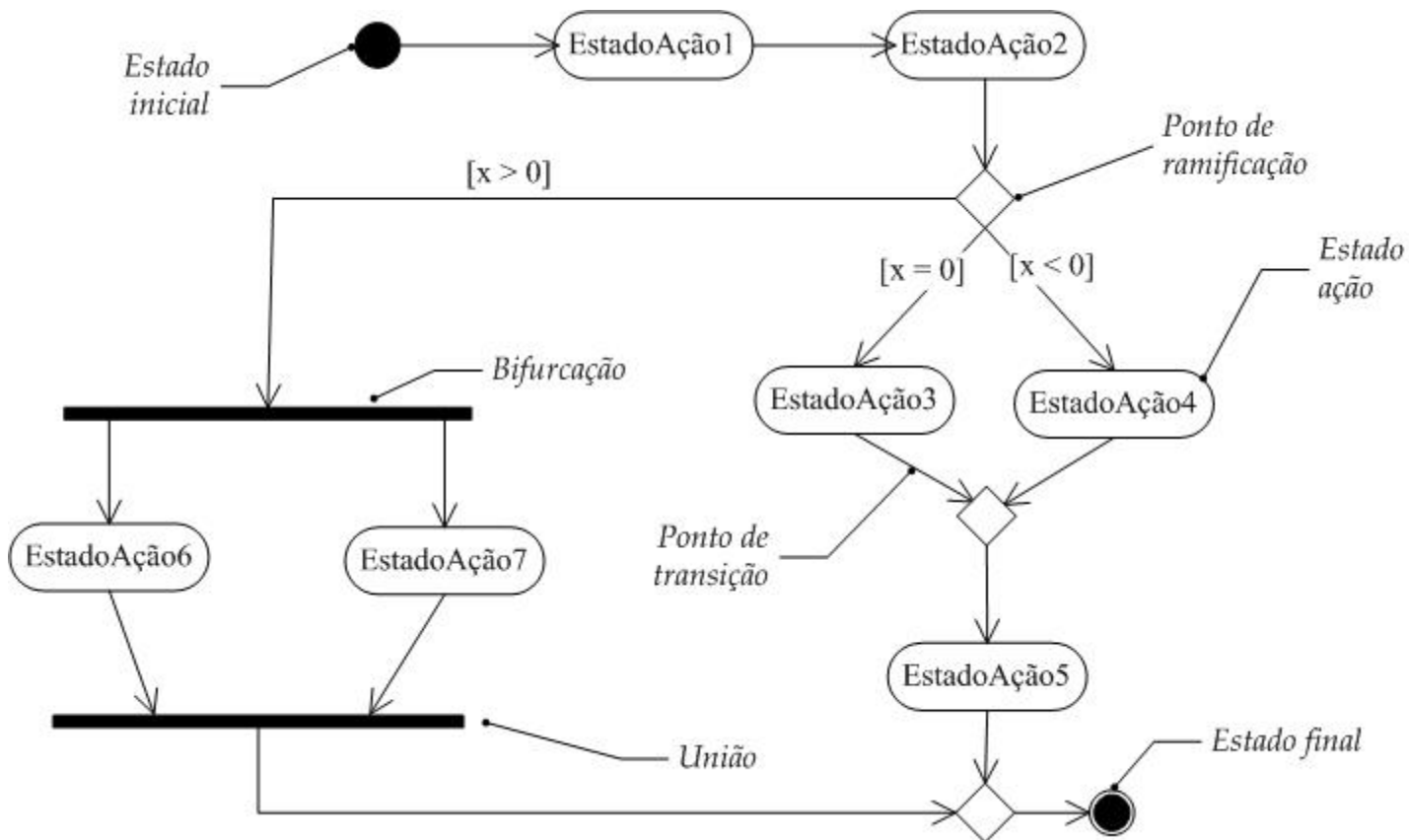
# Fluxos de controle seqüenciais

- Um estado em um diagrama de atividade pode ser:
  - um *estado atividade* leva um certo tempo para ser finalizado.
  - um *estado ação*: realizado instantaneamente.
- Deve haver um *estado inicial* e pode haver vários *estados finais* e *guardas* associadas a transições.
  - pode não ter estado final, o que significa que o processo ou procedimento é cíclico.
- Uma *transição de término* significa o término de um passo e o conseqüente início do outro.
  - Em vez de ser disparada pela ocorrência de um evento, é disparada pelo término de um passo.

