Análise de Sistemas

TEMA: DIAGRAMA DE CLASSES

PROFESSOR: ALLAN FERNANDES BALARDINO

Parte 1

Modelagem conceitual estrutural

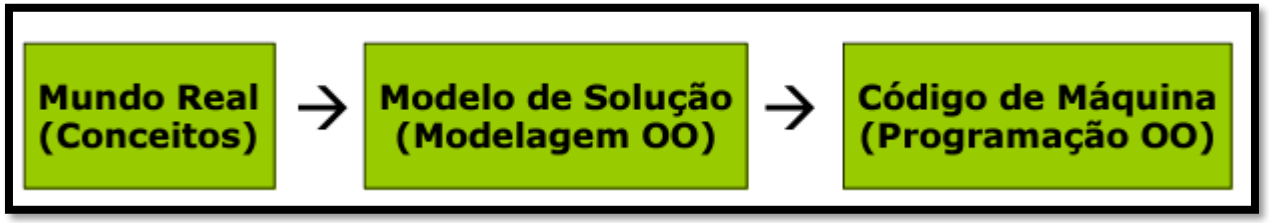
- Modelo conceitual: abstração da realidade segundo um conceito;
- Foco no domínio do problema;
- Independente da tecnologia a ser adotada na resolução;
- Entender e comunicar aspectos do domínio;

Modelagem conceitual estrutural

- Objetivo: descrever informações relevantes a serem representadas
- Entidades e relacionamentos;
- Principais modelos:
  - Diagrama de Entidade Relacionamento;
  - Diagrama de Classes UML;
- Foco em orientação a objetos: diagramas de classe;

Orientação a objetos

- Mundo é composto por objetos;



- Orientação a objetos:
  - Gerenciar a complexidade do mundo real, **abstraindo** o conhecimento relevante para o contexto e **encapsulando-o** em entidades;

Abstração

- Humanos lidam com a complexidade do mundo extraindo a parte relevante para determinado contexto, abstraindo o que é importante:
  - Mapas para representar territórios: político, relevo, temperatura...
- Quais características uma pessoa tem?
- Quais são relevantes para o cadastro de usuários em uma biblioteca?
- E para um sistema bancário?

Encapsulamento

- Separação de aspectos externos de um objeto que precisam ser acessíveis por outros objetos;
- Para dirigir um carro é necessário conhecer mínimos detalhes da mecânica?
- Para utilizar caixa bancário, precisamos conhecer o mecanismo que emite notas, extrato...?
- O que encapsular em uma determinada classe?

Classes

- Descreve uma categoria de objetos que possuem a mesma estrutura:
  - Atributos e relacionamentos;
  - Comportamentos;
- Naturalmente desenvolvemos nosso conhecimento de mundo a partir de categorias;
- Simplifica o entendimento da complexidade do mundo;
- Porque quando vejo um leão sei que estou em perigo mesmo sem nunca tê-lo visto antes?

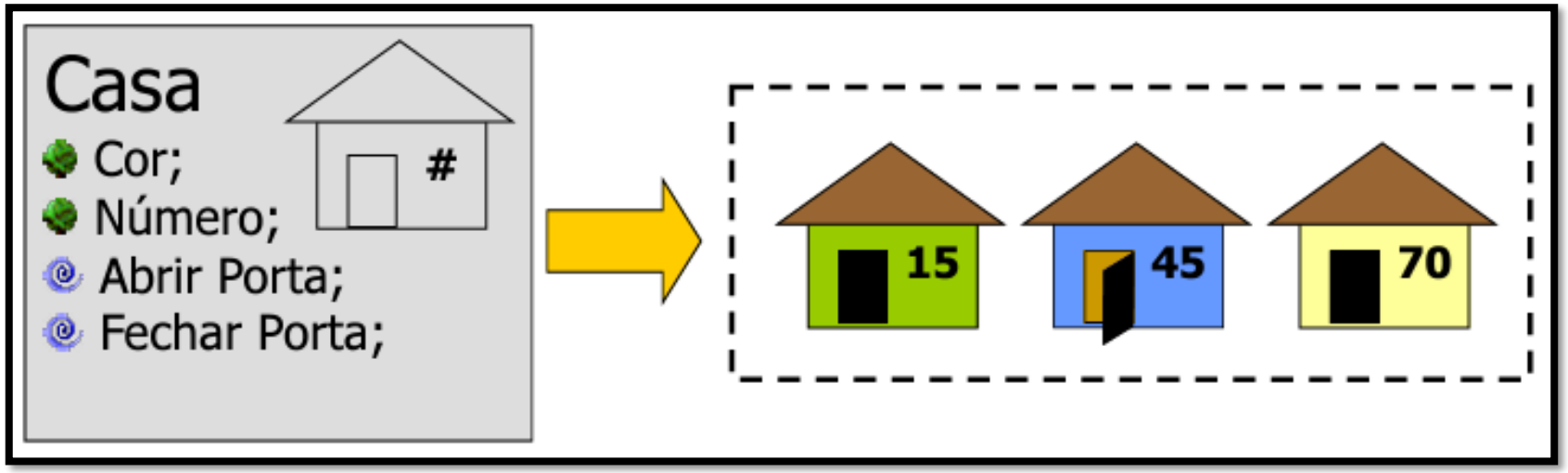
Objetos

- Materialização da classe;
- Elementos que são criados durante o funcionamento do sistema;
- Classe pessoa:
  - Objetos:
    - João;
    - Maria;
    - José

Classes e instâncias

- Objeto: instância da classe;
- Orientação a objetos:
  - Objetos representam entidades reais;
  - Executam algum papel no sistema;
- Classes: abstrações que capturam comportamento comum a um conjunto de objetos;
- Modelamos classes, não objetos;
- Tarefa importante na análise de OO: quais classes devem ser incluídas no modelo?

Classes e instâncias



Classes e instâncias

- Nome das classes:
  - Sem acentos ou espaçamento;
  - Nome compostos iniciando com Letra maiúscula;
  - Sempre no singular;

- Exemplos:
  - PessoaFisica, Cliente, PedidoDeCompra, NotaFiscal;

Possíveis candidatos a classe

- Elementos que são parte do domínio do problema:
  - Pessoas, clientes, turmas, produtos, etc.
- Papéis desempenhados por pessoas no sistema:
  - Operador, gerente, administrador...
- Unidades organizacionais:
  - Setor, departamento, local;

Critérios para seleção de classes

Direções para se levar em consideração ao definir classes:

- Reter informação úteis:
  - Foco no objetivo do sistema;
- Ter atributos e operações comuns a todas suas instâncias;
- Avaliar se realmente existe a necessidade de ser tratado como classe (somente um atributo ou relação) ou se pode apenas ser uma propriedade;
  - E-mail / Usuario;

Para refletir

No contexto de um sistema de biblioteca, onde pessoas de uma instituição podem consultar ao acervo e pegar livros emprestados;

- Quais classes seriam de interesse do sistema?
  - Eventos?
    - Empréstimo;
  - Papéis?
    - Usuário;
    - Funcionário;
  - Locais?
    - Setores;
  - Objetos físicos?
  - Livros, Revistas, Mapas...

Atributos

- O que a classe possui?
- Todo objeto que seja deste tipo deverá possuir;
- Exemplo:
  - Classe Carro: Montadora, modelo, quantidade de portas, tipo de combustível... O que será relevante?
- Tipos de dados:
  - Primitivos: String, Boolean, integer, date...
  - Específicos de domínio: CPF, ISBN, Endereço.
  - Enumerações: DiaDaSemana {Seg, Ter, ..., Dom}
- Padrão de nomes semelhante ao de classe, com separação entre nomes compostos podendo também ser pelo caracter underline (_);

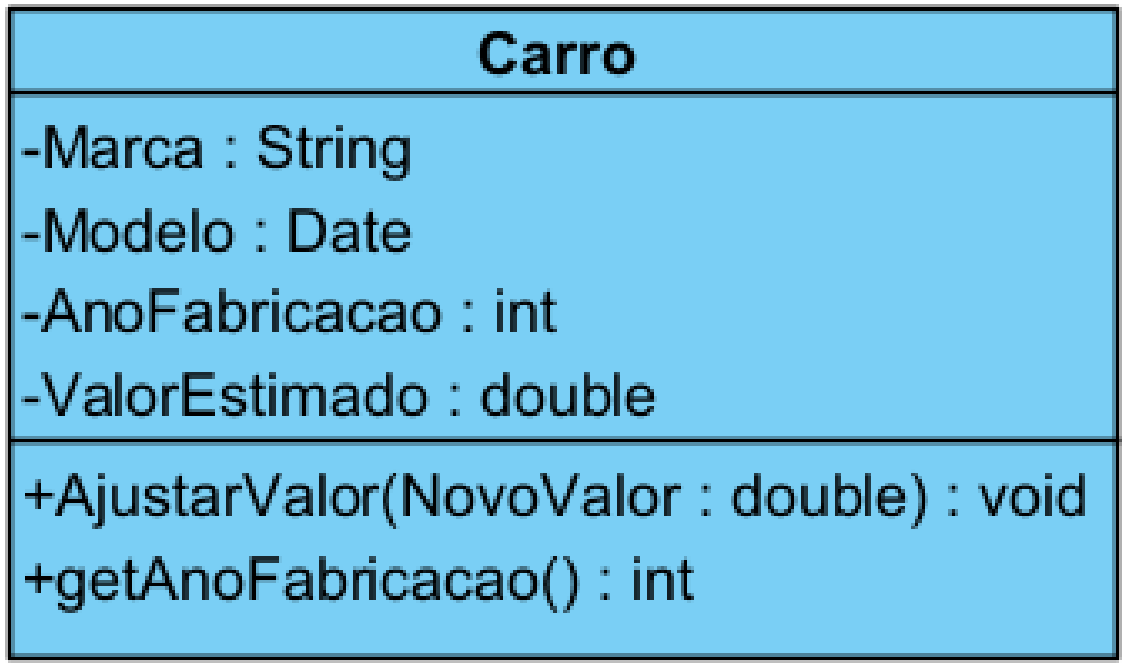
Operações (métodos)

- O que a classe faz?
- Ações que todo objeto deste tipo deverá ser possível fazer;
- Exemplo:
  - Classe Carro: o que um carro faz? Modelar o que for interessante para o contexto;
  - Acelera, freia, abre porta, ligar...
- Podem receber parâmetros de entrada;
- Padrão de nomes semelhante também semelhante de atributo;

Representação em UML

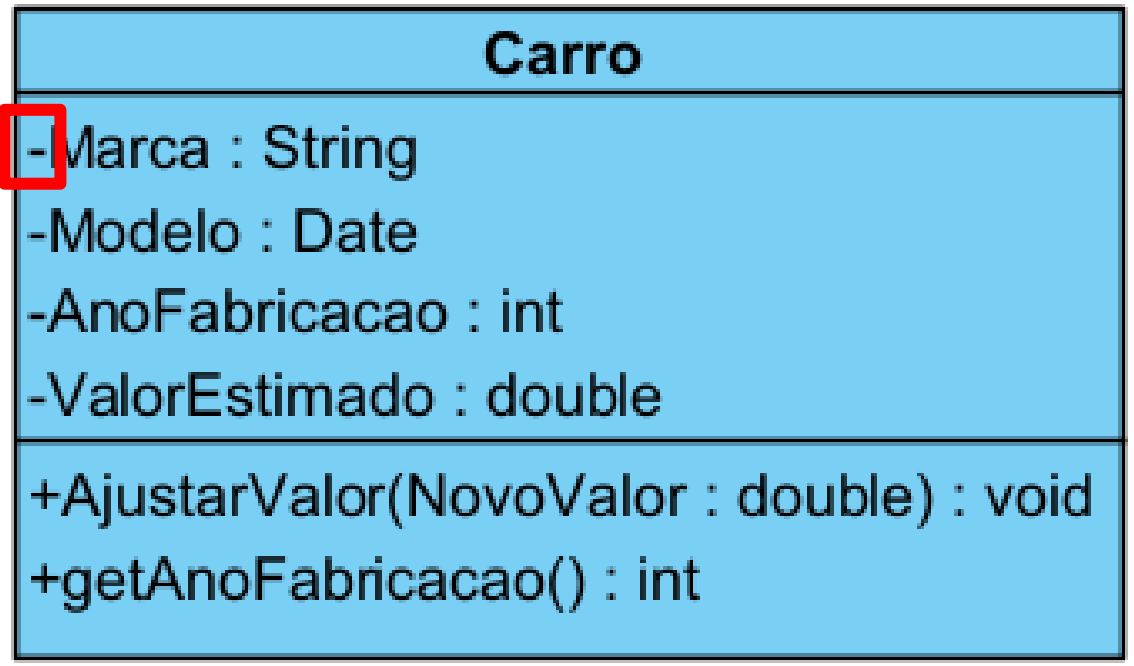
- UML*: Unified Modeling Language*;
- Padroniza documentações gerais no projeto de sistemas (engenharia de software);
- Diagrama de classes, definindo:
  - Nome da classe;
  - Atributos (tipo, visibilidade);
  - Métodos: tipo de retorno, visibilidade, parâmetros (se houver));

