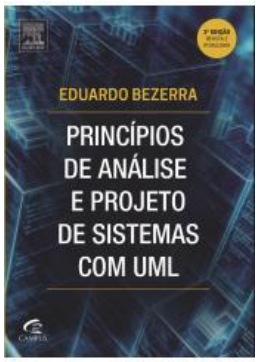


Princípios de Análise e Projeto de Sistemas com UML

3ª edição, 2015, Eduardo Bezerra

Alguns Exercícios Resolvidos

Capítulo 1

Exercício 1.1

Sim, porque ele representa graficamente um objeto do mundo real (a cidade).
Características:

- Ele abstrai e só exibe as informações consideradas relevantes do objeto.
- (Encapsulamento) O objeto Mapa fornece serviços a outros objetos através de sua interface. Todos os detalhes de como esses serviços são implementados estão encapsulados pelo objeto. Os "clientes" deste objeto só precisam conhecer a sua interface para utilizá-lo.
- (Herança) Um objeto mapa pode ser considerado como um tipo especial de figura e como tal pode herdar propriedades e operações de uma figura (carregar, salvar, aumentar (zoom-in), diminuir (zoom-out), etc.).
- (Polimorfismo) Um objeto mapa pode ser formado de outros objetos (por exemplos, rios, rodovias, lagos, etc.). Quando houver a necessidade de desenhar o mapa, a mesma mensagem (desenhar) pode ser enviada a seus objetos componentes. Ao receber tal mensagem, cada objeto componente a implementa da forma mais adequada para o seu caso.

Exercício 1.2

- **a)** Mensagens são enviadas para outros objetos através da sua interface.
- **b)** Objetos têm uma fronteira (a sua interface). Cada objeto tem um comportamento interno que não é visível de fora (pelo princípio do encapsulamento).
- **c)** Objetos de um sistema de software colaboram entre si para realizar as funcionalidades externamente visíveis desse sistema. Esses objetos se comunicam requisitando serviços uns aos outros para resolver problemas ou para realizar uma função do sistema ao qual pertencem.

Exercício 1.3

Sim. Porque de acordo com o primeiro desses princípios, qualquer coisa é um objeto.

Exercício 1.4

Há sim. Os modelos utilizados, tanto no contexto do livro, quanto nessas áreas, servem para tornar algo complexo mais inteligível para a capacidade humana e também usam símbolos e notações para representar os objetos reais.

Exercício 1.5

- **Objeto**: um objeto é qualquer coisa que possamos ver, sentir, tocar ou idealizar. Exemplos de objetos são: Eu, esse texto, sua conta de email, etc. Computacionalmente falando, um objeto é uma abstração de algo do mundo real. Em vista disso, um objeto nesse é bem mais simples que a sua contrapartida original e representas as características e comportamentos relevantes desse original.
- **Classes**: corresponde a um agrupamento de objetos. Ela retém os comportamento que serão característicos aos objetos que dela fazem parte. Por exemplo: Pessoa é uma classe onde estamos contidos todos nós, porém quando pensamos em alguém em específico, estamos pensando no objeto, quando idealizamos uma imagem de uma pessoa fizemos usando as características da classe. Por exemplo: Camisa é uma classe. A camisa que estou usando agora é um objeto dessa classe.
- **Herança**: é a capacidade que uma classe tem de herdar características da classe que está acima dela hierarquicamente. Por exemplo: Camisa é uma subclasse da classe Roupa, logo ela herda comportamentos e funções. Todas as duas tem a comportamento de servir como vestimentas para quem a usa.
- **Mensagem**: é um pedido ou informação passado de um objeto para outro. Por exemplo, quando giramos o objeto chave na ignição do carro, ele passa a mensagem para o carro de queremos que ele funcione, em reposta o carro executa essa tarefa.

Capítulo 4

Exercício 4.8

A tabela a seguir exibe as alternativas possíveis entre relacionamentos entre atores e casos de uso em um diagrama de casos de uso. As células da tabela com um X indicam possibilidade. As células não preenchidas indicam impossibilidade.