# Fluxos de controle seqüenciais

- Um *ponto de ramificação* possui uma única transição de entrada e várias transições de saída.
  - Para cada transição de saída, há uma condição de guarda associada.
  - Quando o fluxo de controle chega a um ponto de ramificação, uma e somente uma das condições de guarda deve ser verdadeira.
  - Pode haver uma transição com **[else]**.
- Um *ponto de união* reúne diversas transições que, direta ou indiretamente, têm um ponto de ramificação em comum.

# Diagrama de atividade





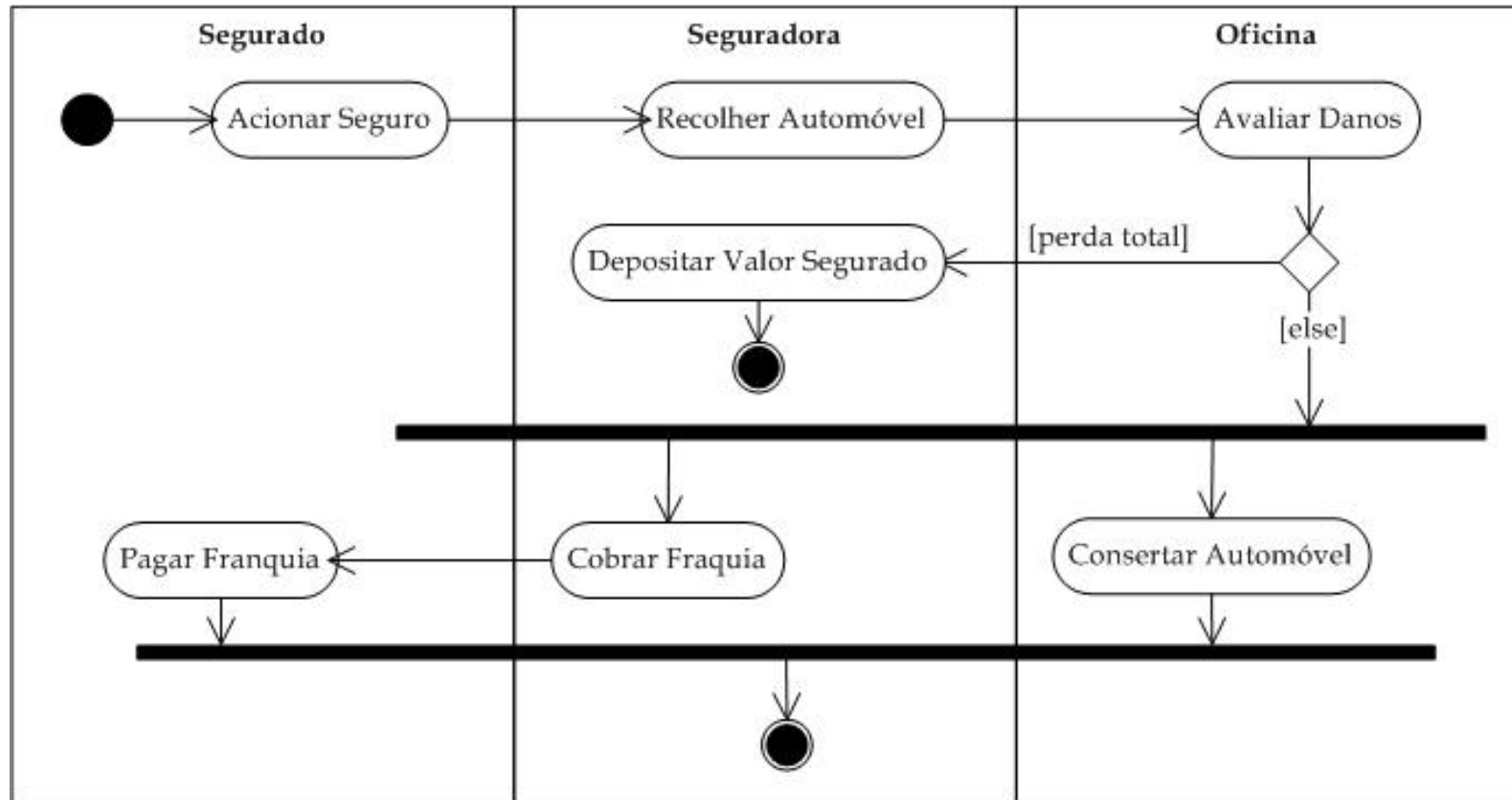
# Fluxos de controle paralelo

- Fluxos de controle paralelos: dois ou mais fluxos sendo executados simultaneamente.
- Uma **barra de bifurcação** recebe uma transição de entrada, e cria dois ou mais fluxos de controle paralelos.
  - cada fluxo é executado independentemente e em paralelo com os demais.
- Uma **barra de junção** recebe duas ou mais transições de entrada e une os fluxos de controle em um único fluxo.
  - Objetivo: sincronizar fluxos paralelos.
  - A transição de saída da barra de junção somente é disparada quando todas as transições de entrada tiverem sido disparadas.

# Fluxos de controle paralelos

- Algumas vezes, as atividades de um processo podem ser distribuídas por vários agentes que o executarão.
  - processos de negócio de uma organização.
- Isso pode ser representado através de *raias de natação* (swim lanes).
- As raias de natação dividem o diagrama de atividade em *compartimentos*.
- Cada compartimento contém atividades que são realizadas por uma entidade.