Representação em UML

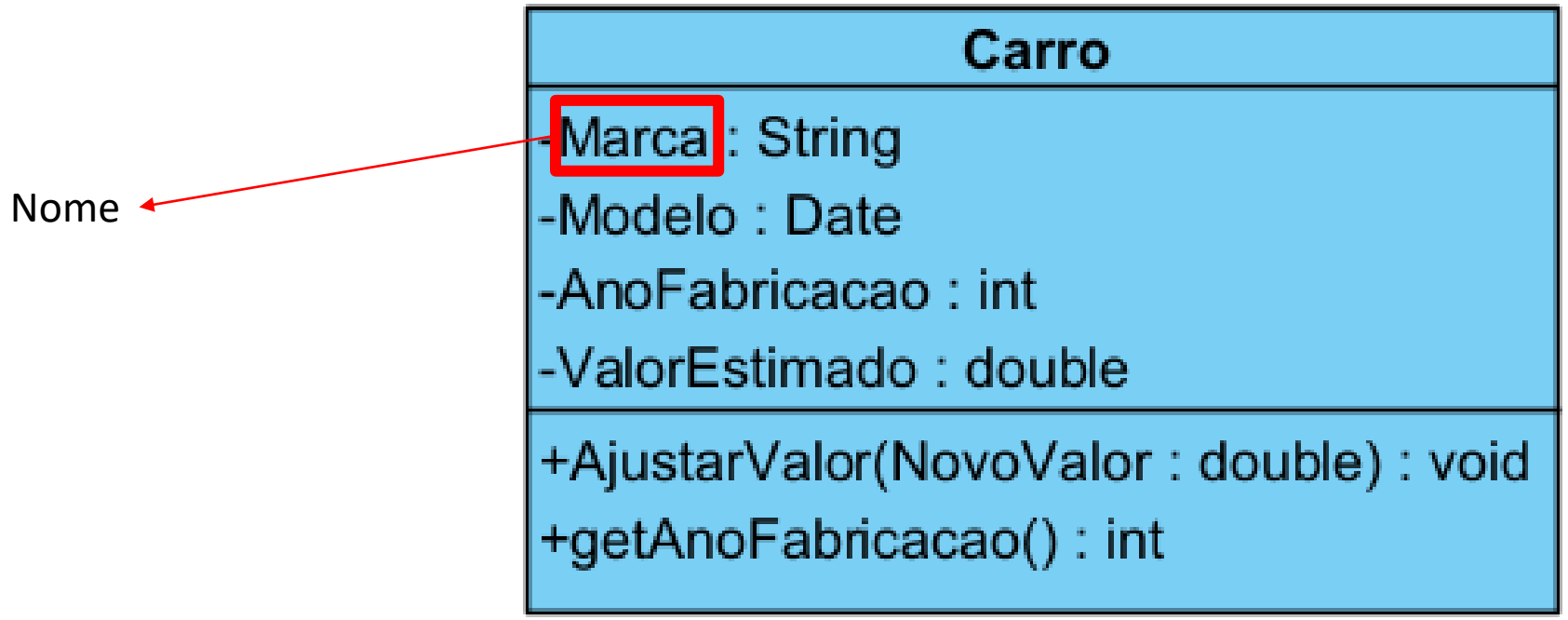


Representação em UML

Visibilidade:
+ Publico
- Privado

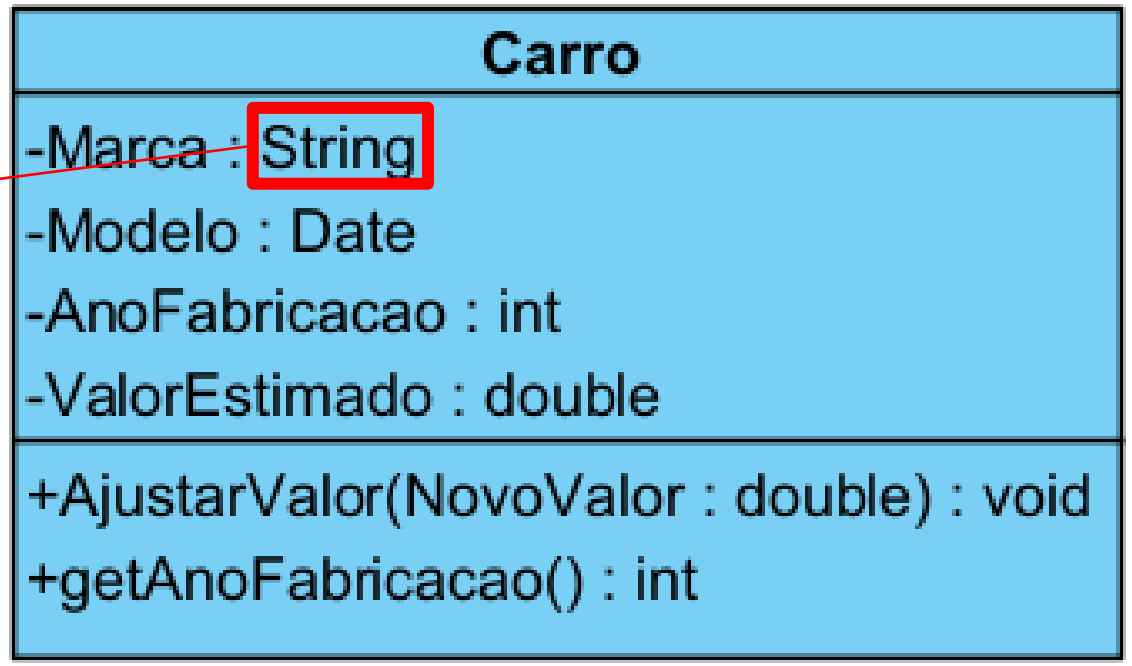


Representação em UML



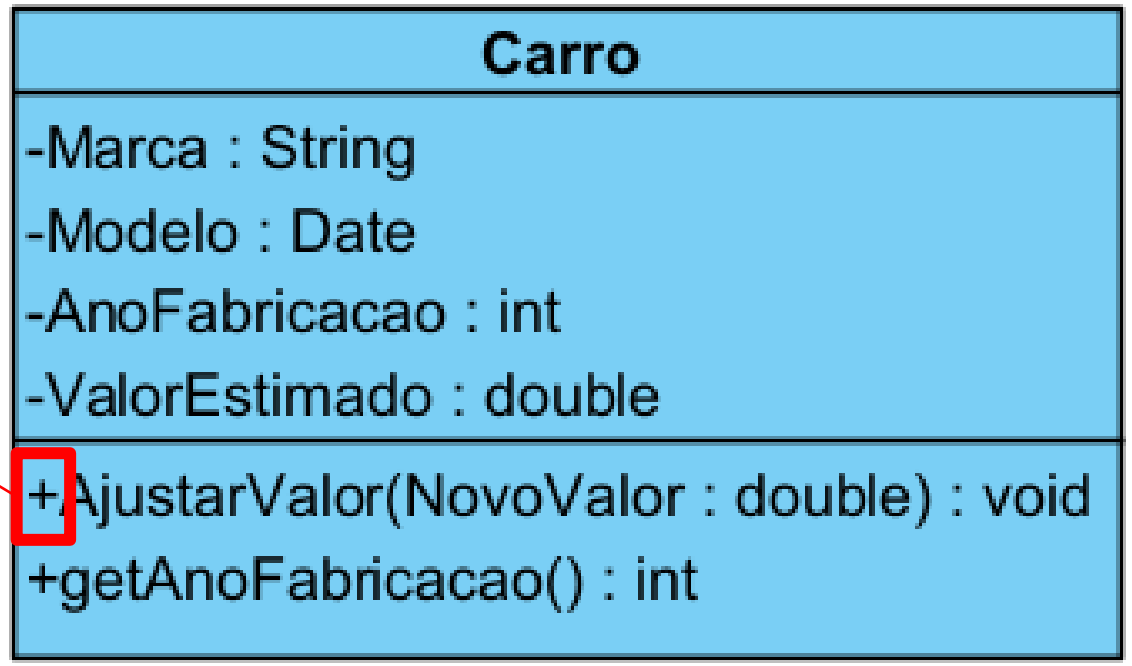
Representação em UML

Tipo atributo



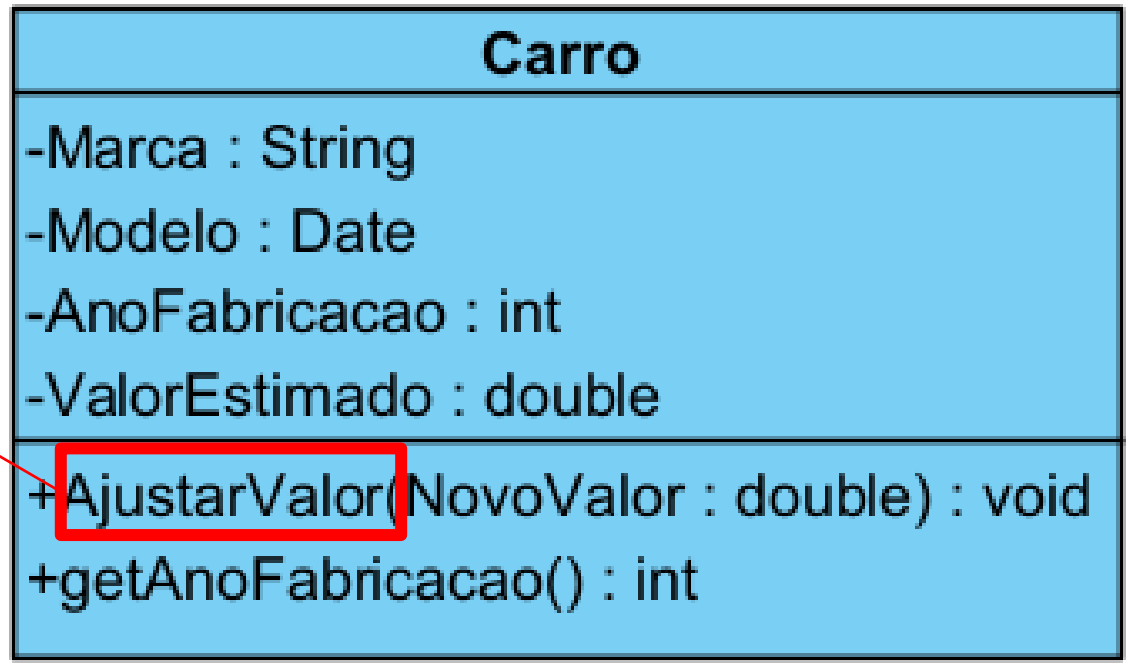
Representação em UML

Visibilidade:
+ Publico
- Privado



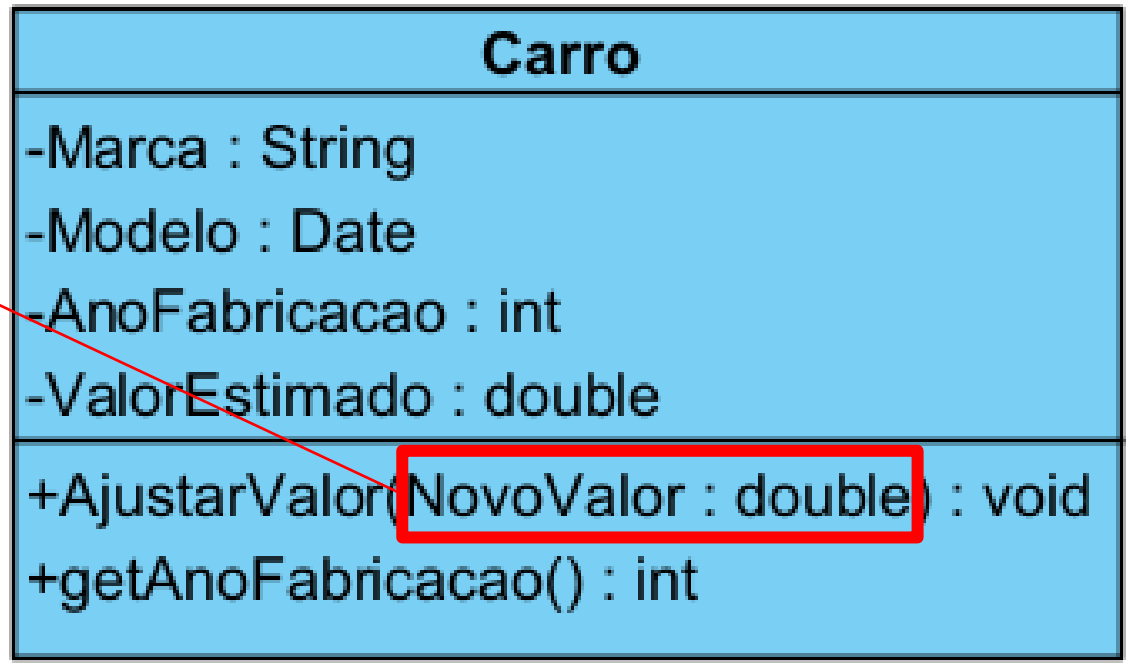
Representação em UML

Nome

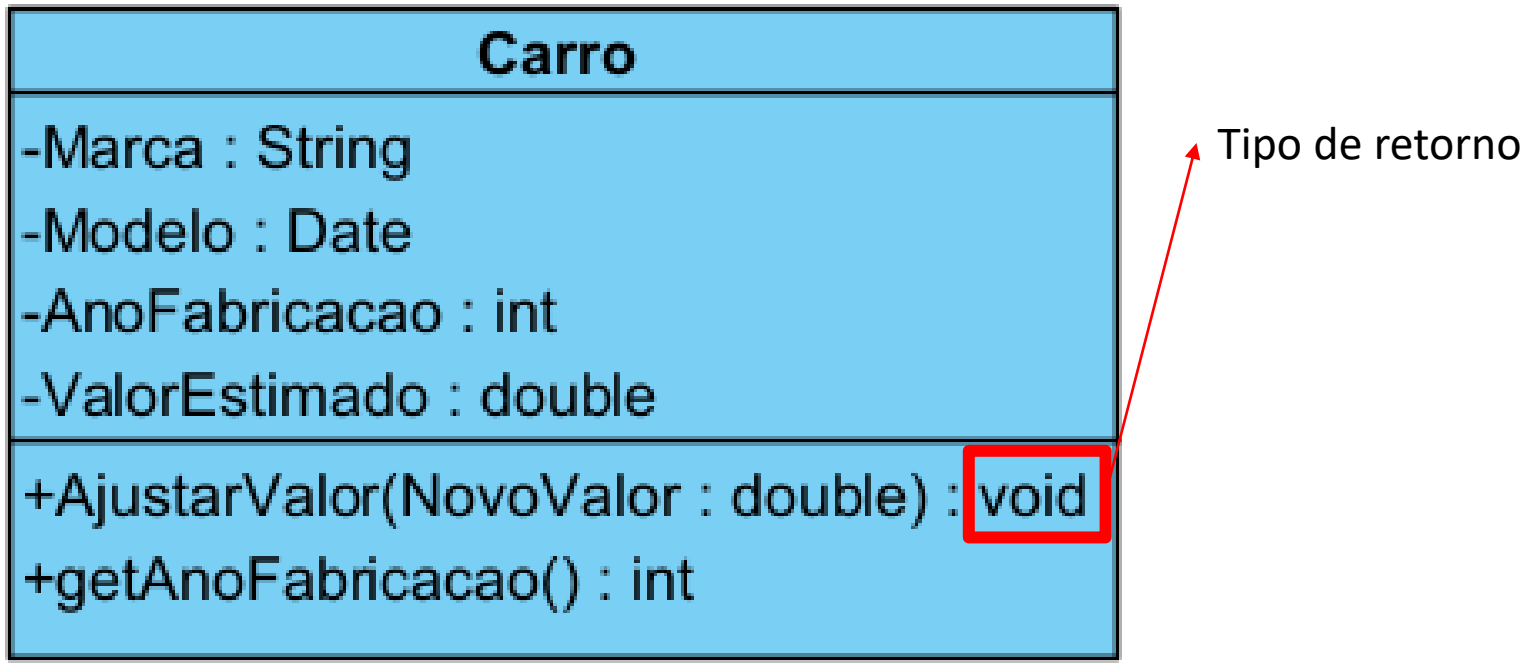


Representação em UML

Nome e tipo de parâmetro(s)



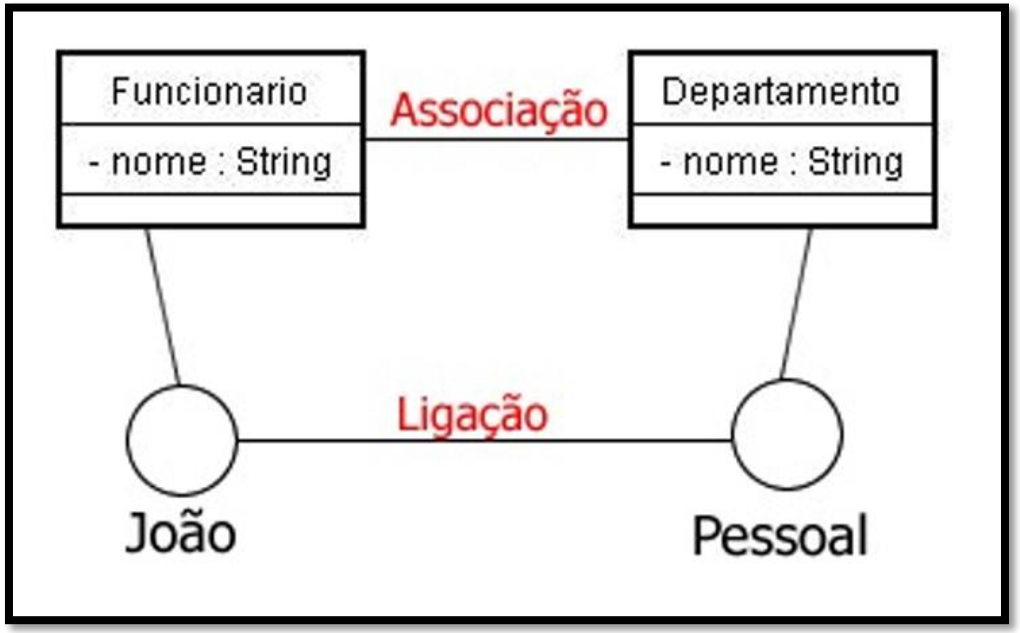
Representação em UML



Ligações e associações

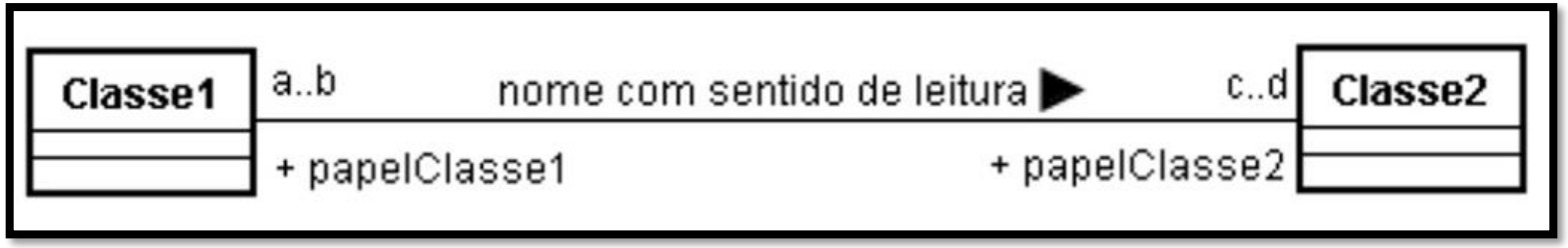
