Modelagem de Classes

Prof. Humberto Torres Marques Neto
Novembro / 2018
PUC MINAS

Principais Objetivos

- ✓ Apresentar os conceitos relacionados às disciplinas de modelagem de negócio e modelagem estrutural a partir dos requisitos identificados.
- ✓ Discutir e propor soluções para os principais problemas do processo de análise.
- ✓ Desenvolver habilidades de uso de técnicas de análise, modelagem de negócio e modelagem estrutural.

Referências Bibliográficas Básicas

BEZERRA, Eduardo. Princípios de análise e projeto de sistemas com UML. Rio de Janeiro: Campus, 2002.

BOOCH, Grady, RUMBAUGH, James, JACOBSON, Ivair. UML: Guia do Usuário. 2 ed. Rio de Janeiro: Campus, 2005.

COAD, Peter, YOURDON, Edward. Análise baseada em objetos. Rio de Janeiro: Campus, 1996. 225p.

FOWLER, Martin, SCOTT, Kendall. UML Essencial: um breve guia para linguagem-padrão de modelagem de objetos. 2 ed. Porto Alegre: Bookman, 2000.

LARMAN, Craig. Utilizando UML e padrões: uma introdução à análise e projeto orientado a objetos. 2 ed. Porto Alegre: Bookman, 2004.

Algumas Definições

- ✓ Domínio do problema
  - "Um campo de atividades sob estudo ou consideração" (COAD e YOURDON, 1996. p. 7)
- ✓ Complexidade do domínio do problema
  - "... a software development project alters the rules of the problem." (BOOCH, 1994. p. 5)
- ✓ Responsabilidade do sistema
  - "Uma organização de elementos relacionados de modo a formar um todo." (COAD e YOURDON, 1996. p. 8)

O que é um Objeto?

- ***"Uma abstração de alguma coisa em um domínio de problemas, exprimindo as capacidades de um sistema de manter informações sobre ela, interagir com ela, ou ambos; um encapsulamento de valores de Atributos e de seus Serviços exclusivos. (sinônimo: Ocorrência)." (COAD e YOURDON, 1996. p. 50)***
- ***"... an object represents an individual, identifiable item, unit, or entity, either real or abstract, with a well-definied role in the problem domain." (SMITH and TOCKEY, apud BOOCH,1994. p. 82)***
- ***"An object has state, behavior, and identity; the structure and behavior of similar objects are defined in their common class; the terms instance and object are interchageable" (BOOCH, 1994. p. 83)***
- ***"Um objeto é qualquer coisa, real ou abstrata, a respeito da qual armazenamos dados e os métodos que os manipulam." (MARTIN e ODELL, 1995. p. 18)***
- ***"Um objeto é simplesmente alguma coisa que faz sentido no contexto de uma aplicação." (RUMBAUGH et. al. p. 31)***

O que é uma Classe de Objetos?

- ***"Uma descrição de um ou mais Objetos com um conjunto uniforme de Atributos e Serviços, incluindo uma descrição de como criar novos Objetos na Classe." (COAD e YOURDON, 1996. p. 50)***
- ***"A class is a set of objects that share a common structure and a common behavior" (BOOCH, 1994. p. 103)***
- ***"Uma classe é uma implementação de um tipo de objeto. Ela especifica uma estrutura de dados e os métodos operacionais permissíveis que se aplicam a cada um de seus objetos." (MARTIN e ODELL, 1995. p. 26)***
- ***"Uma classe de objetos descreve um grupo de objetos com propriedades semelhantes (atributos), o mesmo comportamento (operações), os mesmos relacionamentos com outros objetos e a mesma semântica " (RUMBAUGH et. al. p. 32)***
- ***"Uma classe é uma descrição de um conjunto de objetos que compartilham os mesmos atributos, operações, relacionamentos e semântica" (BOOCH et al., 2000 p. 49)***

Diagrama de Classes da UML (BOOCH et al, 2005)

- ✓ Um *diagrama de classes* é um diagrama que mostra um conjunto de classes, interfaces e colaborações e seus relacionamentos.
- ✓ São utilizados para fazer a modelagem da visão estática de um sistema.
- ✓ Oferece suporte para os requisitos funcionais de um sistema.

Abstrações do Modelo de Classes

- ✓ **O *modelo de classes de domínio*** representa as classes no domínio do negócio em questão. Não leva em consideração restrições inerentes à tecnologia a ser utilizada na solução de um problema.
- ✓ **O *modelo de classes de especificação*** é obtido através da adição de detalhes ao modelo anterior conforme a solução de software escolhida.
- ✓ **O *modelo de classes de implementação*** corresponde à implementação das classes em alguma linguagem de programação.

Transparência retirada de (BEZERRA, 2002)

Identificação de Classes (Como procurar?)

- ✓ Identificação de conceitos relacionados ao domínio do problema.
- ✓ A partir do detalhamento dos casos de uso (identificação de substantivos e estruturas substantivas).
  - Vantagem: ***simplicidade***
  - Desvantagem: ***depende da qualidade do detalhamento***
- ✓ Definição das responsabilidades das classes.

Identificação de Classes (O que procurar?)

- ✓ Lista de Categorias de Classes Conceituais (LARMAN, 2004)
  - Objetos físicos ou tangíveis (Registro, Aeronave)
  - Especificações, projetos ou descrição de coisas (Especificação de Produto, Descrição de Vôo)
  - Lugares (Loja, Aeroporto)
  - Transações (Venda, Pagamento, Reserva)
  - Linhas de itens de transações (Linha de Item de Venda)
  - Papéis desempenhados por pessoas (Caixa, Piloto)
  - Contêineres de outras coisas (Loja, Armário, Aeronave)
  - Coisas de um contêiner (Item, Passageiro)

Identificação de Classes (O que procurar?)

- ✓ Lista de Categorias de Classes Conceituais (LARMAN, 2004)
  - Outros sistemas (Sistema de Autorização de Pagamento, Controle de Tráfego Aéreo, Bomba de Combustível)
  - Substantivos abstratos (Fome, Acrofobia)
  - Organizações (Departamento de Vendas, Objeto Linha Aérea)
  - Eventos (Venda, Pagamento, Reunião, Vôo, Acidente, Aterrissagem)
  - Processos (Vendendo um Produto, Reservando um Assento)
  - Regras e políticas (Política de Reembolsos, Política de Cancelamentos)

Identificação de Classes (O que procurar?)

- ✓ Lista de Categorias de Classes Conceituais (LARMAN, 2004)
  - Catálogos (Catálogo de Produtos, Catálogo de Peças)
  - Registro financeiros, trabalhistas, de contratos, de assuntos legais (Recibo, Diário de Caixa, Contrato de Emprego, Diário de Manutenção)
  - Serviços e instrumentos financeiros (Linha de Crédito, Ações)
  - Manuais, documentos, artigos de referência (Lista Diária de Mudança de Preços, Manual de Consertos)

Identificação de classes

- ✓ Um software construído no paradigma da orientação a objetos é composto de objetos em colaboração para realizar as tarefas deste sistema.
- ✓ Como todo objeto pertence a uma classe, quando se fala na identificação das classes, o objetivo na verdade é saber quais objetos irão compor o sistema.

Transparência retirada de (BEZERRA, 2002)

Identificação de classes

- ✓ De uma forma geral, a identificação de classes se divide em duas atividades.
  - Primeiramente, ***classes candidatas*** são identificadas.
  - Depois disso, são aplicados alguns princípios para eliminar classes candidatas desnecessárias.
- ✓ Identificar *possíveis* classes para um sistema não é complicado; o difícil é eliminar deste conjunto o que não é necessário.

Transparência retirada de (BEZERRA, 2002)

Identificação de classes dirigida a responsabilidades

- ✓ Método de identificação onde a ênfase está na identificação de classes a partir de seus *comportamentos* externos relevantes para o sistema.
  - *como* a classe faz para cumprir com suas responsabilidades deve ser abstraído.
- ✓ O esforço recai sobre a identificação das *responsabilidades* que cada classe deve ter.
- ✓ "O método *dirigido a responsabilidades enfatiza o encapsulamento da estrutura e do comportamento dos objetos.*".

Transparência retirada de (BEZERRA, 2002)

Responsabilidades e colaboradores

- ✓ Em sistemas OO, objetos encapsulam tanto dados quanto comportamento.
- ✓ O comportamento de um objeto é definido de tal forma que ele possa cumprir com suas ***responsabilidades***.
- ✓ Uma responsabilidade é uma obrigação que um objeto tem para com o sistema no qual ele está inserido.
  - Através delas, um objeto colabora (ajuda) com outros para que os objetivos do sistema sejam alcançados.