	Entre atores	Entre casos de uso	Entre ator e caso de uso
Comunicação			X
Inclusão		X	
Extensão		X	
Generalização	X	X	

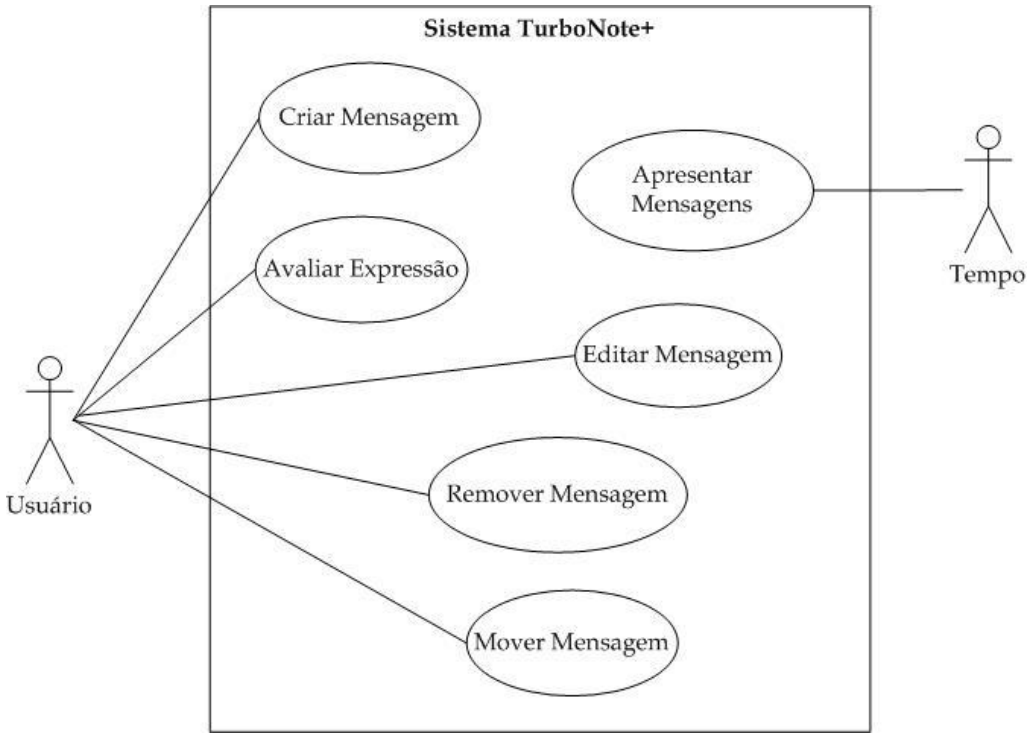
Exercício 4.14

Considerando-se somente o trecho fornecido no exercício, podem ser identificados 3 atores em potencial, a saber: Cliente, Vendedor e Depósito. O primeiro é o ator primário e os dois últimos são atores secundários. O nome do caso de uso correspondente poderia ser Comprar Produtos.

Exercício 4.16

Na situação descrita neste exercício, pode-se definir um ator denominado Empregado . Este seria o ator primário no caso de uso Registrar Horas Trabalhadas. Podemos também criar um ator denominado Gerência. que seria o ator primário no caso de uso Obter Horas Trabalhadas. O diagrama de casos de uso a seguir ilustra a solução aqui descrita.