- Associações: conexão entre classes que representa a existência de relação entre objetos;
  - Potenciais conexões entre classes que também são modeladas em UML;
- Ligação: conexão entre objetos;
  - Modelamos classes e suas associações... ligações entre objetos é consequência!

Ligações e associações



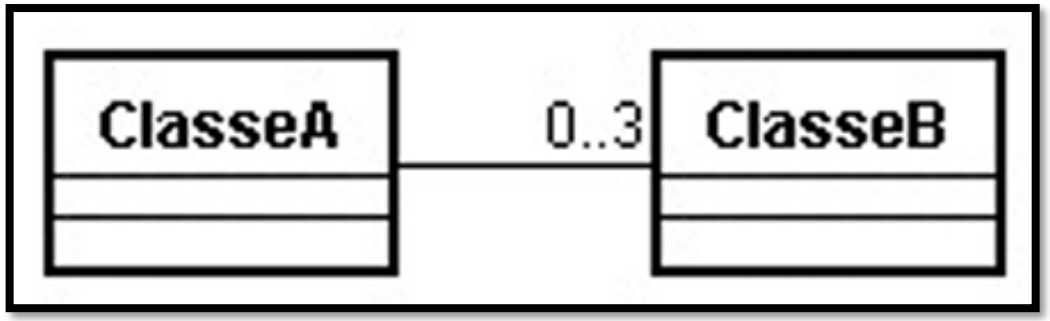
Associações

- Possui um nome específico;
- Papéis: indicam “o que” a classe desempenha em determinada associação;
- Cardinalidades/Multiplicidades: quantidade de objetos que podem estar envolvidos na associação;



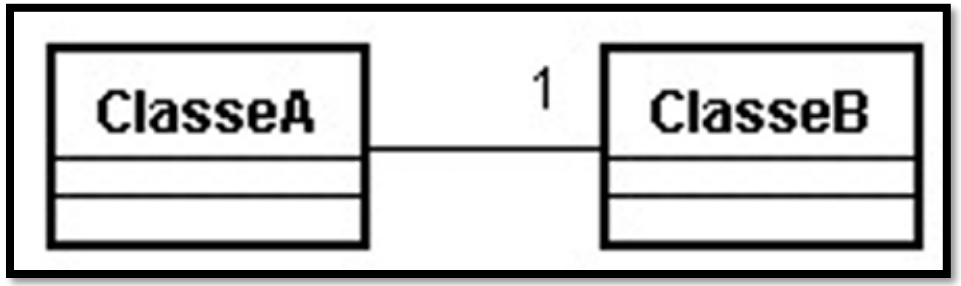
Cardinalidades

- Um objeto da ClasseA pode se relacionar com no mínimo 0 e no máximo 3 objetos da ClasseB:



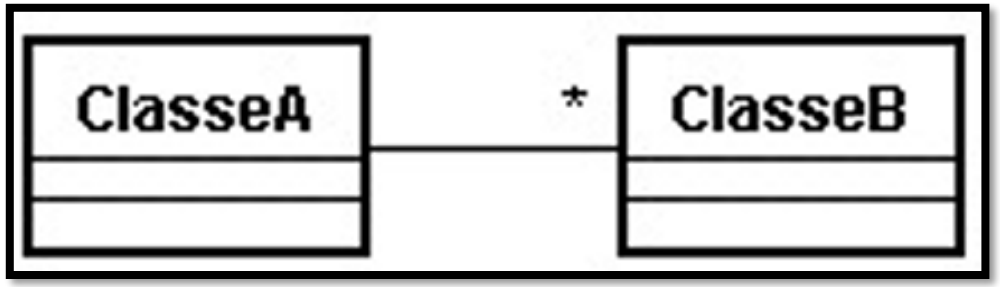
Cardinalidades

- Um objeto da ClasseA deverá se relacionar, necessariamente, com um objeto da ClasseB:



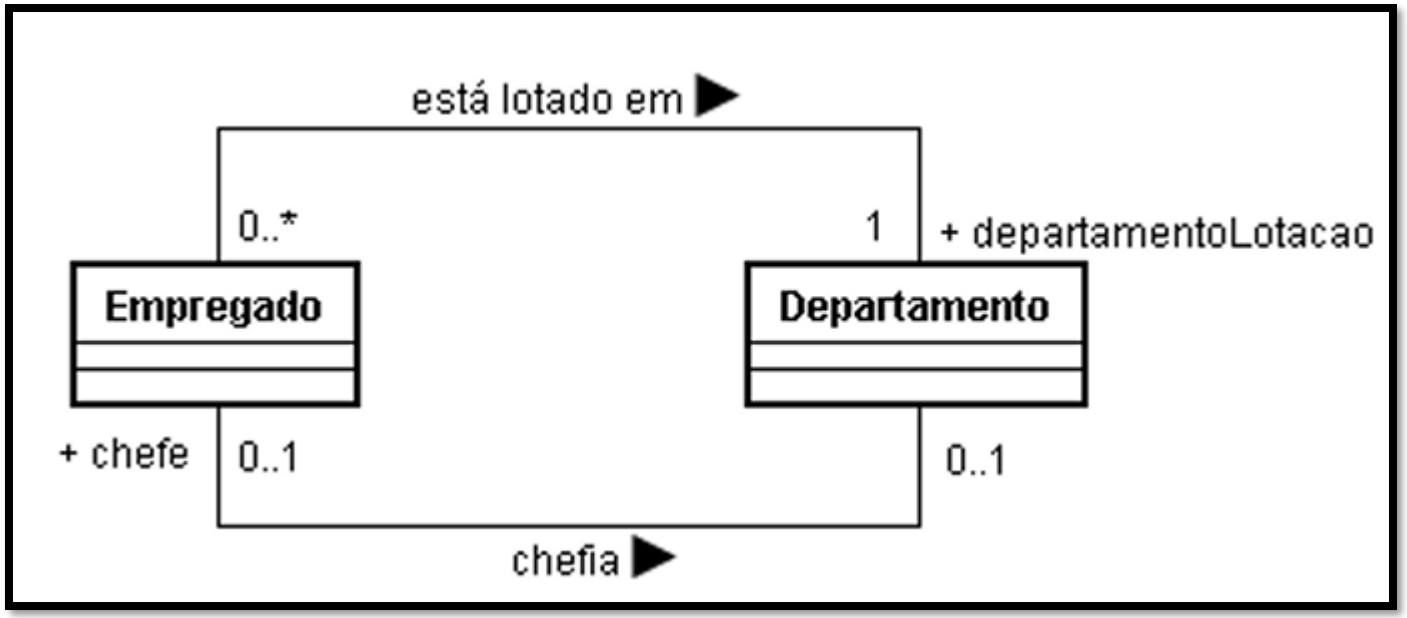
Cardinalidades

- Um objeto da ClasseA poderá se relacionar com nenhum, um ou vários objetos da ClasseB:



Papéis

- Descrição adicional da relação;



Exemplo

- Sistema para loja de departamentos:

  O sistema visa organizar os produtos em departamentos, bem como definir os vendedores que atuam em cada departamento.

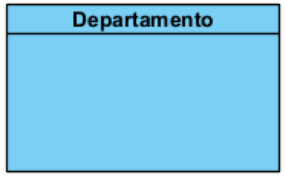
  Cada departamento tem um nome. A partir de cada departamento deve ser possível calcular o valor total de suas vendas.

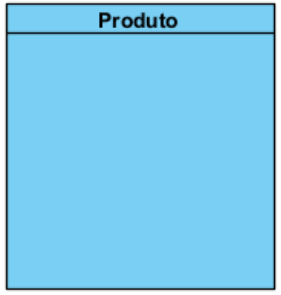
  De cada produto sabe-se o código, o nome, a descrição e o preço. Um mesmo produto pode ser exposto em um departamento e um departamento expõe um ou mais produtos. A partir de um produto, deve ser possível atualizar seu preço.

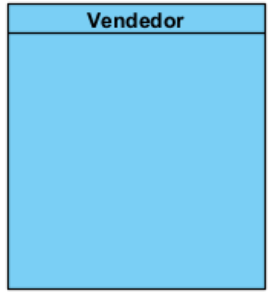
  Sobre cada vendedor sabe-se o nome, a matrícula, o CPF, Salário base e percentual de comissão. Deve ser possível, a partir de um vendedor, atualizar seu salário base. Cada vendedor é alocado em um departamento e um mesmo departamento tem vários vendedores alocados. Além disso, cada departamento tem um vendedor responsável.

Exemplo

- Classes:
  - Departamento;
  - Produto;
  - Vendedor;

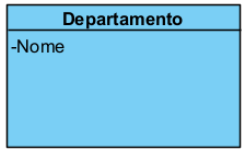


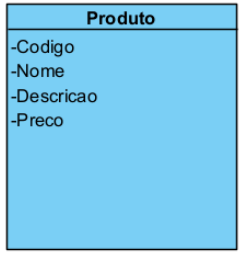




Exemplo

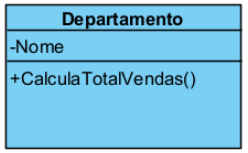
- Classes e atributos:
  - Departamento:
    - Nome;
  - Produto:
    - Código, Nome, Descrição, Preço;
  - Vendedor:
    - Nome, Matrícula, CPF, Salário base, Percentual comissão;

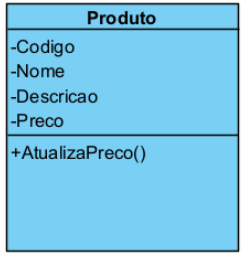


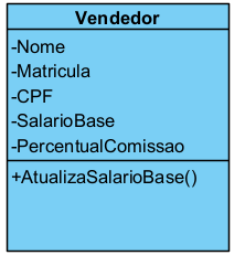


Exemplo

- Classes, atributos e métodos:
  - Departamento:
    - Nome;
    - Calcula total de vendas;
  - Produto:
    - Código, Nome, Descrição, Preço;
    - Atualiza preço;
  - Vendedor:
    - Nome, Matrícula, CPF, Salário base, Percentual comissão;
    - Atualiza salário base;

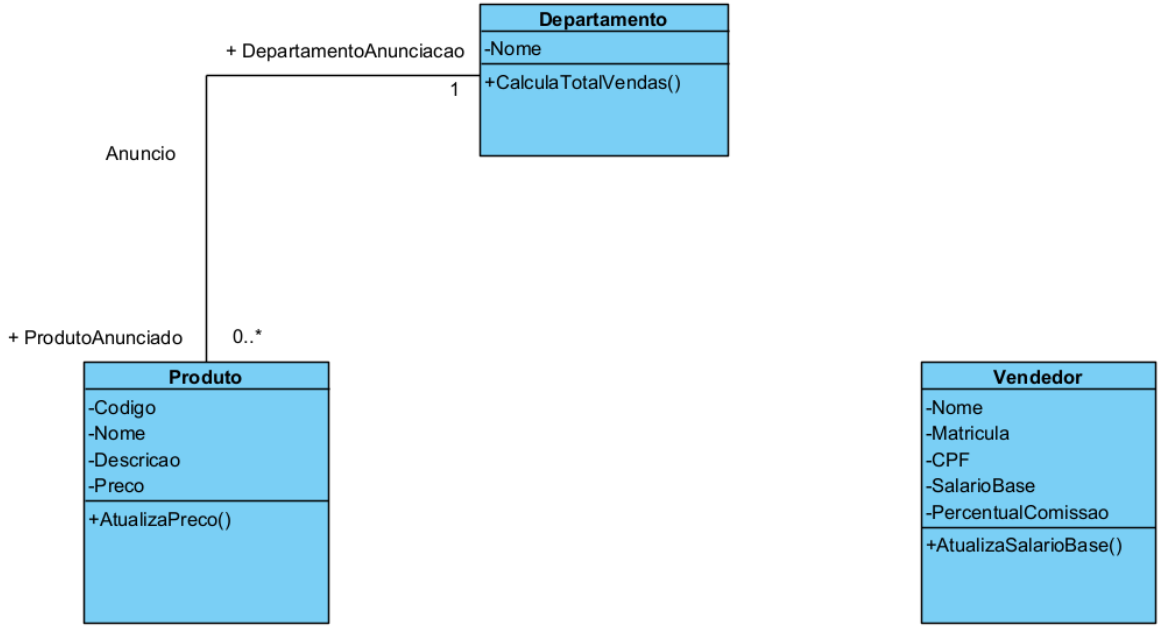






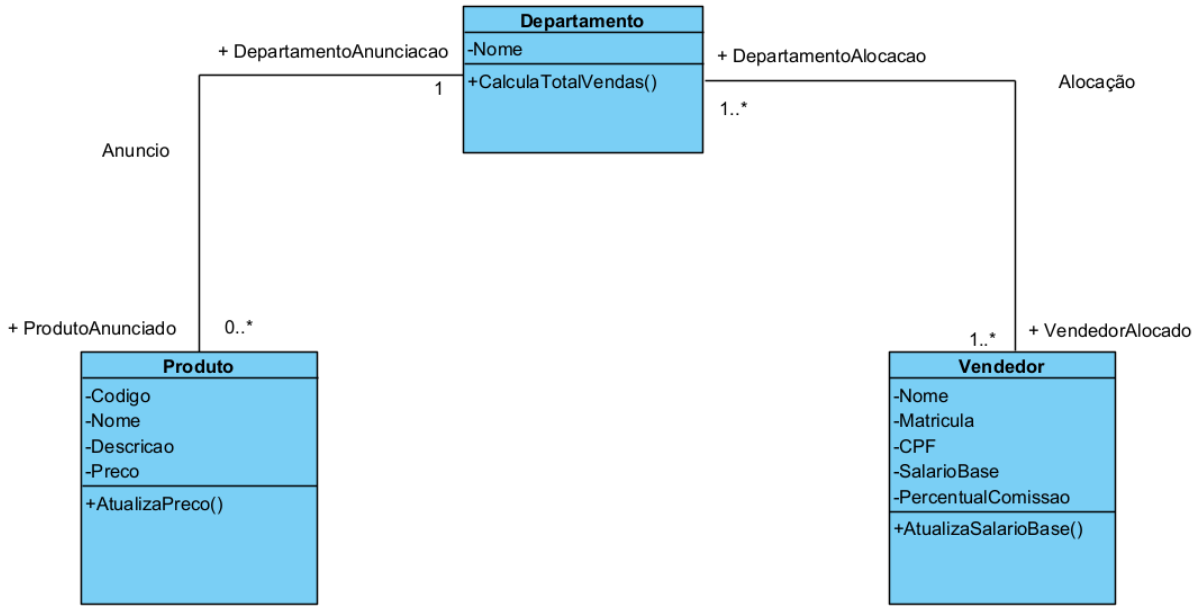
Exemplo

- Associações:
  - Anuncio:
    - Produto é anunciado em apenas um departamento (1). Um Departamento pode ter nenhum, um, ou vários Produtos anunciados (0..*):