Transparência retirada de (BEZERRA, 2002)

Responsabilidades e colaboradores

- ✓ Na prática, uma responsabilidade é alguma coisa que um objeto *conhece* ou *faz. (sozinho ou não)*.
  - Um objeto Cliente *conhece* seu nome, seu endereço, seu telefone, etc.
  - Um objeto Pedido *conhece* sua data de realização e sabe *fazer* o cálculo do seu total.
- ✓ Se um objeto tem uma responsabilidade com a qual não pode cumprir sozinho, ele deve requisitar ***colaborações*** de outros objetos.

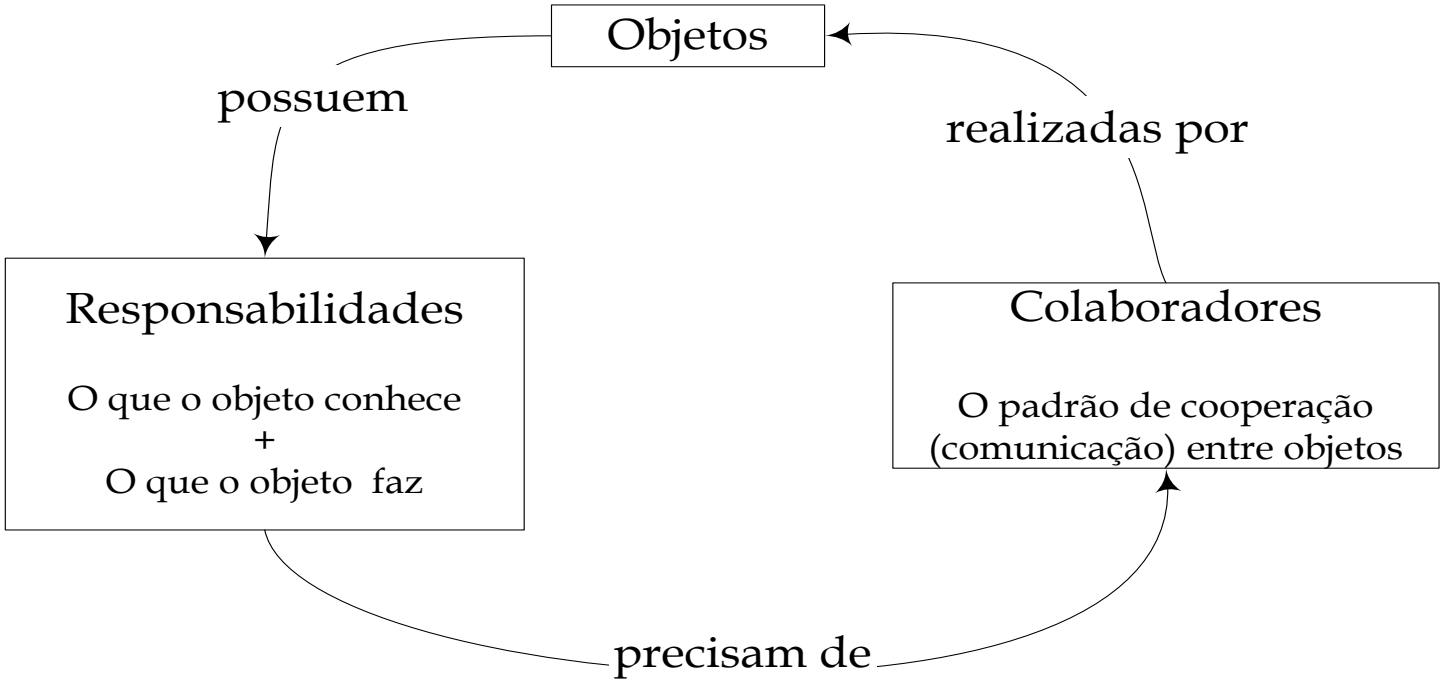
Transparência retirada de (BEZERRA, 2002)

Responsabilidades e colaboradores

- ✓ *Um exemplo*: quando a impressão de uma fatura é requisitada em um sistema de vendas, vários objetos precisam colaborar:
  - um objeto Pedido pode ter a responsabilidade de fornecer o seu valor total
  - um objeto Cliente fornece seu nome
  - cada Item de Pedido informa a quantidade correspondente e o valor de seu subtotal
  - os objetos Produto também colaboraram fornecendo seu nome e preço unitário.

Transparência retirada de (BEZERRA, 2002)

Responsabilidades e colaboradores



Transparência retirada de (BEZERRA, 2002)

Categorias de responsabilidades

- ✓ Costuma-se categorizar os objetos de um sistema de acordo com o tipo de responsabilidade a ele atribuída.
  - objetos de entidade
  - objetos de controle
  - objetos de fronteira
- ✓ Esta categorização foi proposta por Ivar Jacobson em uma técnica denominada Análise de Robustez.

Transparência retirada de (BEZERRA, 2002)

Objetos de Entidade

- ✓ Um objeto de entidade é um repositório para alguma ***informação*** manipulada pelo sistema.
- ✓ Esses objetos representam conceitos do domínio do negócio.
- ✓ Normalmente esses objetos armazenam informações persistentes.
- ✓ Há várias instâncias de uma mesma classe de entidade coexistindo no sistema.

Transparência retirada de (BEZERRA, 2002)

Objetos de Entidade

- ✓ Atores não têm acesso direto a estes objetos.
  - Objetos de entidade se comunicam com o exterior do sistema por intermédio de outros objetos.
- ✓ Objetos de entidade normalmente participam de vários casos de uso e têm um ciclo de vida longo.
  - Um objeto *Pedido* pode participar dos casos de uso *Realizar Pedido* e *Atualizar Estoque*. Este objeto pode existir por diversos anos ou mesmo tanto quanto o próprio sistema.

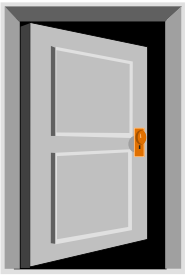
Transparência retirada de (BEZERRA, 2002)

Objetos de Entidade

- ✓ Responsabilidades *de fazer* típicas de objetos de entidade:
  - Informar valores de seus atributos a objetos de controle.
  - Realizar cálculos simples, normalmente com a colaboração de objetos de entidade associados através de agregações.
  - Criar e destruir objetos parte (considerando que o objeto de entidade seja um objeto todo de uma agregação).

Transparência retirada de (BEZERRA, 2002)

Objetos de Fronteira



- ✓ Esses objetos traduzem os eventos gerados por um ator em eventos relevantes ao sistema.
- ✓ Também são responsáveis por apresentar os resultados de uma interação dos objetos em algo inteligível pelo ator.
- ✓ Um objeto de fronteira existe para que o sistema se comunique com o mundo exterior.
- ✓ Por conseqüência, estes objetos são altamente dependentes do ambiente.

Transparência retirada de (BEZERRA, 2002)

Objetos de Fronteira

- ✓ Classes de fronteira realizam a comunicação do sistema com atores, sejam eles outros sistemas, equipamentos ou seres humanos.
- ✓ Há três tipos principais de classes de fronteira:
  - as que se comunicam com o usuário (atores humanos),
  - as que se comunicam com outros sistemas
  - as que se comunicam com dispositivos atrelados ao sistema.

Transparência retirada de (BEZERRA, 2002)

Objetos de Fronteira

- ✓ Tipicamente têm as seguintes responsabilidades *de fazer*:
  - Notificar aos objetos de controle de eventos gerados externamente ao sistema.
  - Notificar aos atores do resultado de interações entre os objetos internos.

Transparência retirada de (BEZERRA, 2002)

Objetos de Fronteira

- ✓ Responsabilidades *de conhecer* de classes de fronteira para interação humana representam informação manipulada através da interface com o usuário.
  - A construção de protótipos pode ajudar a identificar essas responsabilidades.
- ✓ Responsabilidades de conhecer para classes de fronteira que realizam comunicação com outros sistemas representam propriedades de uma interface de comunicação.

Transparência retirada de (BEZERRA, 2002)

Objetos de Controle

- ✓ São a “ponte de comunicação” entre objetos de fronteira e objetos de entidade.
- ✓ Responsáveis por controlar a lógica de execução correspondente a um caso de uso.
- ✓ Decidem o que o sistema deve fazer quando um evento externo relevante ocorre.
  - Eles realizam o controle do processamento
  - Agem como ***gerentes*** (coordenadores, controladores) dos outros objetos para a realização de um ou mais caso de uso.