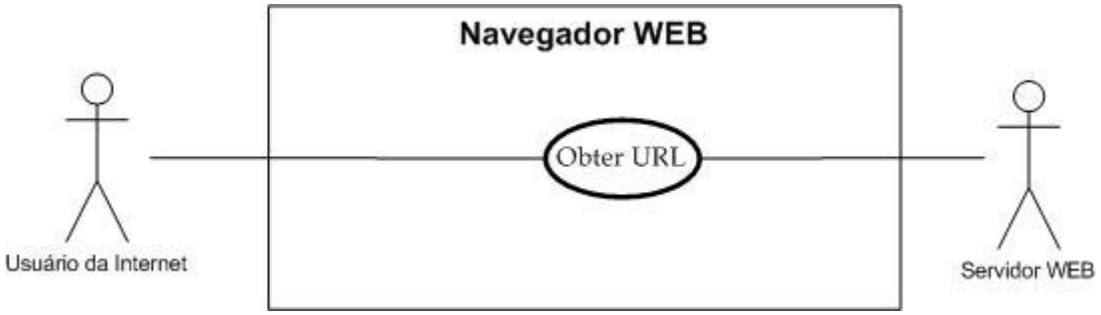
Exercício 4.17

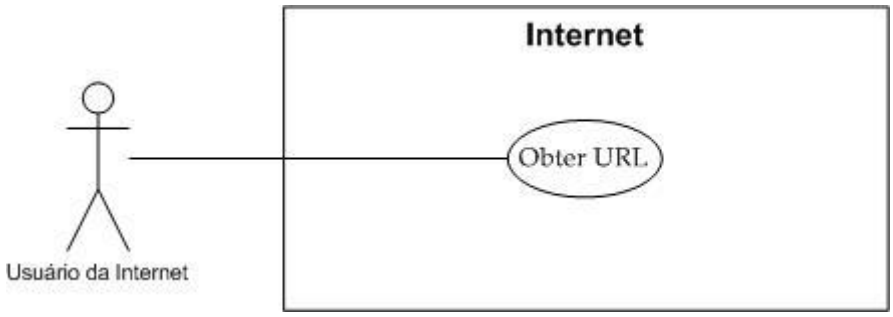


Exercício 4.19

O erro do diagrama está no fato de que atores não se comunicam em um diagrama de casos de uso.

Na primeira alternativa de escopo (o sistema é o browser), o sistema sendo modelado tem que se comunicar (interagir) com dois atores no caso de uso Obter URL: Usuário da Internet e Servidor WEB. Na segunda alternativa de escopo (o sistema é a Internet), o próprio servidor WEB faz parte do sistema sendo modelado e por conta disse não deve ser representado como um ator. Os dois diagramas de casos de uso a seguir ilustram as duas alternativas diferentes.





Exercício 4.20

A primeira e a segunda assertiva são verdadeiras. Na verdade essas assertivas são formas diferentes de declarar a mesma informação: *um ator representa um papel em relação ao sistema*.

Considere o exemplo do exercício 4-16. Pode haver uma pessoa que seja um funcionário comum em um certo projeto, além de ser o gerente em outro projeto. Neste caso, a mesma pessoa assumirá papéis diferentes em instantes distintos em relação ao sistema.

Exercício 4.21

A seguir, são apresentados os nomes de casos de uso de acordo com a nomenclatura adotada no livro. Possíveis nomes para atores primários em cada situação são também fornecidos. Deve-se enfatizar no entanto que isso é somente uma convenção de nomenclatura. Outras convenções podem ser usadas.

a.	Transferir Fundos	Cliente
b.	Comprar Livros	Usuário
c.	Obter Relatório de Vendas	Gerência
d.	Abrir Estadia	Hóspede