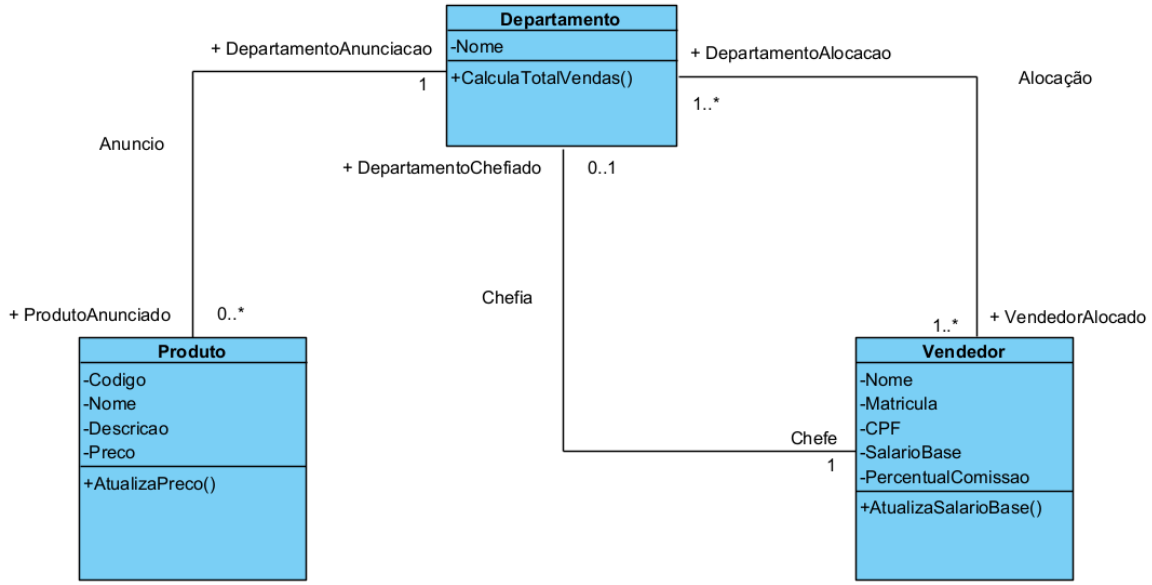
Exemplo

- Associações:
  - Alocação:
    - Vendedor é alocado em um ou mais Departamentos (1..*). Um Departamento pode ter um ou vários Vendedores alocados (1..*):

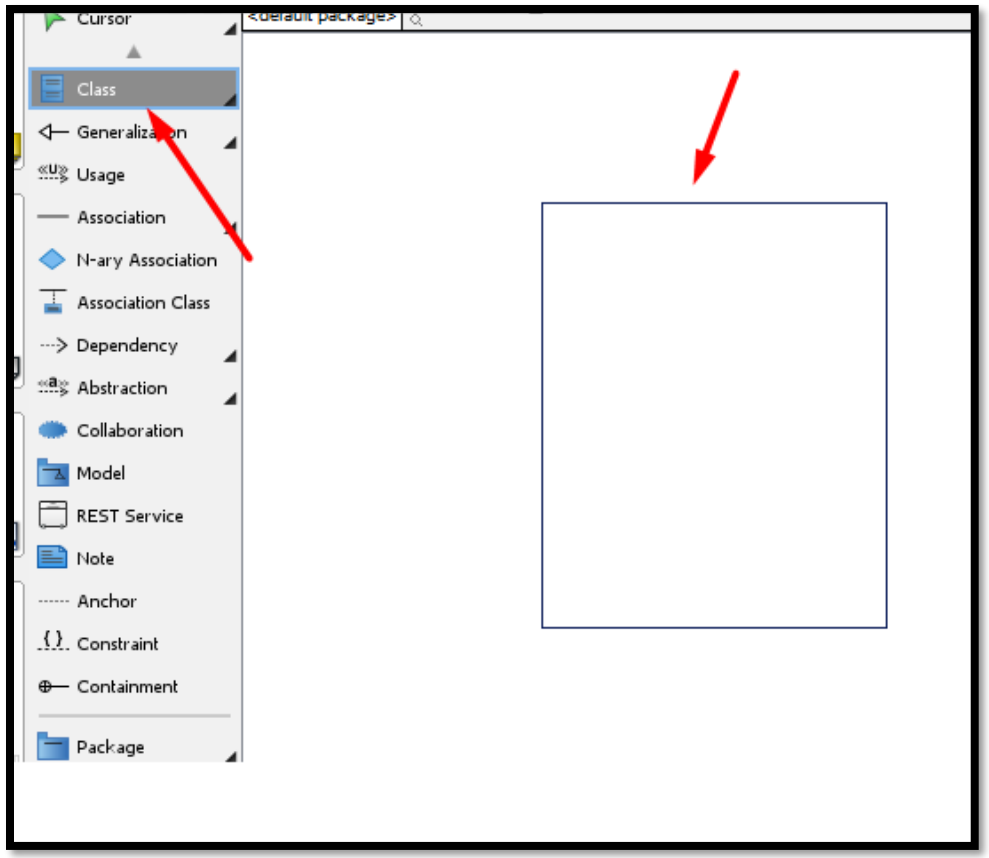


Exemplo

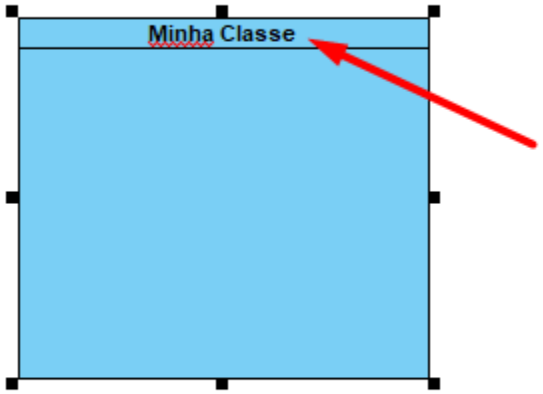
- Associações:
  - Responsável:
    - Um vendedor pode ser responsável por nenhum ou um departamento (0..1). Um departamento terá como responsável apenas um vendedor (1):