Transparência retirada de (BEZERRA, 2002)

Objetos de Controle

- ✓ São bastante acoplados à lógica da aplicação.
- ✓ Traduzem eventos externos em operações que devem ser realizadas pelos demais objetos.
- ✓ Ao contrário dos objetos de entidade e de fronteira, objetos de controle são tipicamente ativos
  - consultam informações e requisitam serviços de outros objetos.

Transparência retirada de (BEZERRA, 2002)

Objetos de Controle

- ✓ Responsabilidades *de fazer* típicas:
  - Realizar monitorações, a fim de responder a eventos externos ao sistema (gerados por objetos de fronteira).
  - Coordenar a realização de um caso de uso através do envio de mensagens a objetos de fronteira e objetos de entidade.
  - Assegurar que as regras do negócio estão sendo seguidas corretamente.
  - Coordenar a criação de associações entre objetos de entidade.

Transparência retirada de (BEZERRA, 2002)

Objetos de Controle

- ✓ Responsabilidades *de conhecer* estão associadas manter valores acumulados, temporários ou derivados durante a realização de um caso de uso.
- ✓ Podem também ter o objetivo de manter o estado da realização do caso de uso.
- ✓ Têm vida curta: normalmente existem somente durante a realização de um caso de uso.

Transparência retirada de (BEZERRA, 2002)

Divisão de responsabilidades

- ✓ A categorização de responsabilidades implica em que cada objeto é especialista em realizar um de três tipos de tarefa:
  - se comunicar com atores (fronteira)
  - manter as informações do sistema (entidade)
  - coordenar a realização de um caso de uso (controle).
- ✓ A importância dessa categorização está relacionada à capacidade de adaptação do sistema a eventuais mudanças

Transparência retirada de (BEZERRA, 2002)

Divisão de responsabilidades

- ✓ Se cada objeto tem funções específicas dentro do sistema, eventuais mudanças no sistema podem ser:
  1. menos complexas
  2. mais localizadas.
- ✓ Uma eventual modificação em uma parte do sistema tem menos possibilidades de resultar em mudanças em outras partes.

Transparência retirada de (BEZERRA, 2002)

Divisão de responsabilidades

Tipo de mudança	Onde mudar
Mudanças em relação à interface gráfica, ou em relação à comunicação com outros sistemas.	Fronteira
Mudanças nas informações manipuladas pelo do sistema	Entidade
Mudanças em funcionalidades complexas (lógica do negócio)	Controle

Transparência retirada de (BEZERRA, 2002)

Divisão de responsabilidades

- ✓ Exemplo: vantagem de separação de responsabilidades em um sistema para uma loja de aluguel de carros.
  - Se o sistema tiver que ser atualizado para que seus usuários possam utilizá-lo pela Internet, a lógica da aplicação não precisaria de modificações.
  - Considerando a lógica para calcular o valor total das locações feitas por um cliente: se esta lógica estiver encapsulada em uma classe de controle, somente esta classe precisaria de modificação.

Transparência retirada de (BEZERRA, 2002)

Divisão de responsabilidades

- ✓ A construção de um sistema que faça separação das responsabilidades de apresentação (fronteira), de lógica da aplicação (controle) e de manutenção dos dados (entidade):
  - facilita também o reuso dos objetos no desenvolvimento de sistemas de software semelhantes.
  - ajuda no desacoplamento entre elementos do sistema

Transparência retirada de (BEZERRA, 2002)

Modelagem CRC (*Class Responsabilities and Colaborators*)

- ✓ Se baseia fortemente no paradigma da orientação a objetos, onde objetos cooperam uns com os outros para que uma tarefa do sistema seja realizada.
- ✓ Efetiva quando profissionais que não têm tanta experiência com o paradigma da orientação a objetos estão envolvidos na identificação de classes.
  - realizada em conjunto por especialistas de domínio e desenvolvedores

Transparência retirada de (BEZERRA, 2002)

Modelagem CRC

- ✓ A modelagem CRC não faz parte da UML.
- ✓ A princípio, essa técnica foi proposta como uma forma de ensinar o paradigma da orientação a objetos a iniciantes.
- ✓ Contudo, a sua simplicidade de notação a tornou particularmente interessante para ser utilizada na identificação de classes de domínio.

Transparência retirada de (BEZERRA, 2002)

Modelagem CRC

- ✓ Especialistas do negócio e desenvolvedores trabalham em conjunto para identificar classes, juntamente com suas responsabilidades e colaboradores.
- ✓ Estes profissionais se reúnem em uma sala, onde tem início uma ***sessão CRC***.
- ✓ Uma sessão CRC envolve por volta de meia dúzia de pessoas: especialistas de domínio, projetistas, analistas e um moderador.
- ✓ A cada pessoa é entregue um ***cartão CRC***.

Transparência retirada de (BEZERRA, 2002)

Cartão CRC

Nome da classe (especialidade)	
Responsabilidades	**Colaboradores**
1ª responsabilidade 2ª responsabilidade ... n-ésima responsabilidade	1º colaborador 2º colaborador .. n-ésimo colaborador

Transparência retirada de (BEZERRA, 2002)

Exemplo (Cartão CRC)

ContaBancária (entidade)	
Responsabilidades	**Colaboradores**
1.Conhecer o seu cliente. 2.Conhecer o seu número. 3.Conhecer o seu saldo. 4.Manter um histórico de transações. 5.Aceitar saques e depósitos.	Cliente HistóricoTransações

Transparência retirada de (BEZERRA, 2002)

Modelagem CRC

- ✓ Na distribuição dos cartões pelos participantes, deve-se considerar as categorias de responsabilidades.
- ✓ Para cada cenário de caso de uso típico, pode-se começar com:
  - um objeto de fronteira para cada ator participante do caso de uso;
  - um objeto de controle para todo o caso de uso;
  - normalmente há vários objetos de entidade.

Transparência retirada de (BEZERRA, 2002)

Modelagem CRC

- ✓ Configuração inicial:
  - O moderador da sessão pode desempenhar o papel do objeto controlador
  - Outro participante desempenha o papel do objeto de fronteira.
  - Um outro participante pode simular o ator (ou atores, se houver mais de um).
  - Os demais representam objetos de entidade.

Transparência retirada de (BEZERRA, 2002)

Modelagem CRC

- ✓ Uma vez distribuídos os cartões pelos participantes, um conjunto de cenários de cada caso de uso é selecionado.
- ✓ Para cada cenário, uma sessão CRC é realizada.
  - Se o caso de uso não for tão complexo, ele pode ser analisado em uma única sessão.
- ✓ Normalmente já existem algumas classes candidatas para um certo cenário.

Transparência retirada de (BEZERRA, 2002)

Modelagem CRC

- ✓ A sessão CRC começa com a simulação do ator primário disparando a realização do caso de uso.
- ✓ Os demais participantes encenam a colaboração entre objetos para que o objetivo do ator seja alcançado.
- ✓ Através dessa encenação, as classes, responsabilidades e colaborações são identificadas.

Transparência retirada de (BEZERRA, 2002)

Modelagem CRC (procedimento)

1. Selecionar um conjunto de cenários de casos de uso.
2. Para um dos cenários:
   a) Examinar a sua seqüência de passos para identificar as responsabilidades do sistema em relação a cada um desses passos.
   b) Identificar classes relevantes que devem cumprir com as responsabilidades.
3. Repetir o passo 2 para o próximo cenário e modificar a alocação de responsabilidades e a definição de classes.

Transparência retirada de (BEZERRA, 2002)

Dicas para atribuição de responsabilidades

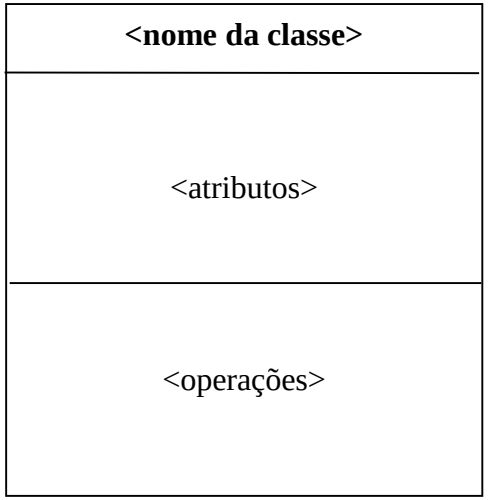
- ✓ Associar responsabilidades com base na especialidade da classe.
- ✓ Distribuir a inteligência do sistema
- ✓ Agrupar as responsabilidades conceitualmente relacionadas
- ✓ Evitar responsabilidades redundantes

Transparência retirada de (BEZERRA, 2002)

Representação Gráfica

- ✓ Uma classe é representada graficamente como um retângulo sub-dividido em três partes distintas:
  - **Nome da classe** (começa com letra maiúscula, deve estar no singular e estar coerente com o domínio do problema)
  - **Atributos** (representa alguma propriedade do item que está sendo modelado para todos, ou a maioria, dos objetos da classe)
  - **Operações** (é uma abstração de algo que pode ser feito com um objeto da classe; pode ser um conjunto vazio)

Representação Gráfica



Exemplos

Aluno
Matrícula Nome Endereço DataVestibular ...
matricular() trancarMatrícula() formar() cancelarMatrícula() alterarDados()

Imóvel
Identificação Endereço Proprietário DataInícioAluguel DataFimAluguel
cadastrar() alugar() finalizarAluguel() ...