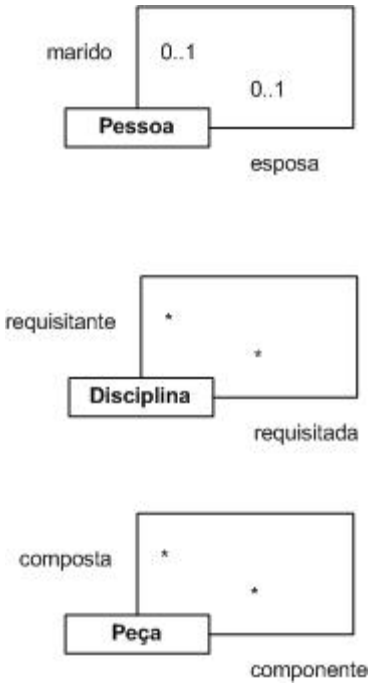
Capítulo 5

Exercício 5-2

Na modelagem de cartões CRC, utiliza-se cartões de tamanho fixo (normalmente com as dimensões aproximadas de 10cm x 15cm). O fato de as dimensões utilizadas serem as mesmas para todo cartão contribui para uma distribuição mais uniforme das responsabilidades. Isso porque quando o cartão CRC correspondente a uma certa classe já foi todo preenchido com responsabilidades, e uma nova responsabilidade deve ser

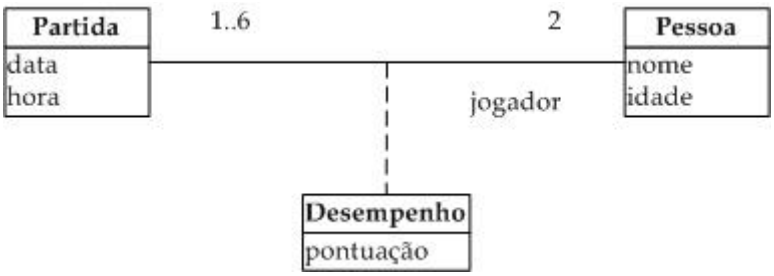
atribuída, é hora de o modelador considerar a criação de uma nova classe para cumprir com essa responsabilidade, ou então atribuir essa responsabilidade a uma outra classe..

Exercício 5-4

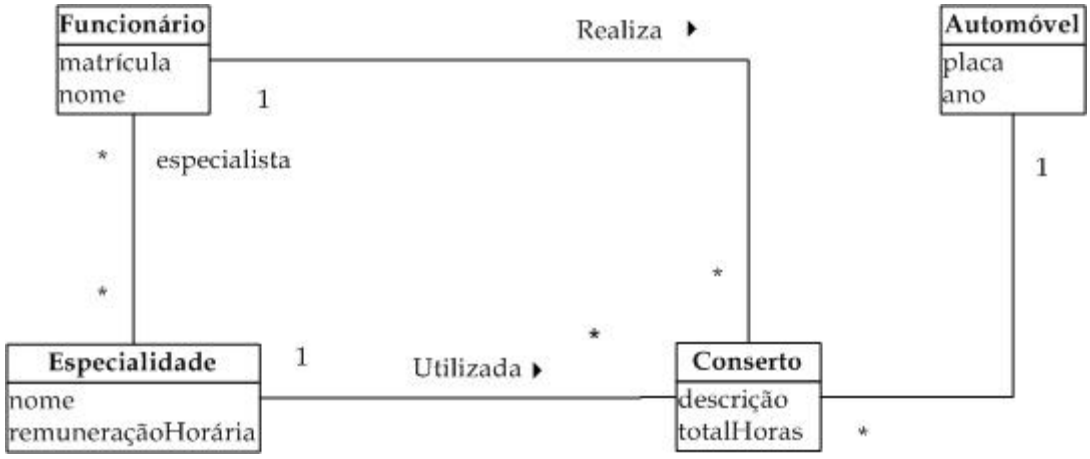


Exercício 5-5

Exercício 5-7



Exercício 5-8



Exercício 5-11

Em um modelo de classes, as responsabilidades atribuídas aos objetos devem ser o mais uniformemente distribuídas. Em muitos casos, um modelo no qual há uma classe que seja responsável pela maioria das atribuições do sistema muito provavelmente está mal balanceado quanto à distribuição de responsabilidades. Sempre que o modelador precisar de mais do que as dimensões usuais de um cartão CRC para enumerar as responsabilidades de uma classe, ele deve ser questionar se esta classe não está sobrecarregada com muitas responsabilidades.