# Exemplo (Raias de Natação)



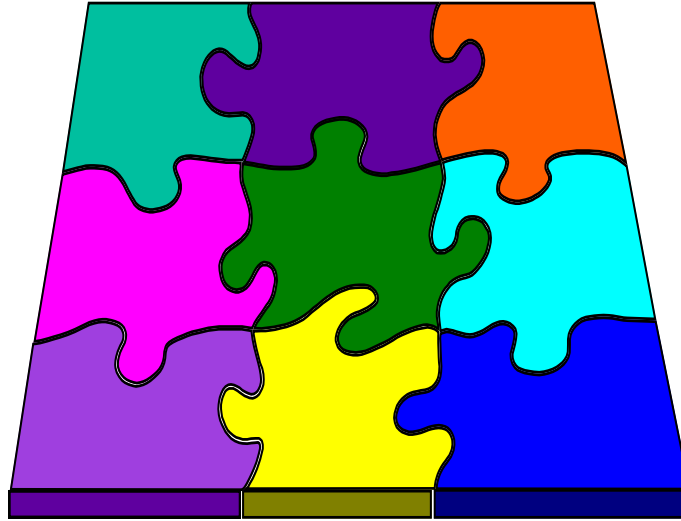


Diagrama de atividade no processo de desenvolvimento iterativo

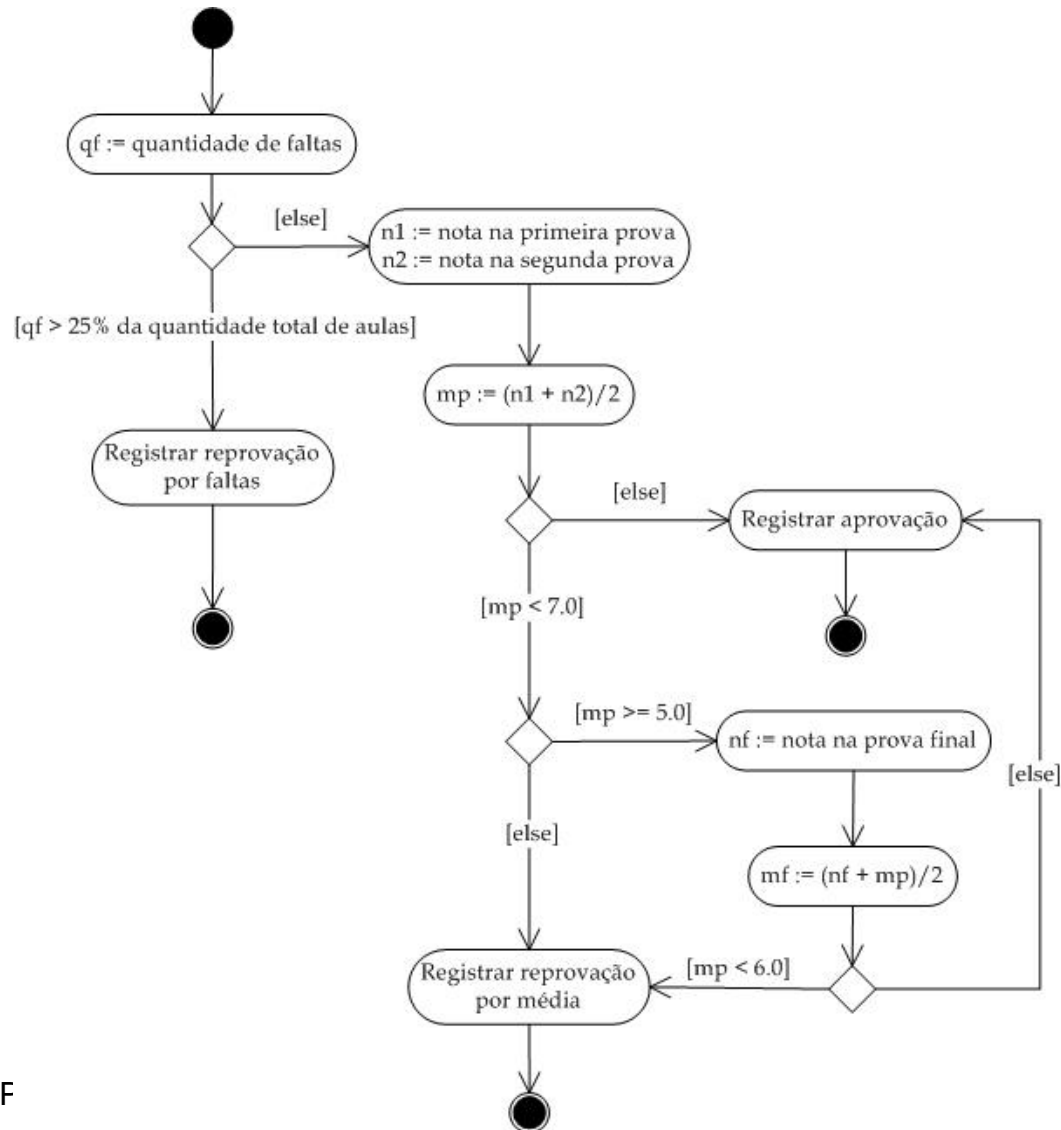
# Usos de diagramas de atividades

- Não são freqüentemente utilizados na prática...
- Importante: na orientação a objetos o sistema é dividido em objetos, e não em módulos funcionais como na Análise Estruturada (Diagrama de Fluxos de Dados).

# Modelar o processo do negócio

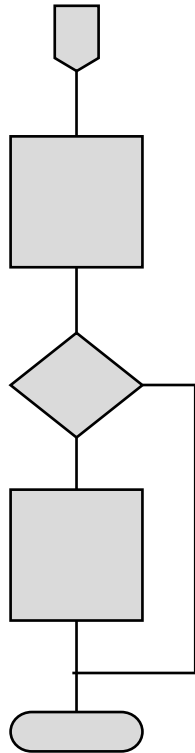
- *Modelagem* também é um processo de entendimento.
  - o desenvolvedor constrói modelos para entender melhor um problema.
- Neste caso, o enfoque está em entender o comportamento do sistema no decorrer de diversos casos de uso (*processos de negócio*).
  - como determinados casos de uso do sistema se relacionam no decorrer do tempo.

# Modelar o processo do negócio



# Modelar a lógica de um caso de uso

- A realização de um caso de uso requer que alguma computação seja realizada.
  - Esta computação pode ser dividida em atividades.
  - “Passo P ocorre até que a C seja verdadeira”
  - “Se ocorre C, vai para o passo P”.
- Nessas situações, é interessante complementar a descrição do caso de uso com um diagrama de atividade.