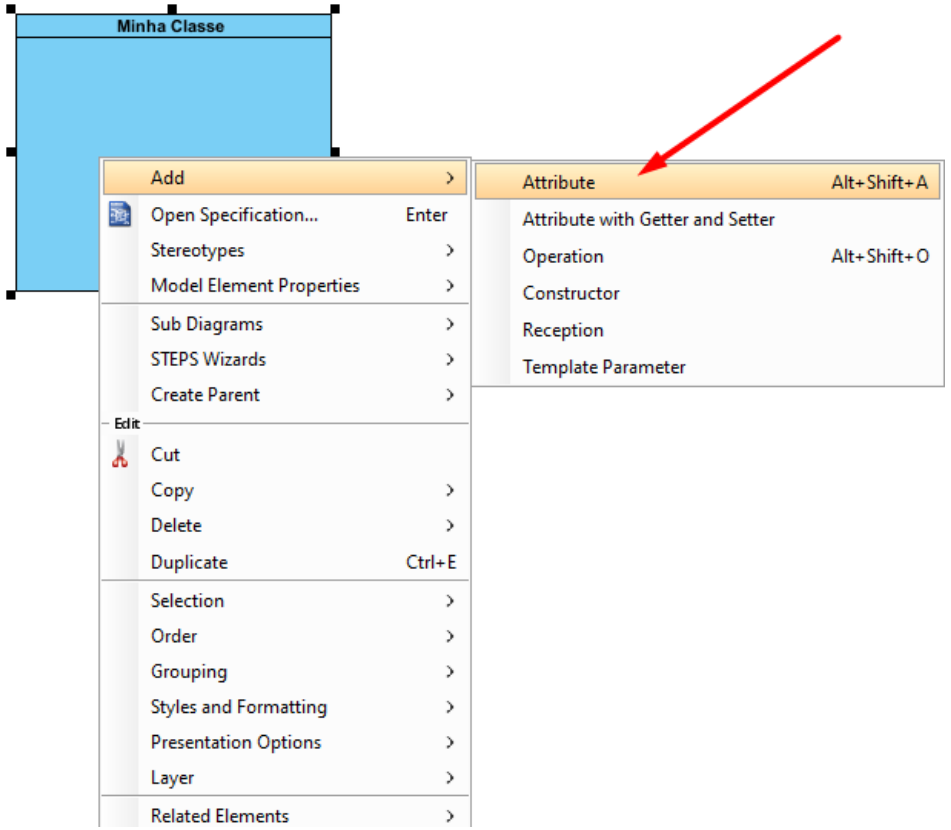
Diagramas de classe no Visual Paradigm



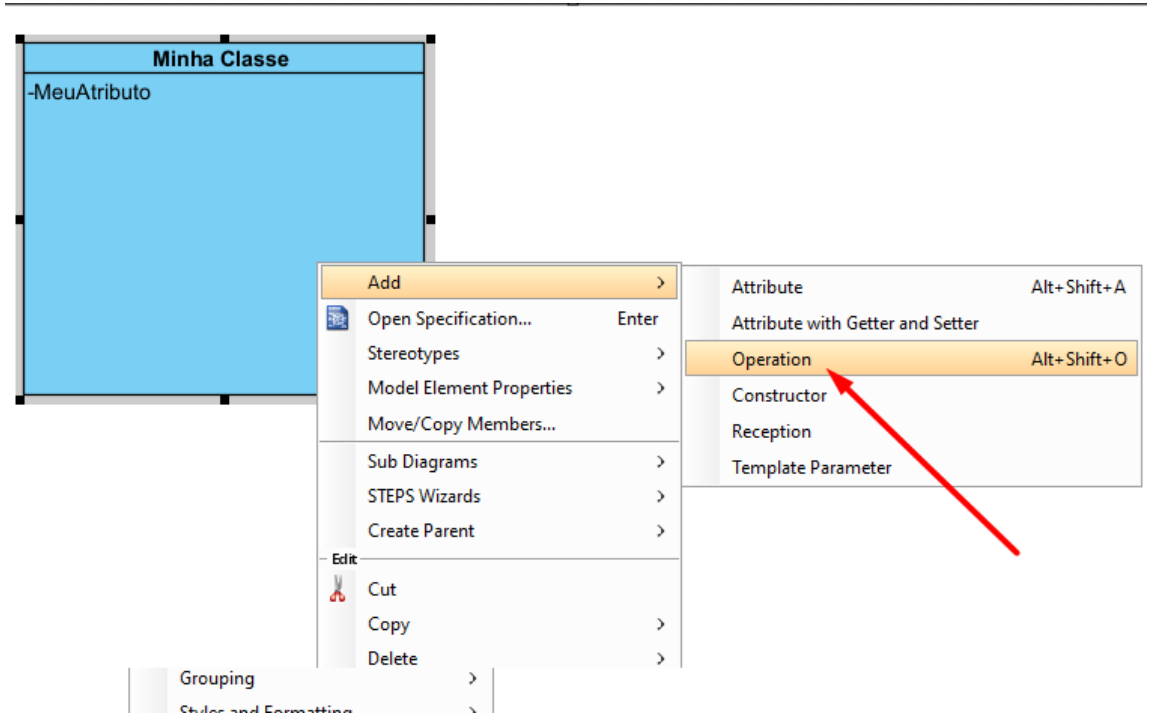
Diagramas de classe no Visual Paradigm



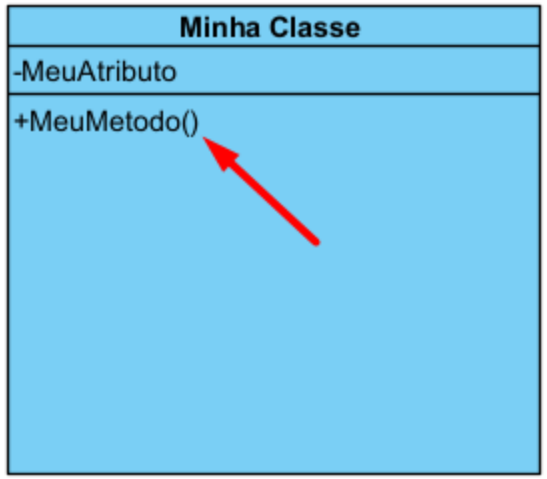
Diagramas de classe no Visual Paradigm



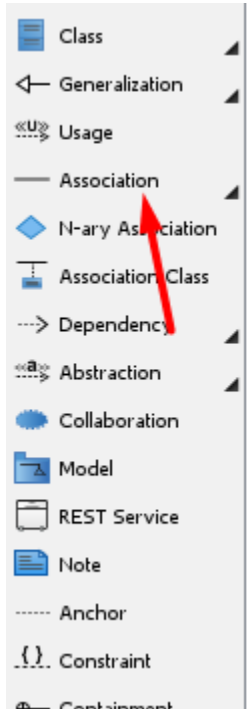
Diagramas de classe no Visual Paradigm

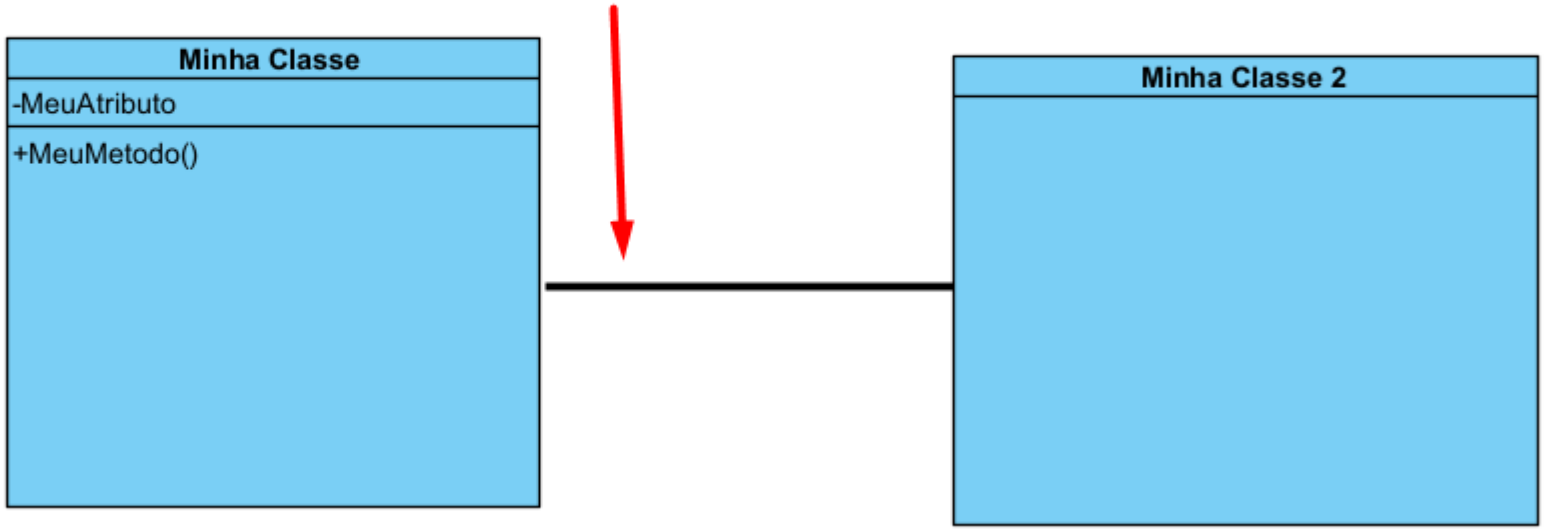


Diagramas de classe no Visual Paradigm



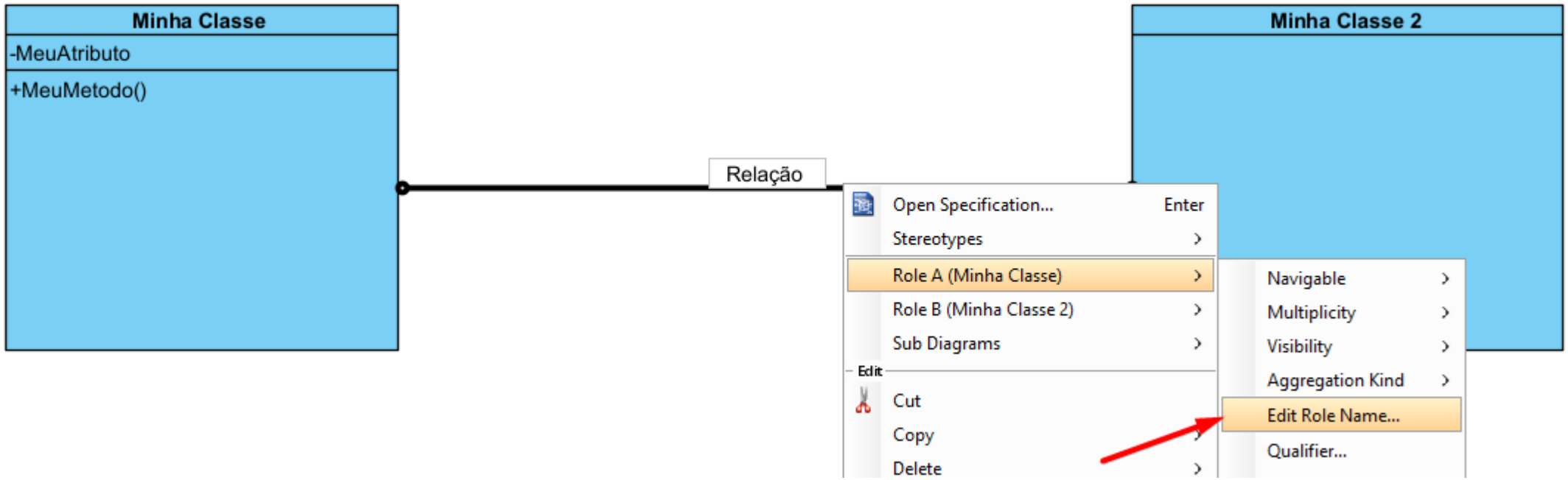
Diagramas de classe no Visual Paradigm



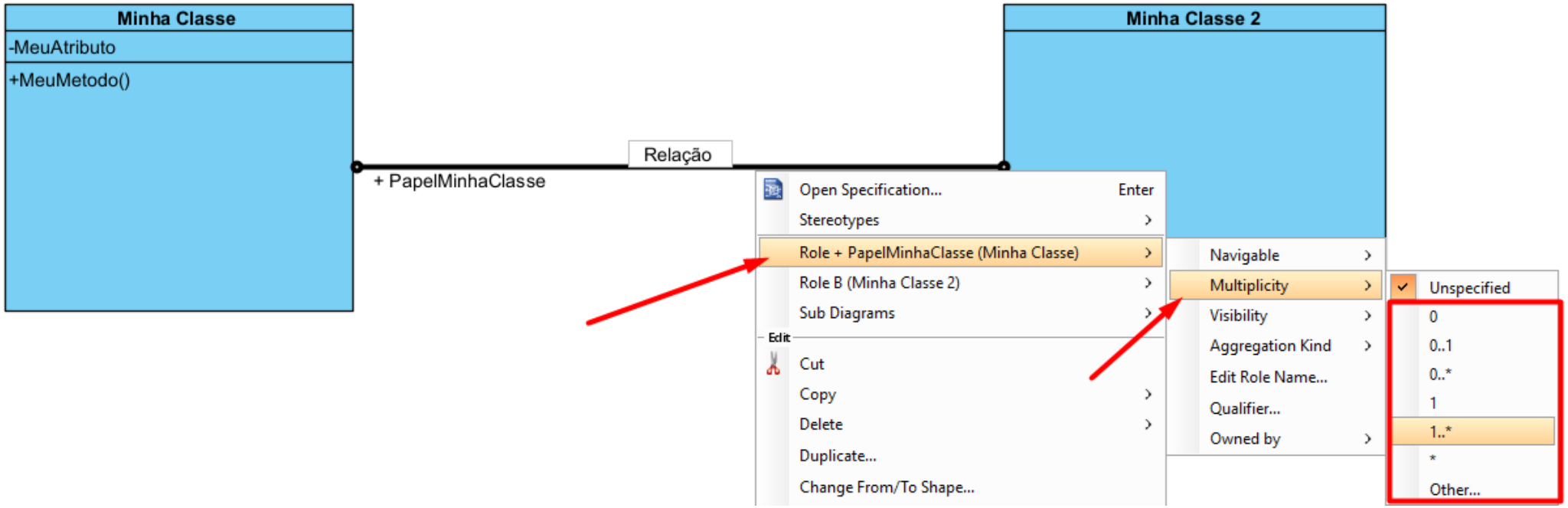


Diagramas de classe no Visual Paradigm

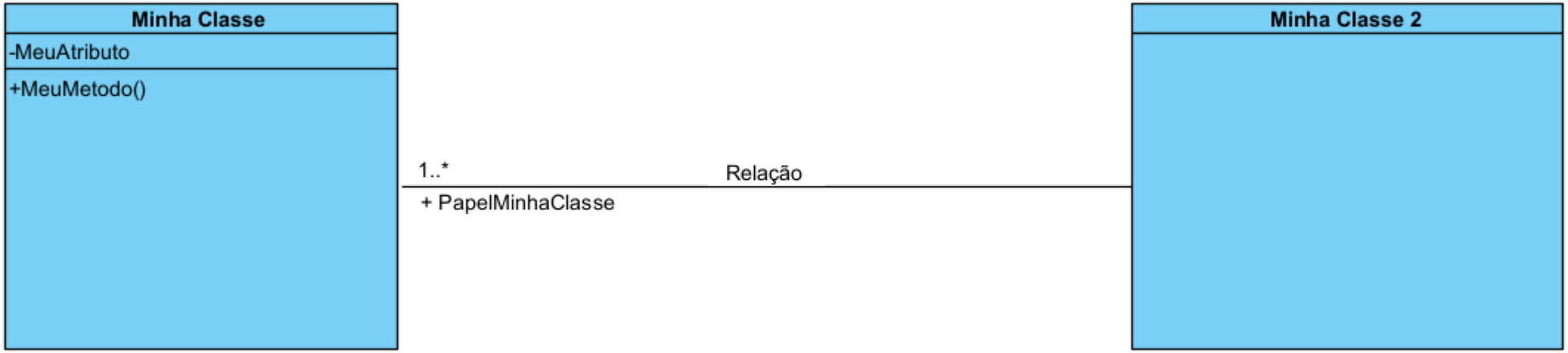
Diagramas de classe no Visual Paradigm



Diagramas de classe no Visual Paradigm



Diagramas de classe no Visual Paradigm



Exercício de fixação

A secretaria de esportes de uma cidade é responsável por realizar competições esportivas em eventos durante o ano.

Considere um sistema responsável por controlar essas competições, armazenando informações de atletas, equipes e das próprias competições. De um atleta, serão consideradas as informações de nome, CPF, peso, altura e data de nascimento. A partir de um atleta deverá ser possível alterar as informações de peso e altura. Nesse contexto, um atleta poderá estar escrito em no máximo 3 equipes.

Das equipes a serem cadastradas, deverão constar informações como: nome, esporte, treinador e data de fundação. A partir de uma equipe deve ser possível alterar seu treinador. Adicionalmente, uma equipe poderá estar inscrita em nenhuma, uma ou várias competições. Além disso deverá ser liderada por exatamente um atleta.

Já para competição, será necessário armazenar seu nome, data de início, data final e local. A partir de uma competição deve ser possível alterar sua data de início, data final e local.