PedidoCompra
Número DataPedido DataAtendimento Cliente
incluir() cancelar() emitir() atualizar()

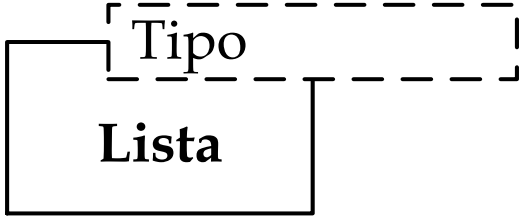
Classe Parametrizada

- ✓ Uma coleção pode ser representada em um diagrama de classes através uma **classe parametrizada** (*template*) .
  - É uma classe utilizada para definir outras classes.
  - Possui operações ou atributos cuja definição é feita em função de um ou mais parâmetros.
- ✓ Uma coleção pode ser definida a partir de uma classe parametrizada, onde o parâmetro é o tipo do elemento da coleção.

Transparência retirada de (BEZERRA, 2002)

Notação para uma Classe Parametrizada

- ✓ Há duas notações possíveis na UML para uma classe parametrizada.
- ✓ Exemplo: classe parametrizada **Lista**:



Lista<Tipo>

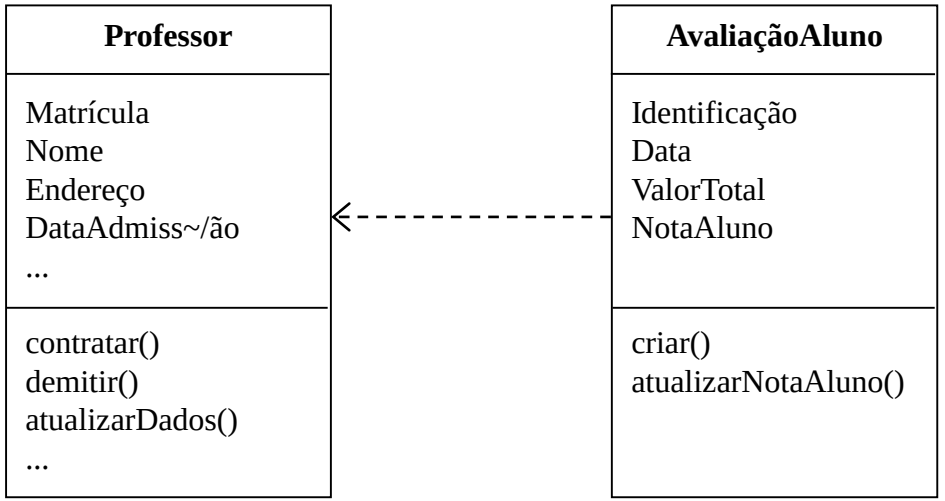
Transparência retirada de (BEZERRA, 2002)

Relacionamentos

- ✓ "... há um número muito pequeno de classes que trabalham sozinhas. Em vez disso, a maioria das classes colaboram com outras de várias maneiras." (*BOOCH et al., 2000 p. 60*)
- ✓ "Um relacionamento é uma conexão entre itens. Em uma modelagem orientada a objetos, os três relacionamentos mais importantes são as dependências, as generalizações e as associações. Um relacionamento é representado graficamente como um caminho, com tipos diferentes de linhas para diferenciar os tipos de relacionamentos." (*BOOCH et al., 2000 p. 62*)

Dependência

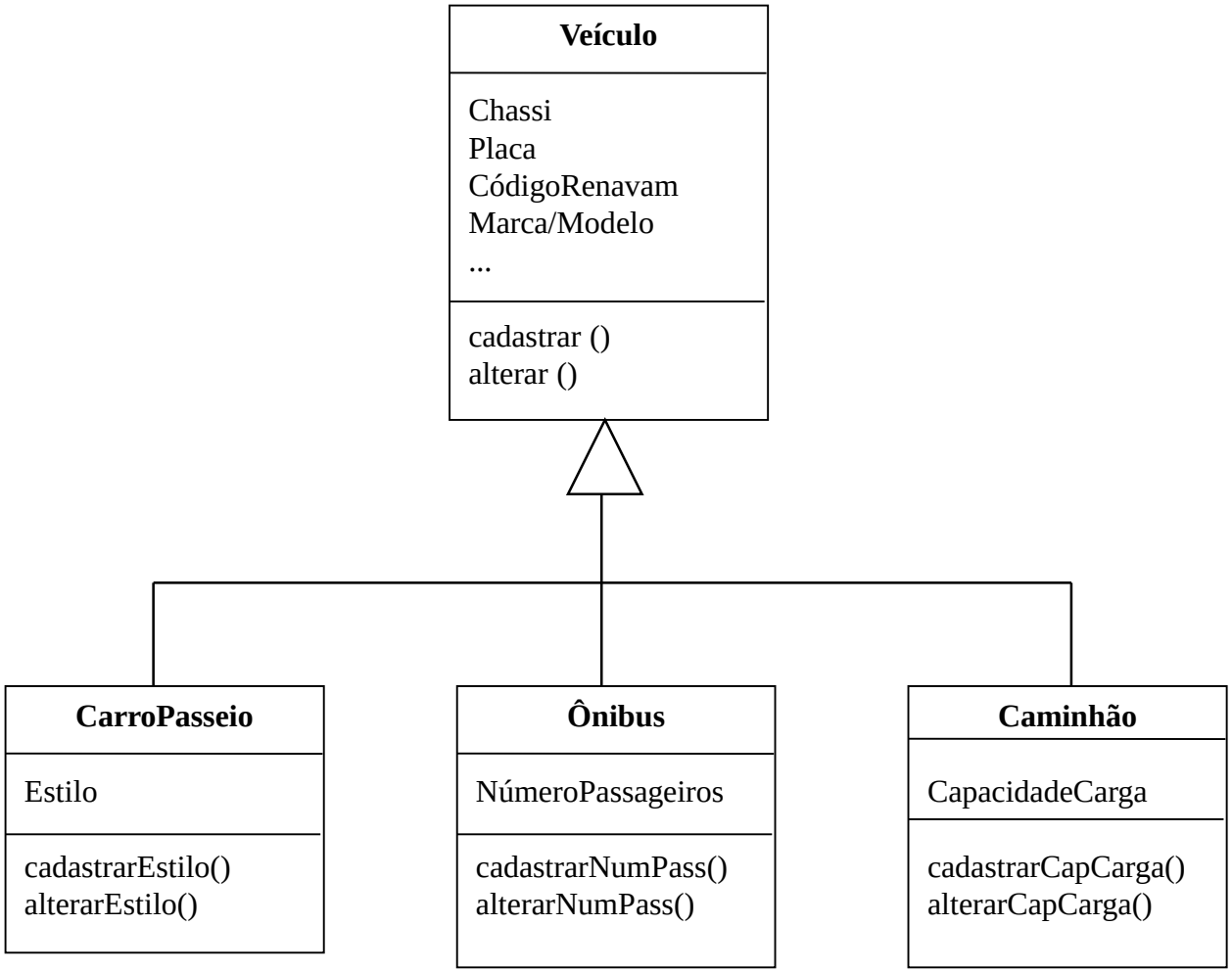
- ✓ “Uma dependência é um relacionamento de utilização, determinando as modificações na especificação de um item, (...) mas não necessariamente o inverso.” (*BOOCH et al., 2000 p. 62*)
- ✓ “... é representada graficamente como linhas tracejadas apontando o item do qual o outro depende.” (*BOOCH et al., 2000 p. 62*)
- ✓ Exemplo:



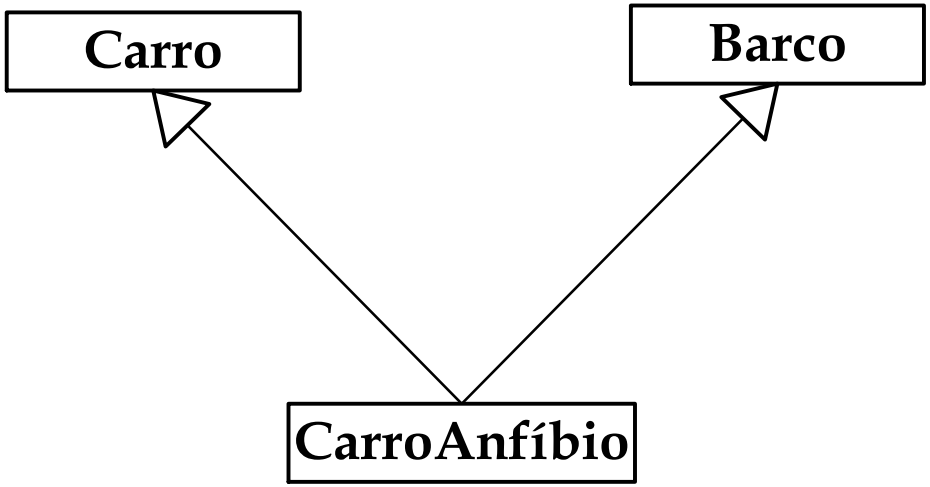
Generalização

- ✓ "... é um relacionamento entre itens gerais (chamados superclasses ou classes-mãe) e tipos mais específicos desses itens (chamados subclasses ou classes-filha)" (*BOOCH et al., 2000 p. 63*)
- ✓ São chamados de relacionamentos "**é um tipo de**"
- ✓ Os objetos da classe filha herdam propriedades da mãe, principalmente, seus atributos e operações, por isso, são utilizados para representar relacionamentos de herança
- ✓ Os objetos da classe filha podem ter atributos e operações específicos
- ✓ "... é representada graficamente como linhas sólidas apontando a mãe." (*BOOCH et al., 2000 p. 63*)

Generalização (Um exemplo)



Generalização (Herança Múltipla)



Exemplo extraído de BEZERRA, 2002

Associação

- ✓ "... é um relacionamento estrutural que especifica objetos de um item conectados a objetos de outros itens." (*BOOCH et al., 2000 p. 63*)
- ✓ Pode existir entre objetos da mesma classe
- ✓ "... é representada graficamente como uma linha sólida conectando a mesma classe ou classes diferentes." (*BOOCH et al., 2000 p. 64*)
- ✓ Uma associação pode ser caracterizada por:
  - Nome (identifica da natureza do relacionamento)
  - Papel

Associação (Como encontrar?)

- ✓ Lista de Associações Comuns (LARMAN, 2004)
  - A é uma parte física de B (Gaveta-Registro de um PDV, Asa-Aeronave).
  - A é uma parte lógica de B (Linha de Item de Venda e Venda, Perna de Vôo e Rota de Vôo)
  - A está fisicamente contido em B (Item-Prateleira, Passageiro-Aeronave)
  - A está logicamente contido em/sobre B (Descrição de Item e Catálogo, Vôo e Programação de Vôo)
  - A é uma descrição de B (Descrição de Item e Item, Descrição de Vôo e Vôo)
  - A é uma linha de item de uma transação ou relatório B (Linha de Item de Venda e Venda, Serviço de Manutenção e Registro de Manutenção)

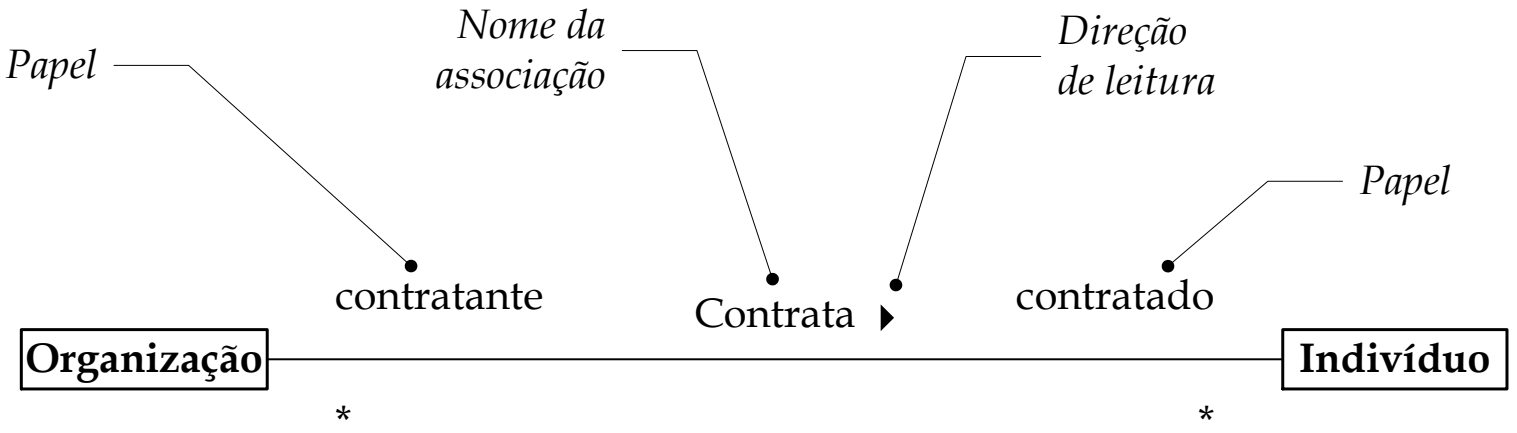
Associação (Como encontrar?)

- ✓ Lista de Associações Comuns (LARMAN, 2004)
  - A é conhecido/registrado/relatado/captado por B (Venda-Registro, Reserva e Manifesto de Vôo)
  - A é um membro de B (Caixa-Loja, Piloto e Linha de Aérea)
  - A é uma subunidade organizacional de B (Departamento-Loja, Manutenção-LinhaAérea)
  - A usa ou gerencia B (Caixa-Registro, Piloto-Aeronave)
  - A se comunica com B (Cliente-Caixa, Agente de Reservas e Passageiro)
  - A está relacionado a uma transação B (Cliente-Pagamento)

Associação (Como encontrar?)

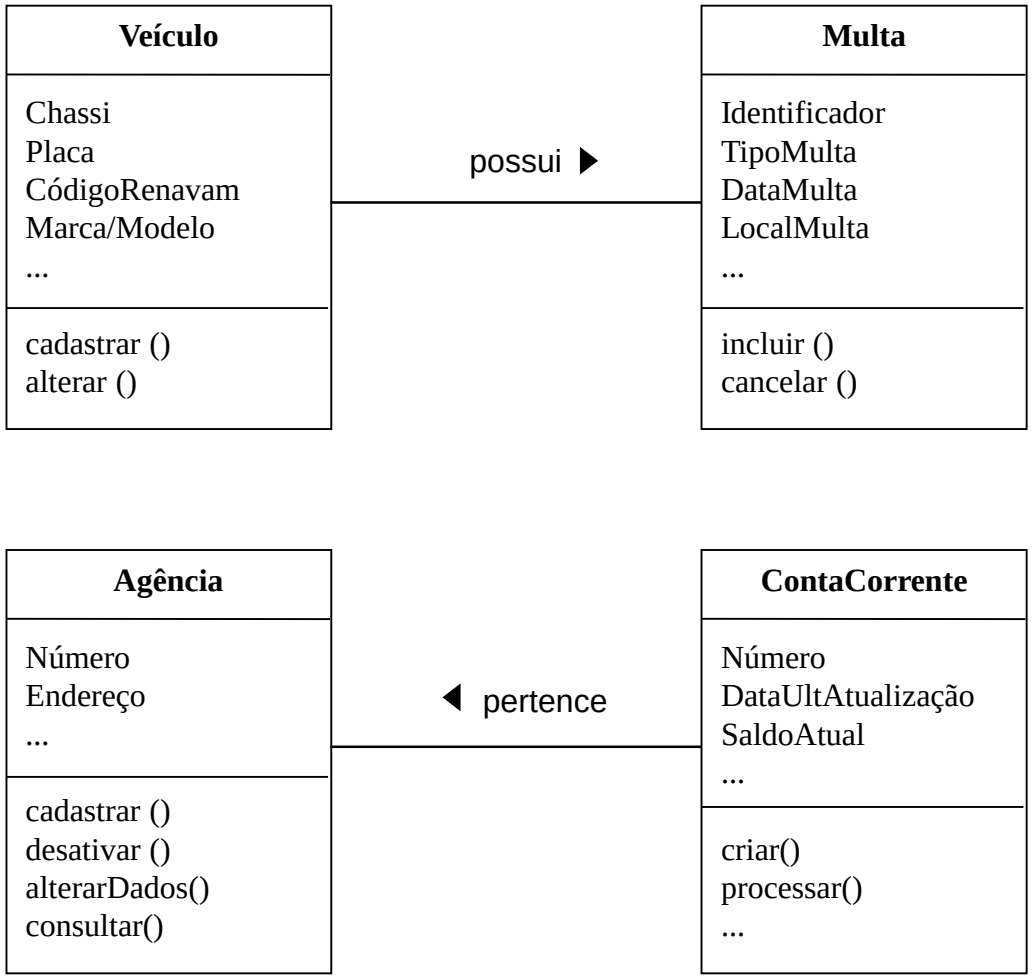
- ✓ Lista de Associações Comuns (LARMAN, 2004)
  - A é uma transação relacionada com uma outra transação B (Pagamento-Venda, Reserva-Cancelamento)
  - A é adjacente a B (Linha de Item de Venda e Linha de Item de Venda, Cidade e Cidade)
  - A é possuído por B (Registro-Loja, Avião-LinhaAérea)
  - A é um evento relacionado com B (Venda-Cliente, Venda-Loja, Partida-Vôo)