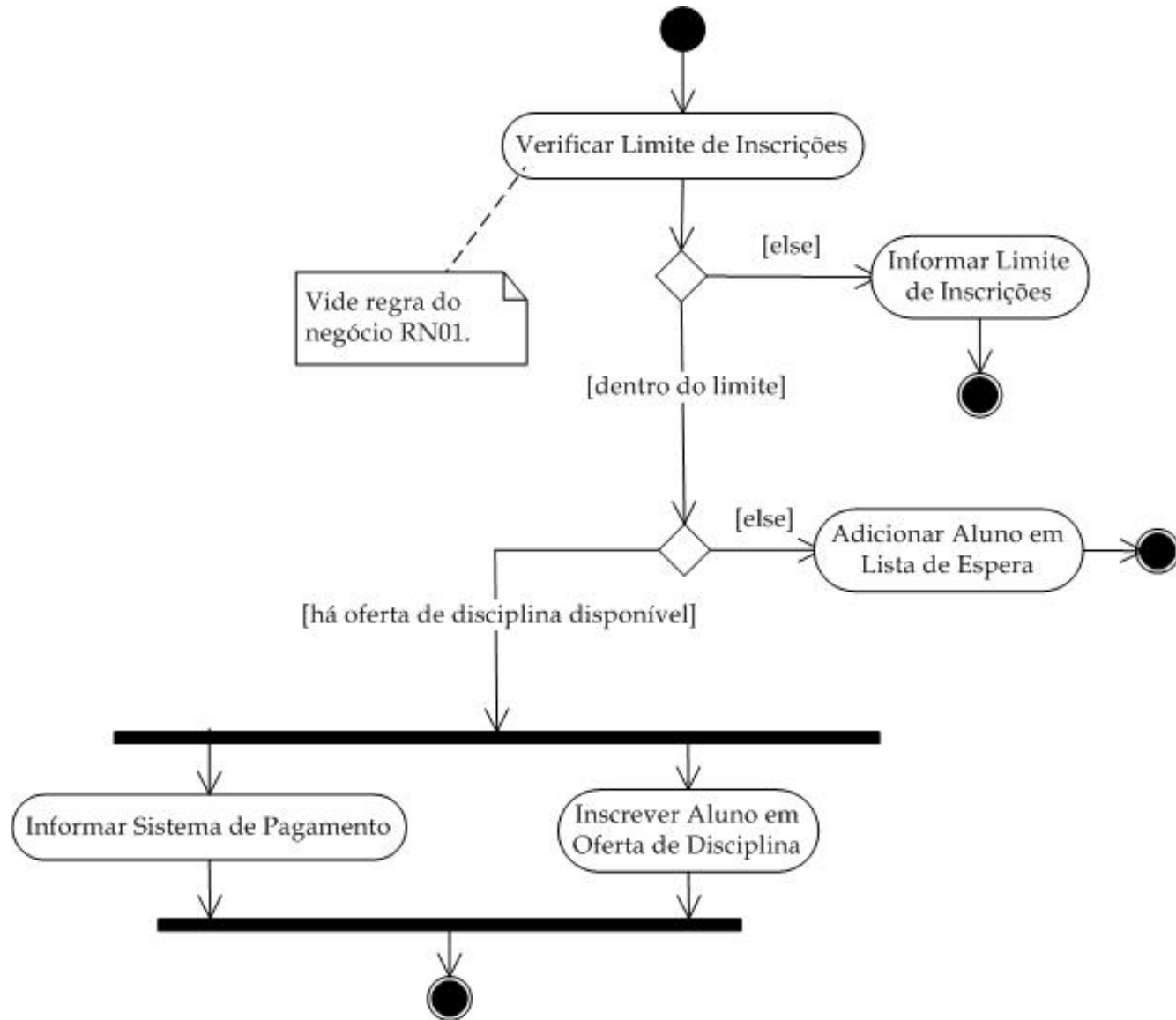




# Modelar a lógica de um caso de uso

- Os fluxos principal, alternativos e de exceção podem ser representados em um único diagrama de atividade.
  - complementar e não substituir a descrição.
- Identificação de atividades através do exame dos fluxos do caso de uso.
- Casos de uso são descritos na perspectiva dos atores, enquanto diagramas de atividade descrevem atividades internas ao sistema.

# Modelar a lógica de um caso de uso



# Modelar a lógica de uma operação

- Quando um sistema é adequadamente decomposto em seus objetos, a maioria das operações são bastante simples.
  - Estas não necessitam de modelagem gráfica.
- No entanto, pode haver a necessidade de descrever a lógica de uma operação mais complexa.
  - Implementação de regras de negócio.