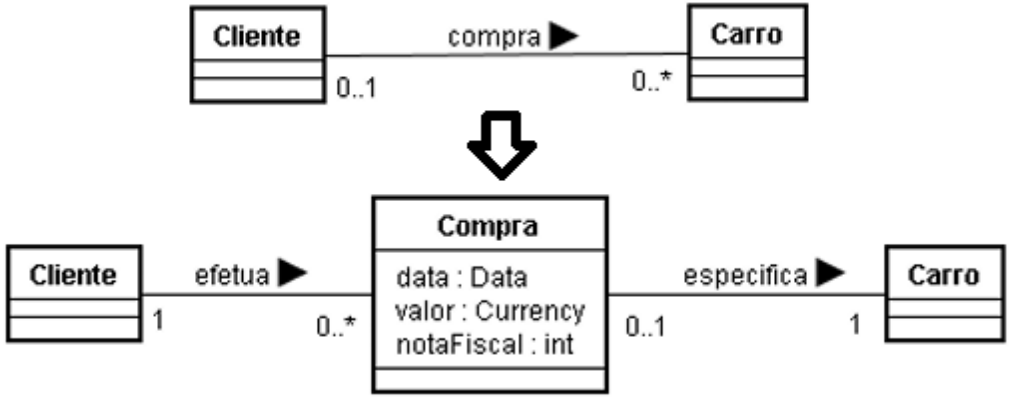
Crie um diagrama de classe para o contexto acima no software Visual Paradigm.

Parte 2

Eventos como classes/associações

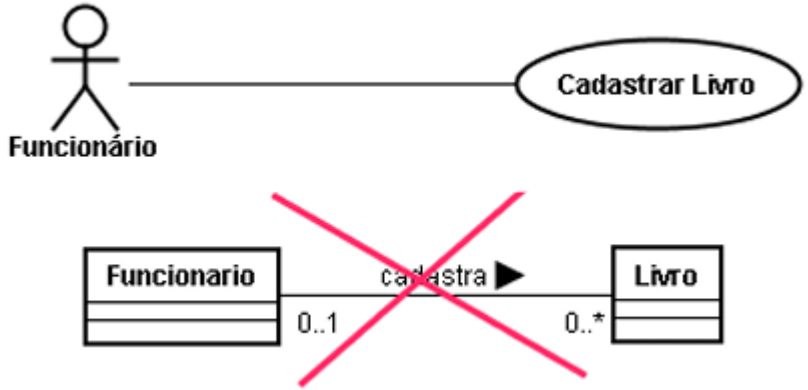
- Classes e atributos geralmente podem ser encontrados de maneira fácil em descrições de minimundo e de casos de uso;

- Contudo, há eventos importantes a nível de negócio que precisam ter sua ocorrência registrada:
  - Capturados inicialmente como associação, mas modelado como classe;



Eventos como classes/associações

- Questionar se aquele evento é relevante para o contexto em questão;
- No exemplo abaixo, será necessário saber qual funcionário cadastrou o livro?
  - Na maioria das vezes, não.



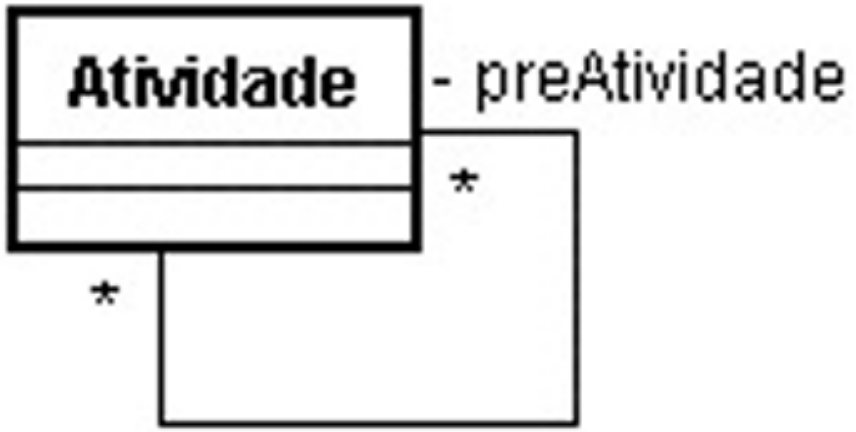
Associações muitos-para-muitos

- Também previstas na modelagem de classes UML;
- Atenção a possíveis atributos do relacionamento (eventos a serem lembrados)

- É requisito do sistema armazenar data de início e fim da alocação?

Relacionamento recursivo

- Ocorre entre elementos de uma mesma classe;
- Maior necessidade de definição de papéis;



Relações todo-parte: composição e agregação

- Associações de forma geral não representam hierarquia entre classes:
  - Mesmo nível, sem importância maior clara entre uma e outra;

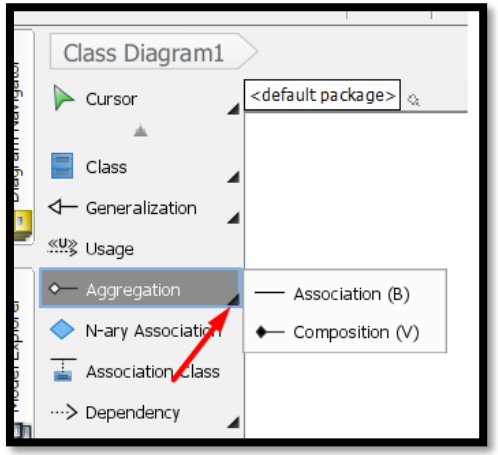
- Porém existem associações consideradas mais fortes, onde um objeto é composto por outro (todo-parte);

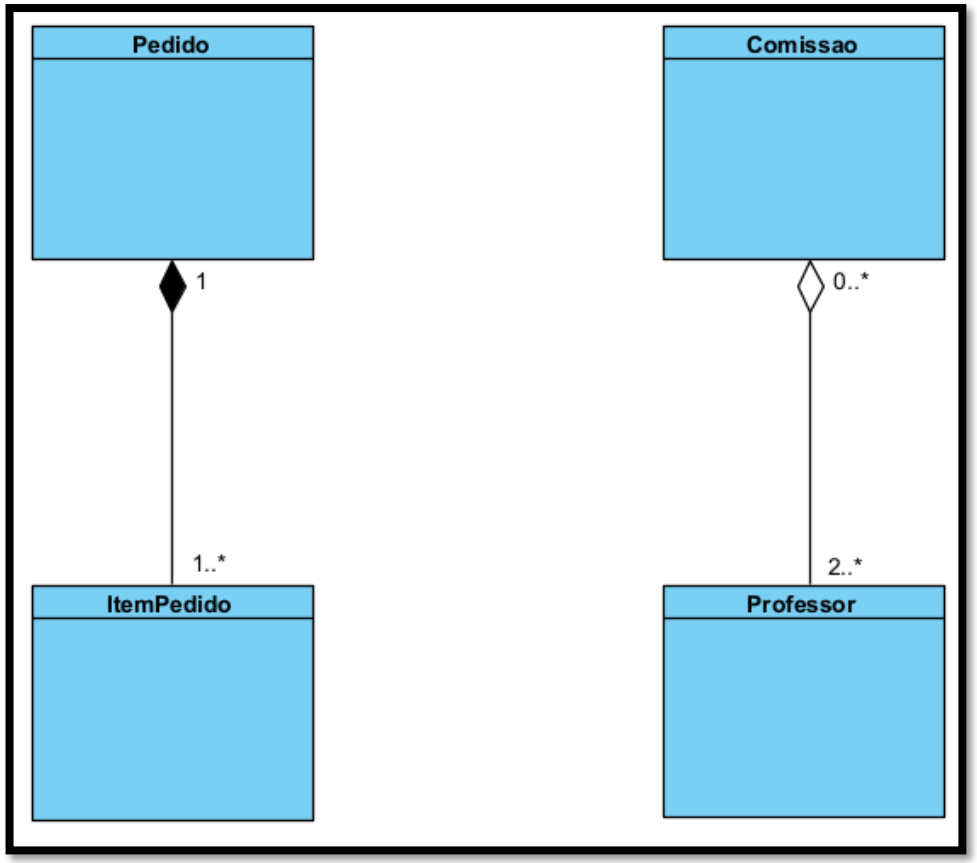
- **Composição**:
  - Componentes (parte) se relaciona a apenas a um único objeto agregado (todo);
  - Não faz sentido a parte existir sem o todo;
  - Exemplo: Pedido, Item de Pedido;

- **Agregação**:
  - Componentes (parte) pode se relacionar com vários objetos agregados (todo);
  - Parte pode continuar existindo se o todo não existir;
  - Exemplo: Professor, Comissão;

Relações todo-parte: composição e agregação

- **Composição**:
  - Losango preenchido na parte do todo;
- **Agregação**:
  - Losango contornado na parte do todo;





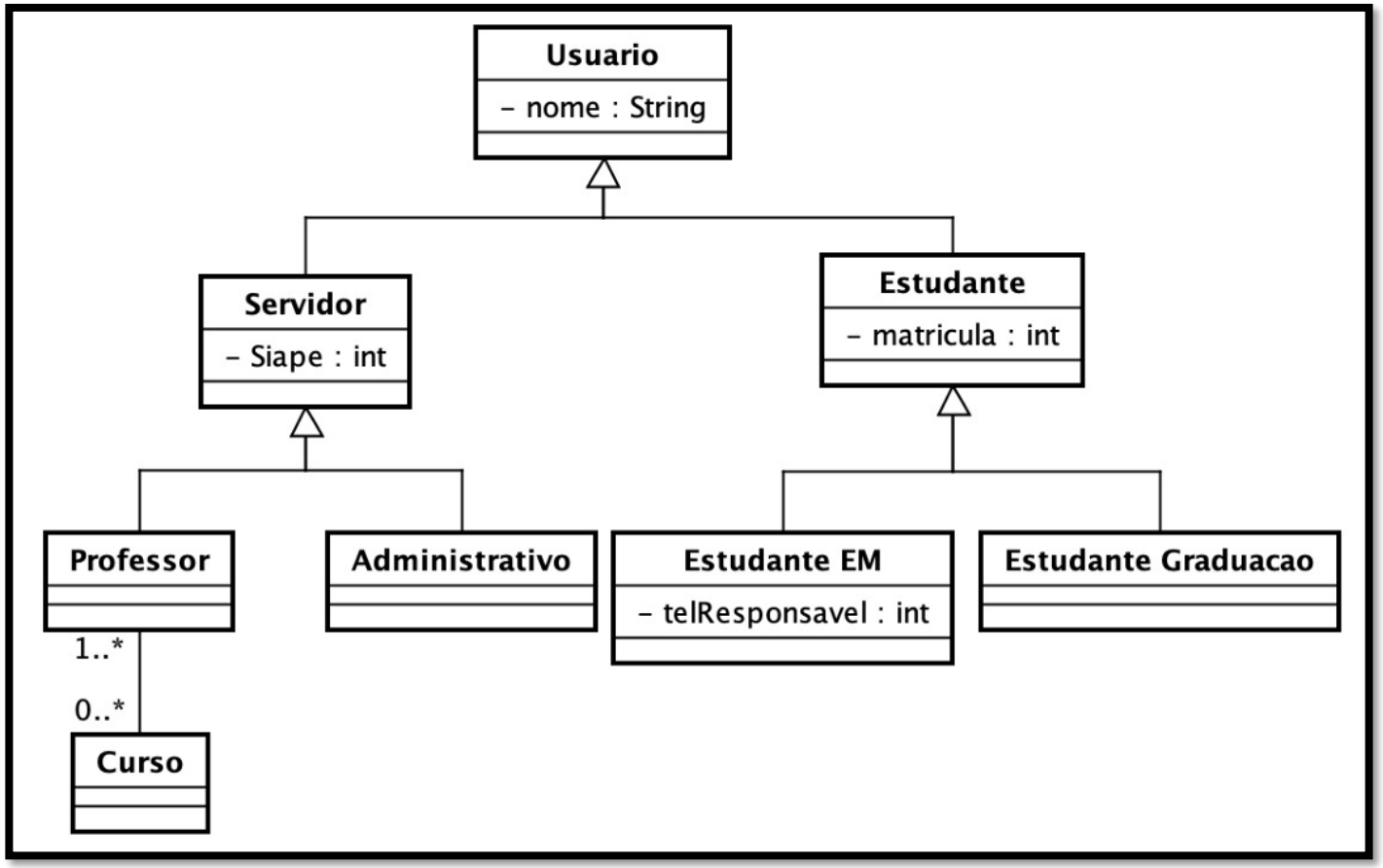
Relações todo-parte: composição e agregação