Associação (Exemplo)



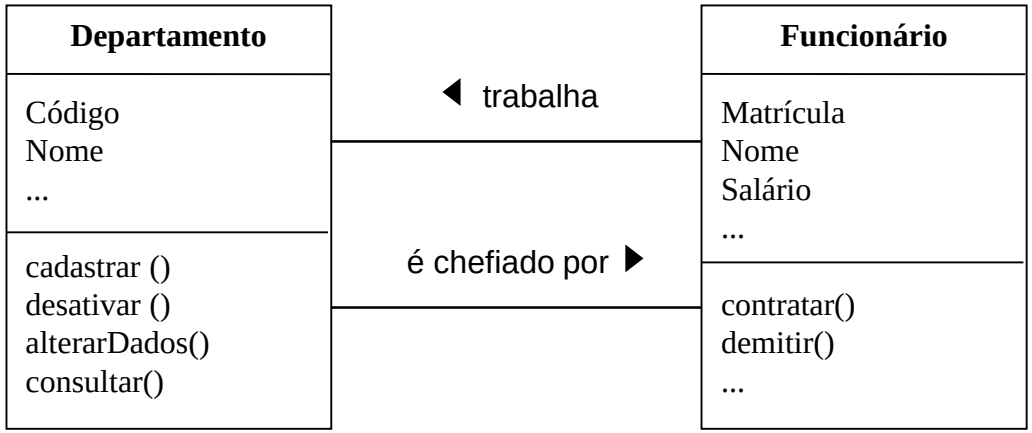
Exemplo extraído de BEZERRA, 2002

Associação (Outros exemplos)



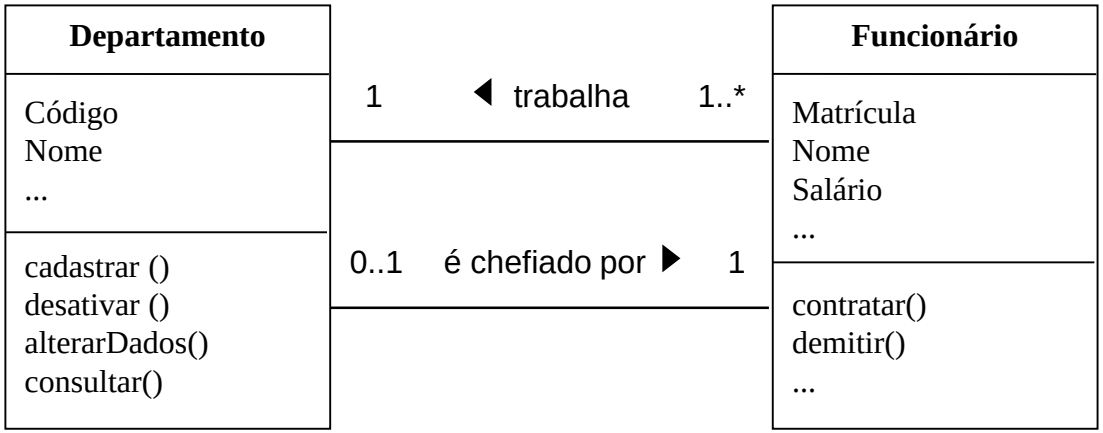
Associação (um caso especial)