Capítulo 7

Exercício 7-3

O diagrama de sequência apresenta uma estrutura de repetição aninhada. A ordem na qual as mensagens são passadas é a seguinte: m1, m2, m2, m2, m1, m2, m2, m2.

Exercício 7-6

Este exercício diz respeito a duas estruturas de controle encontradas em diagramas de interação: a centralizada e a descentralizada.

Estrutura centralizada

Há um pequeno grupo de objetos que controla a realização de uma tarefa do sistema e coordena outros objetos periféricos através do envio de mensagens. A inteligência do sistema está concentrada nesse pequeno grupo de objetos controladores. De fato, os objetos desse pequeno grupo podem se encaixar na categoria de "objetos controladores" (veja detalhes no Capítulo 5).

Na situação extrema desta estrutura de controle, um único objeto contém quase toda a lógica da aplicação e somente obtém pequenos serviços de outros objetos enviando mensagens para eles. Em relação à atribuição de responsabilidades, significa que uma grande quantidade de responsabilidades foi atribuída àquele objeto, tornando-o excessivamente complexo. Muito provavelmente seria mais adequado que algumas de suas responsabilidades fossem atribuídas a outro(s) objeto(s). Nessa situação, a estrutura de controle assume a forma de um garfo (fork) no diagrama de sequência.

Estrutura descentralizada

Dada uma tarefa que o sistema deve realizar, partes dessa tarefa são delegadas a vários objetos. A tarefa é realizada de forma descentralizada. Não há um único grande objeto controlador. Em vez disso, cada objeto faz uma parte da tarefa e envia uma mensagem para outro(s) objeto(s) realizarem uma outra parte. Cada objeto conhece apenas poucos outros objetos. Note que a "inteligência" do sistema fica distribuída pelos objetos que participam da realização da tarefa.

Na situação extrema da estrutura de controle descentralizada, uma tarefa do sistema é realizada por uma sequência de envios de mensagens entre objetos O_1, O_2, ..., O_n, onde o objeto O_i envia uma mensagem para o objeto O_{i+1}. Nesse sentido, cada objeto controla o seu sucessor na sequência de mensagens enviadas. Nessa situação, a estrutura de controle assume a forma de uma escada (stair) no diagrama de sequência.

Vantagens e desvantagens

Há vantagens de desvantagens em ambas as estruturas de controle. A utilização de uma ou de outra estrutura depende muito do problema de modelagem a ser resolvido. No entanto, na prática, um modelo mais flexível e robusto é obtido quando se combinam

essas duas estratégias, considerando suas características particulares. É muito pouco provável que uma boa modelagem de um determinado diagrama de interação seja feita através da utilização exclusiva de uma ou de outra estratégia.

Uma estrutura de controle descentralizada é apropriada quando:

- Os objetos têm uma conexão forte (por agregação, por composição, ou por generalização).
- As mensagens serão sempre enviadas em uma mesma ordem.

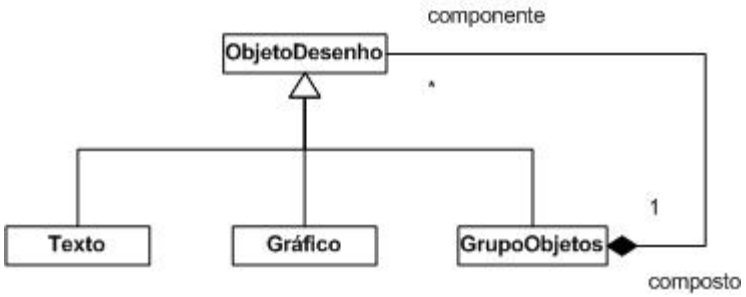
Uma estrutura de controle centralizada é apropriada quando:

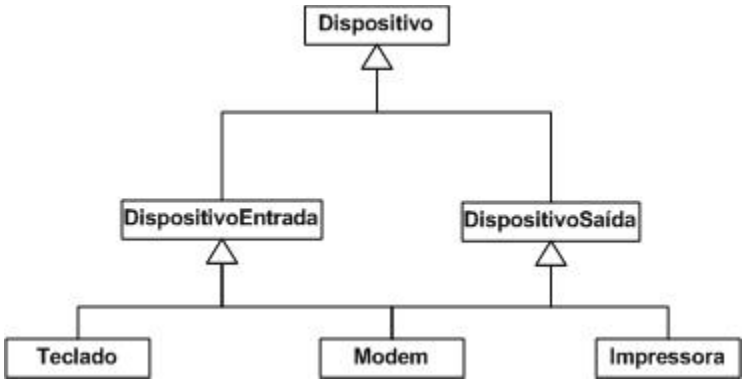
- As mensagens podem ser enviadas em ordens diferentes.
- Novas operações poderiam ser inseridas.

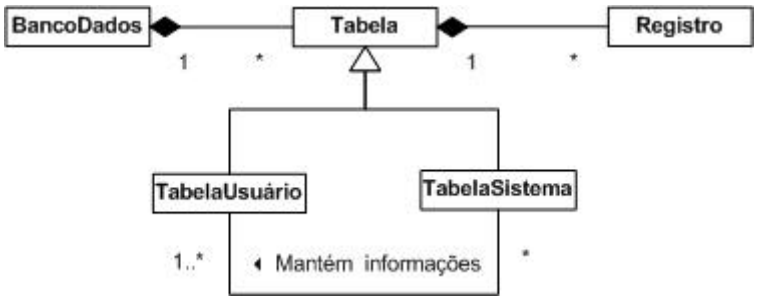
A distribuição de responsabilidades segunda categorias (fronteira, controle e entidade) também pode ajudar na identificação de que estrutura de controle é mais adequada. Normalmente, costumo utilizar uma estrutura centralizada em relação ao objeto controlador de um caso de uso. No entanto, para outros objetos que participem da realização desse caso de uso e que se relacionem por agregação, por composição, ou por generalização, acho mais adequado usar uma estrutura descentralizada. Portanto, em um mesmo caso de uso, é comum a utilização das duas estruturas de controle.

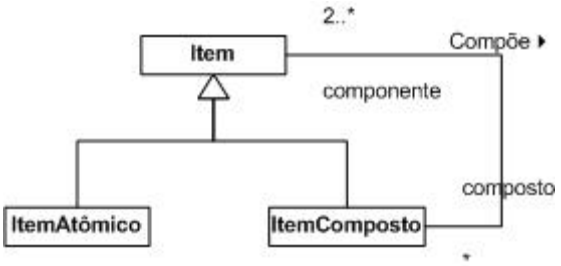
Capítulo 08

Exercício 08-06

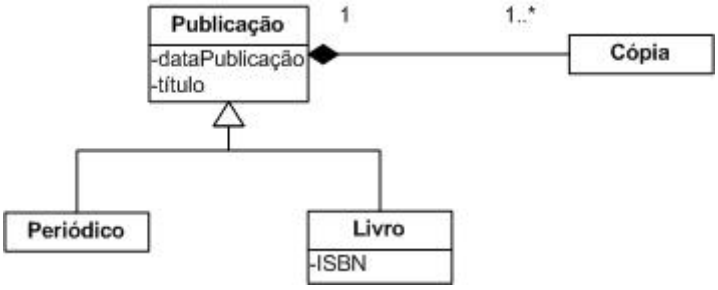




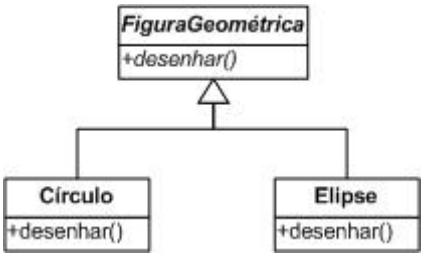




Exercício 08-07



Exercício 08-08



Exercício 08-09