- ✓ Podem existir mais de uma associação entre duas classes

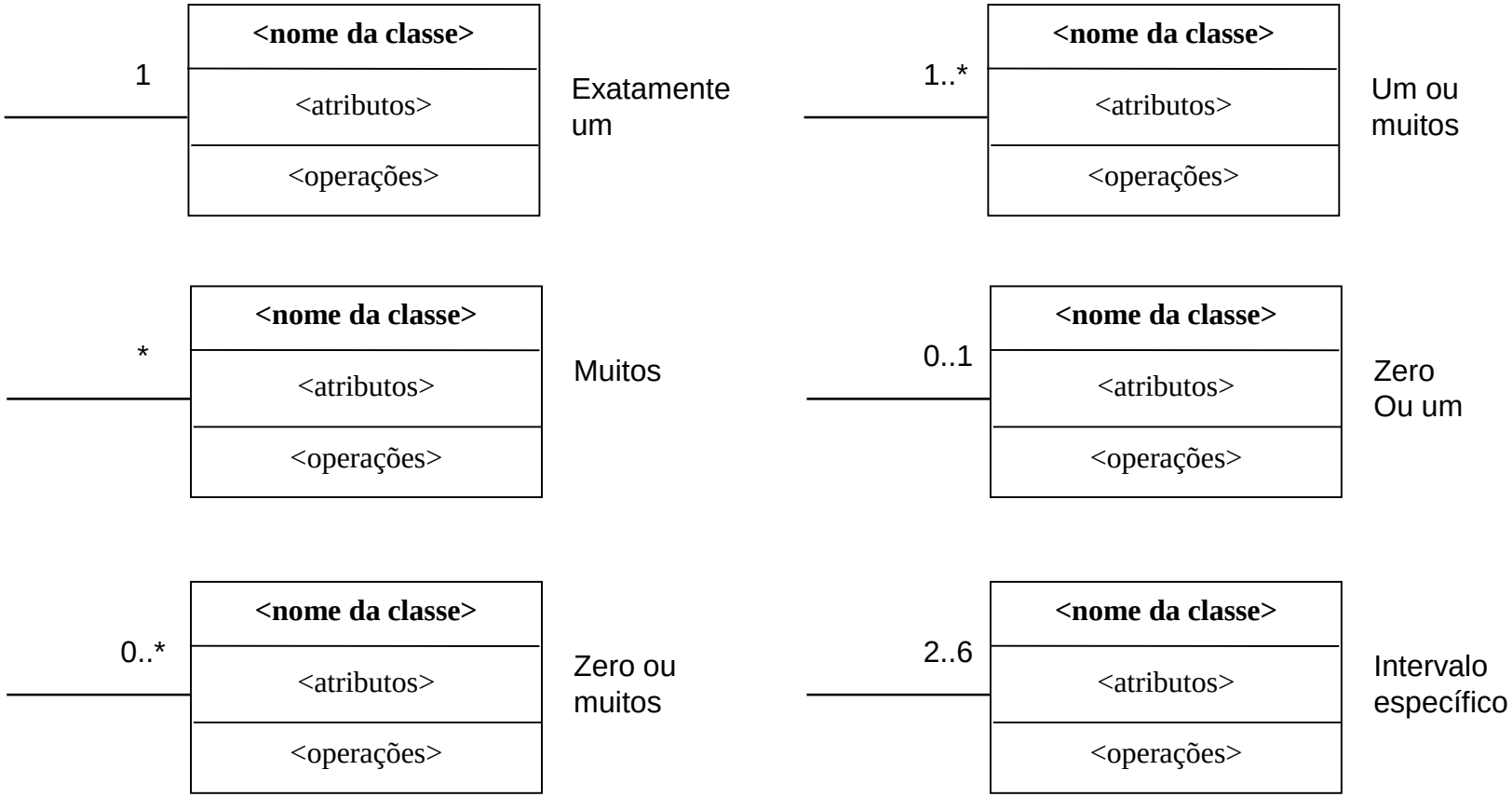


Multiplicidade da Associação

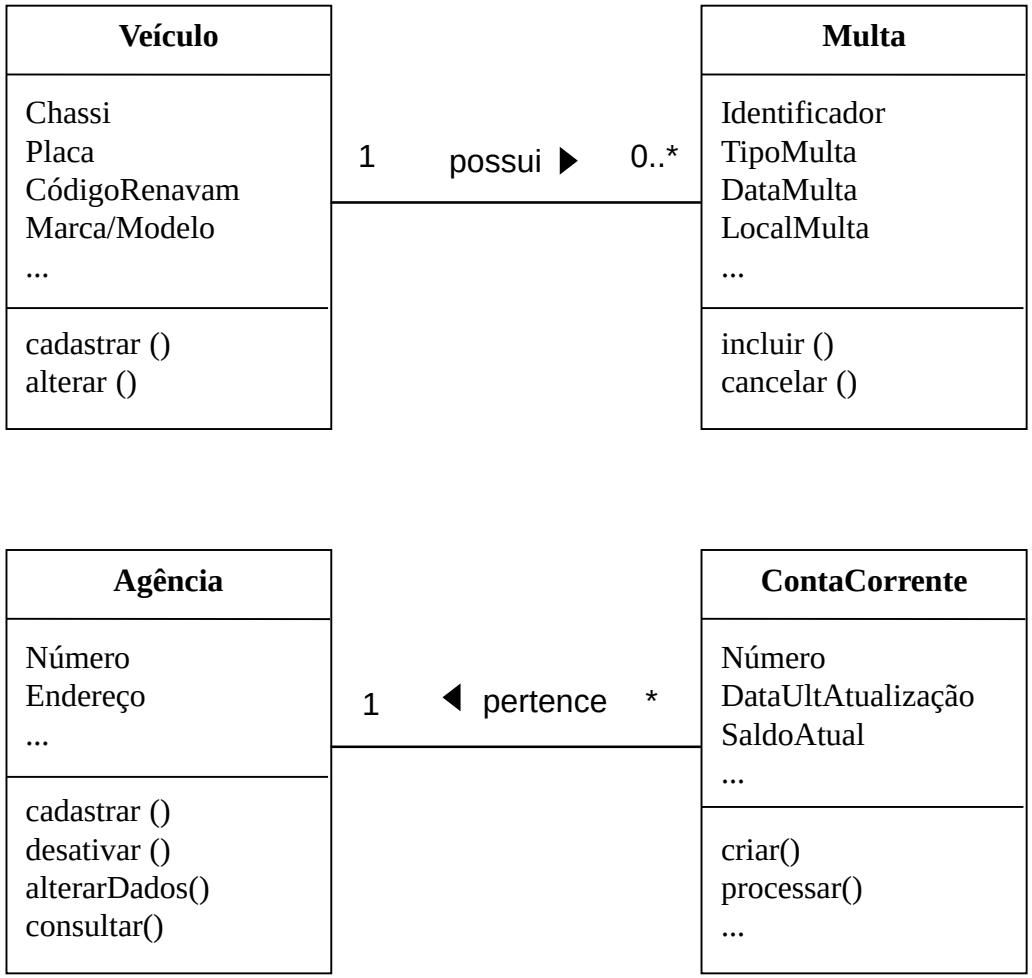
- ✓ Determina a quantidade de objetos de uma classe que pode estar relacionada com objetos da outra classe da associação e vice-versa



Indicadores de Multiplicidade



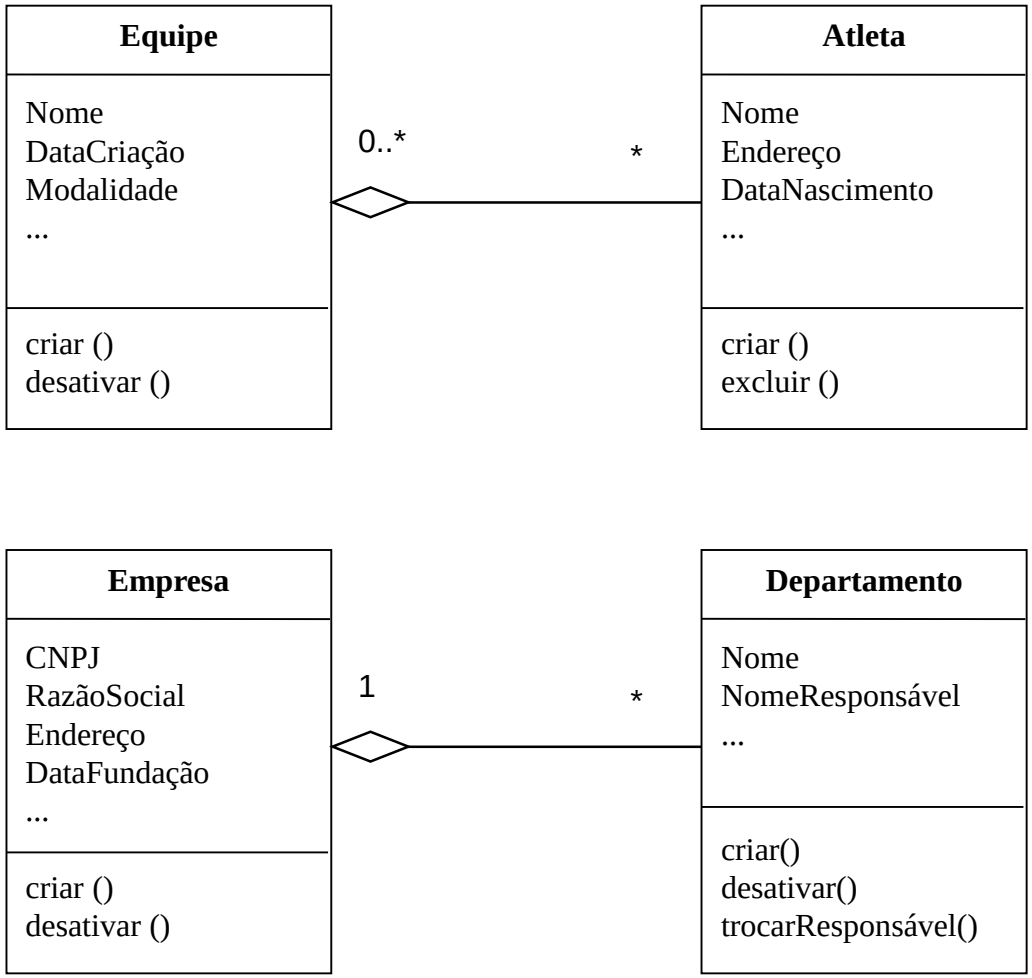
Associação (Exemplos)



Agregação

- ✓ “... representa um relacionamento do tipo ´tem-um´, o que significa que um objeto do todo contém os objetos das partes. A agregação, na verdade, é apenas um tipo especial de associação, especificada utilizando-se uma associação simples com um diamante aberto na extremidade do todo ...” (*BOOCH et al., 2000 p. 66*)

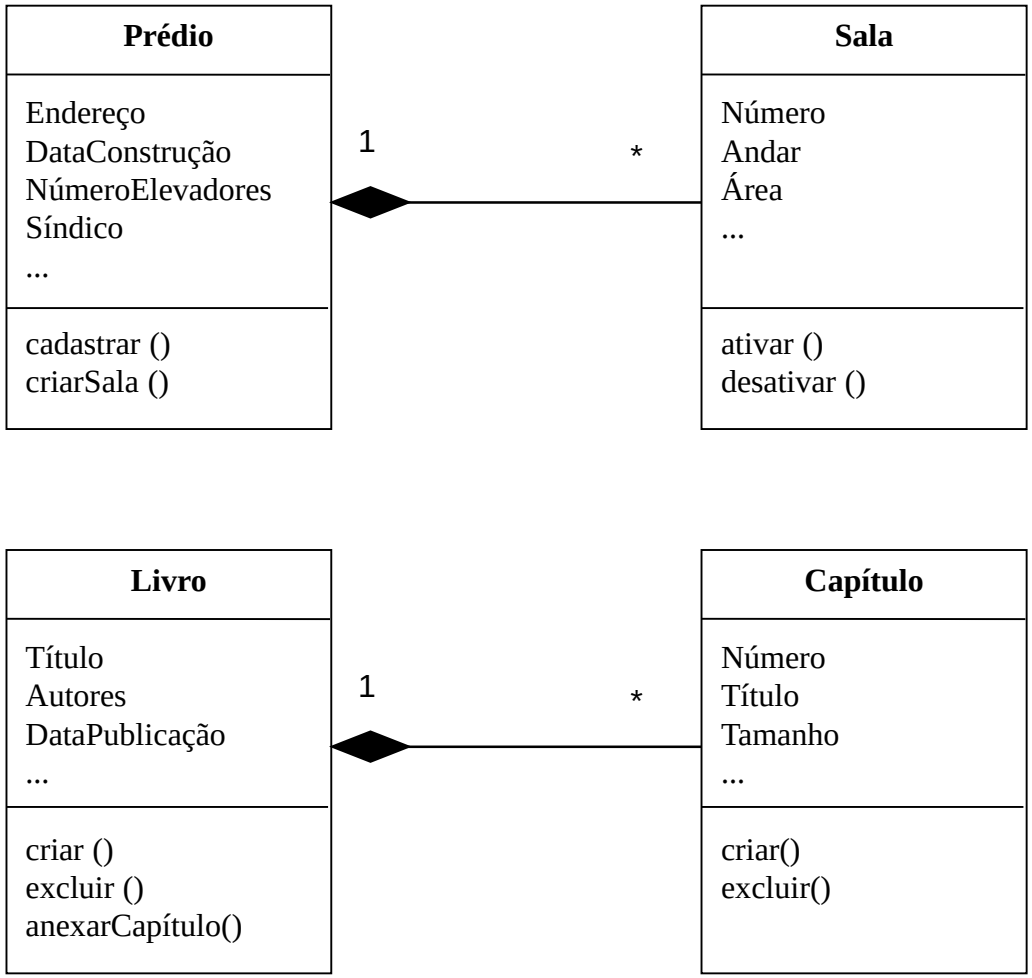
Agregação (Exemplos)



Composição

- ✓ “... é uma forma de agregação, com propriedade bem definida e tempo de vida coincidente como parte do todo. As partes sem multiplicidade fixada podem ser criadas após a composição, mas, uma vez criadas, vivem e morrem com ela. Essas partes também podem ser removidas explicitamente antes da morte do objeto composto.” (*BOOCH et al., 2000 p. 147*)

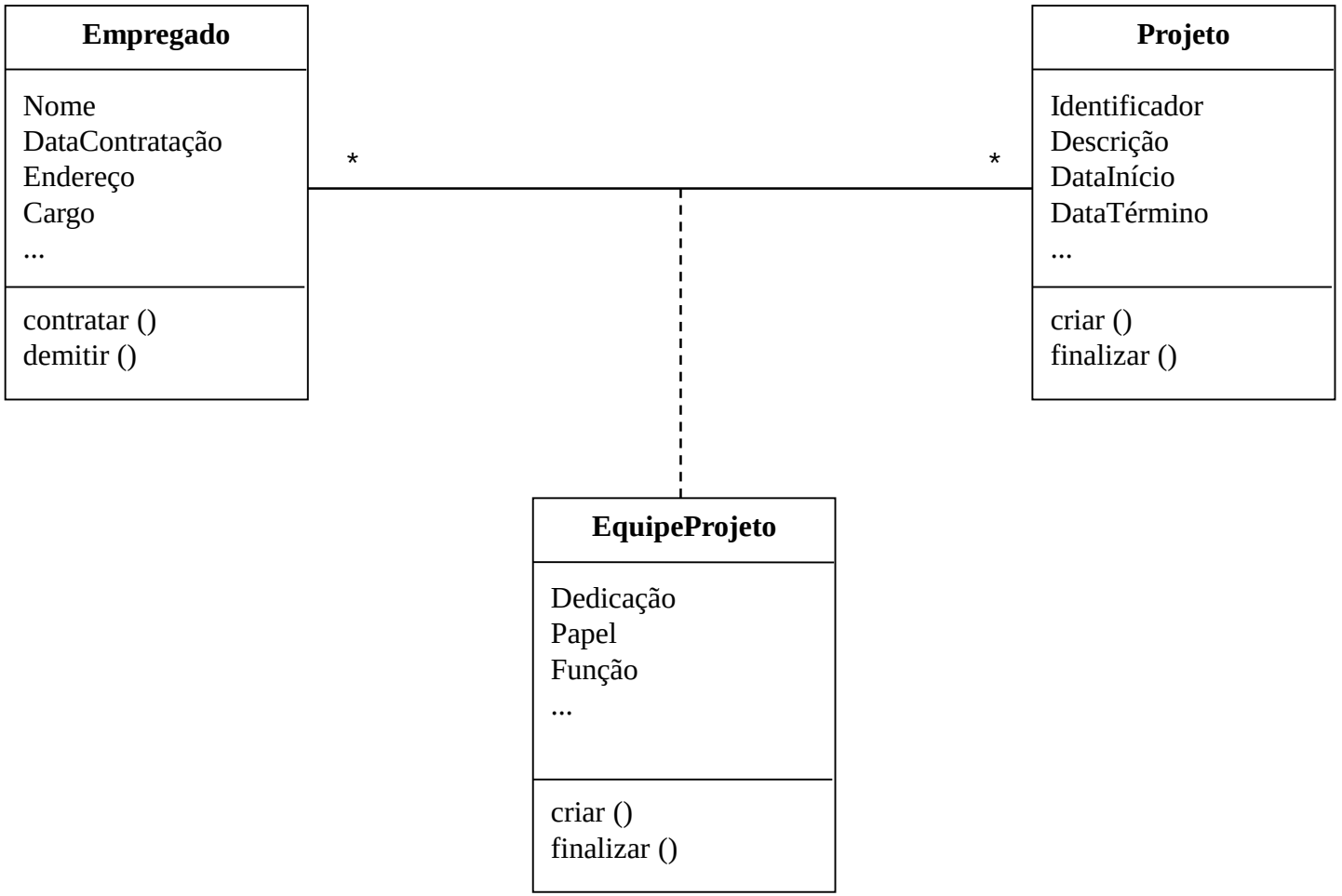
Composição (Exemplos)



Classe de Associação

- ✓ “Uma classe de associação pode ser vista como uma associação que também tem propriedade de classe ou como uma classe que também tem propriedades de associação.” (*BOOCH et al., 2000 p. 148*)

Classe de Associação (Exemplo)

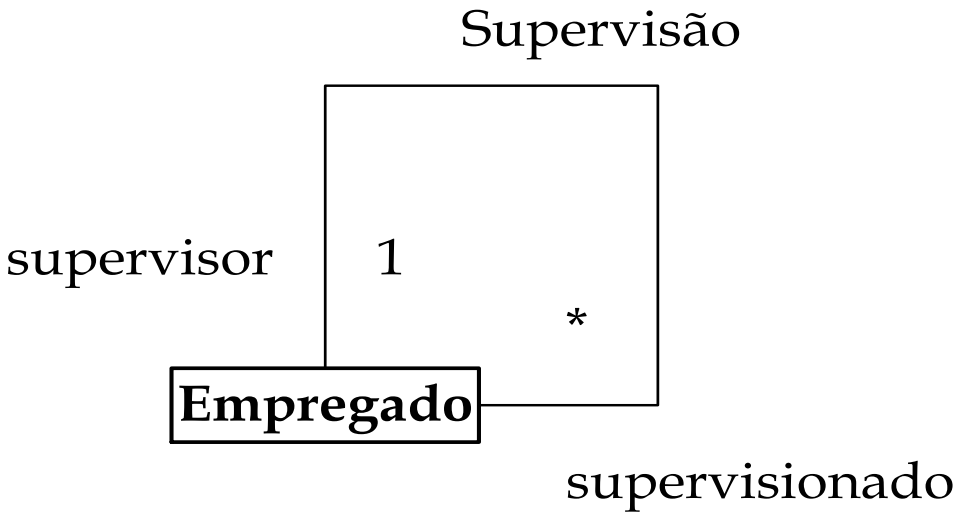


Associações reflexivas

- ✓ Associa objetos da mesma classe.
  - Cada objeto tem um papel distinto na associação.
- ✓ A utilização de papéis é bastante importante para evitar ambigüidades na leitura da associação.
- ✓ Uma associação reflexiva *não* indica que um objeto se associa com ele próprio.
  - Ao contrário, indica que objetos de uma mesma classe se associam

Transparência retirada de (BEZERRA, 2002)

Associação reflexiva (Exemplo)



Exemplo extraído de BEZERRA, 2002

Navegação

- ✓ “A menos que seja especificado o contrário, a navegação por uma associação é bidirecional. Entretanto, em algumas situações, você desejará limitar a navegação a uma única direção. (...) Para representar explicitamente a direção da navegação, você poderá incluir adornos na associação com setas apontando a direção a ser seguida.” (*BOOCH et al., 2000 p. 148*)

Navegação (Exemplo)