- Na agregação não há uma dependência existencial entre “todo” e “parte”;
- A composição é mais forte, havendo dependência existencial entre “parte” e “todo”:
  - Como consequência, sempre que o “todo” for excluído, as “partes” também serão.
- Pode ser difícil perceber que uma relação trata-se de um “todo-parte”. Na dúvida, indicado utilizar associação comum para impor menos restrições;

Generalização / Especialização

- Mecanismo também conhecido como herança;
- Generalização/Especialização: depende da direção da leitura;
- São formadas hierarquias de classes:
  - “Filhos” (ou subclasses) herdar estrutura e comportamento dos “pais” (superclasses) e demais “ancestrais”, de forma indireta;
- Possibilita:
  - Reutilização;
  - Captura explícita de características comuns;
  - Definição incremental de classes;
- Representado por um triângulo contornado no lado da superclasse;

Generalização / Especialização



Cuidados com uso de herança

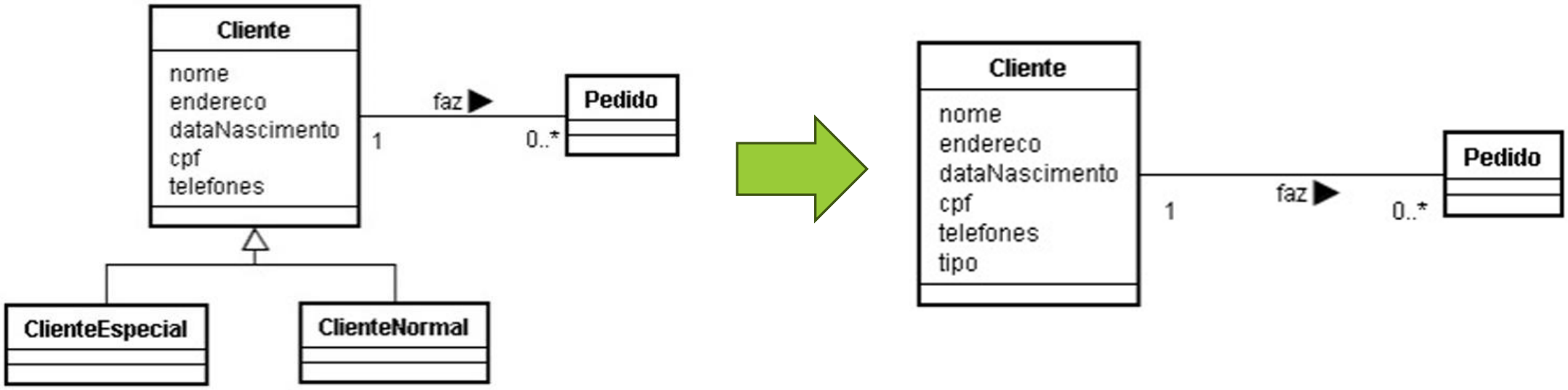
- Semântica da herança:
  - “é um tipo de”;
  - Ex: Universitário é um tipo de Estudante;
- Semântica diferente das associações:
  - Associações mapeiam ligações (relações) entre as classes;
  - Herança indica que instâncias (objetos) da subclasse são também objetos da superclasse;

Cuidados com uso de herança

- Subclasses devem suportar todas as funcionalidades da superclasse;
- Além disso, também as suas particularidades;
- Se a subclasse precisa cancelar características da sua superclasse, atenção pois existe algo errado:
  - Necessariamente todas as funcionalidades da superclasse devem ser úteis a subclasse;
- Funcionalidades comuns a diversas classes devem estar no mais alto nível possível da hierarquia;

Diretrizes para herança

- Classes que não adicionam funcionalidade nem redefinem funcionalidades existentes, podem ser eliminadas (substituídas por atributos de tipos enumerados):



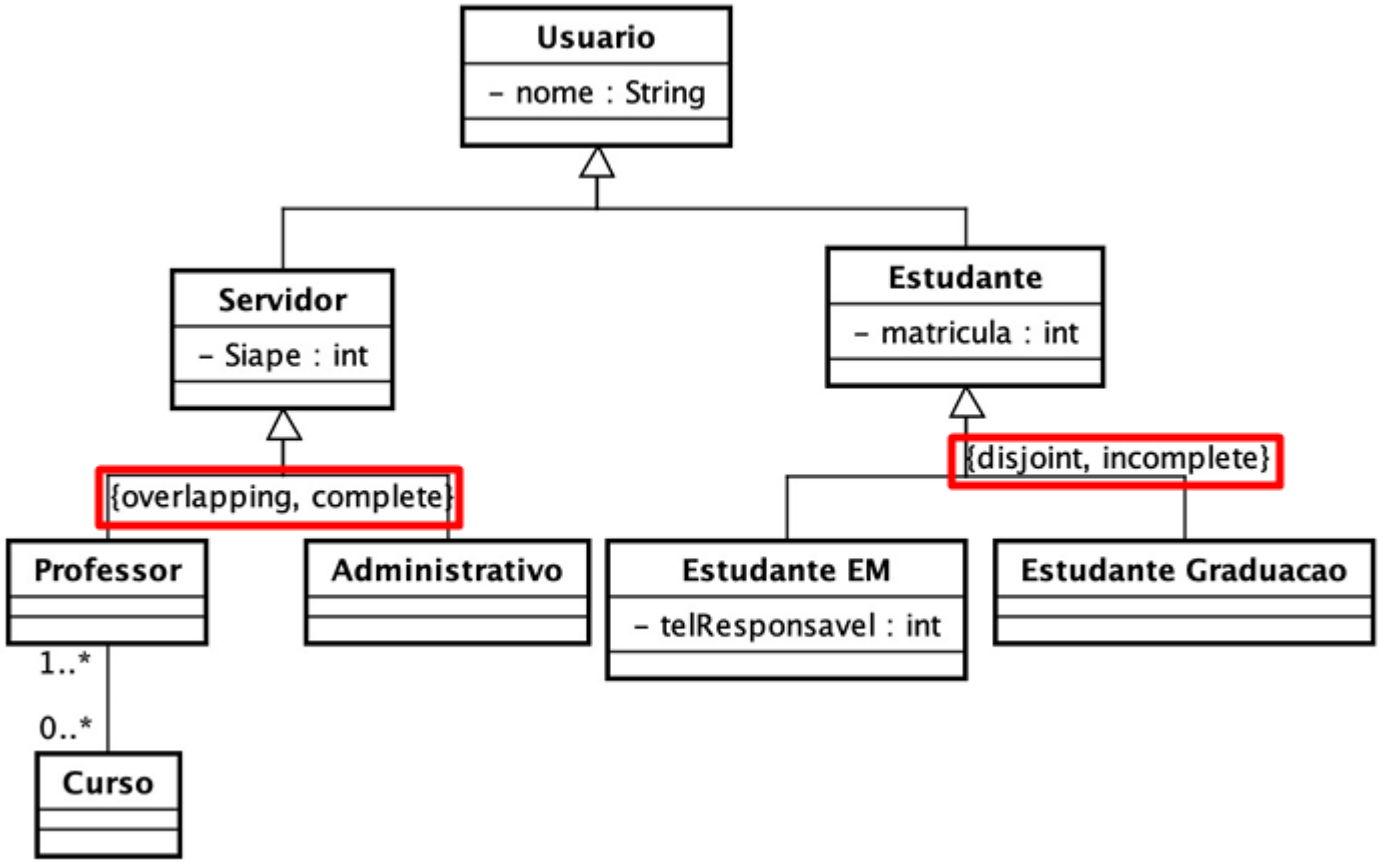
Conceitos avançados em herança

- ***Generalization set***: conjunto que indicará particularidades da generalização, a respeito de:

- **Exclusividade**: indica se a instância de uma superclasse pode ser instância de múltiplas subclasses ou não. Ou seja, se as subclasses do conjunto são mutuamente exclusivas ou não;
  - **Disjoint**: não pode haver sobreposição;
  - **Overlapping**: é permitida a sobreposição;

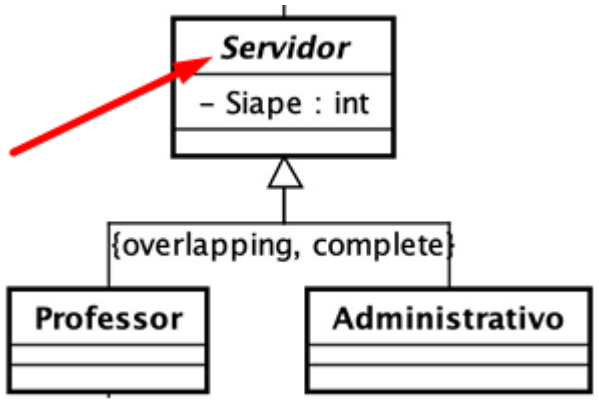
- **Completude**: indica que todas as instâncias da superclasse devem ser instâncias de pelo menos uma subclasse do conjunto;
  - **Complete**: Todas as instâncias da superclasse pertence, necessariamente, a uma de suas subclasses;
  - **Incomplete**: Pode haver instâncias da superclasse que não pertencem a nenhuma de suas subclasses;

Conceitos avançados em herança



Classes abstratas

- Utilizadas para organizar características comuns a diversas subclasses;
- Desenvolvida para ser herdada;
- Não possui instâncias diretas (somente indiretas):
  - Não podem ser instanciadas;
- Nome da classe em itálico no diagrama;



Exercício de fixação

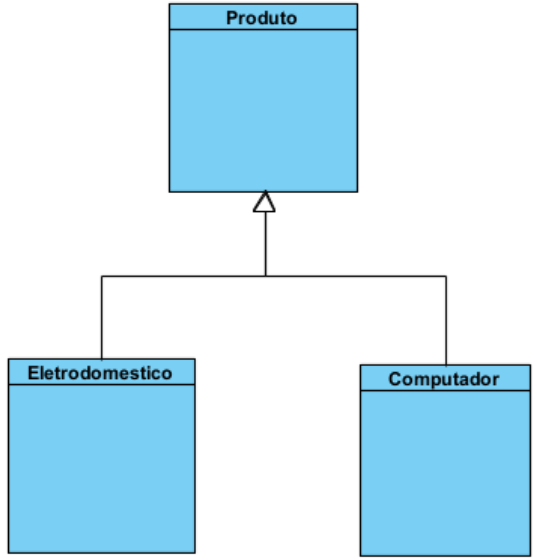
Uma empresa com foco na venda de eletrodomésticos e artigos de informática deseja implementar um sistema para gerenciar as vendas de seus produtos. As entidades levantadas para este sistema são as seguintes:

- **Produto**: A loja vende Eletrodomésticos (como geladeiras, micro-ondas e fogões) e Computadores (desktops e notebooks). Todo produto possui nome, código único, preço e quantidade em estoque. Neste contexto será permitido um produto não ser nem eletrodoméstico, nem computador, ou seja, de um tipo geral. Além disso, não será permitido um mesmo produto pertencer a mais de um tipo.
  - **Eletrodomésticos:** deverão armazenar informações de: capacidade, tipo e eficiência energética;
  - **Computadores**: deverão armazenar informações de: Processador, Memória RAM, Sistema Operacional;
- **Venda**: Toda venda possuirá as informações de: código único, data da venda e valor total. Cada venda pode incluir no mínimo um até vários produtos. Um produto pode estar associado de nenhuma até diversas vendas.
- **Cliente**: Sobre os clientes da loja, deverão ser armazenadas informações de: código único, nome e e-mail. Clientes podem ser classificados como:
  - **Pessoa Física**: deverão armazenar informações de: CPF e data de nascimento;
  - **Pessoa Jurídica**: deverão armazenar informações de: CNPJ e responsável legal;

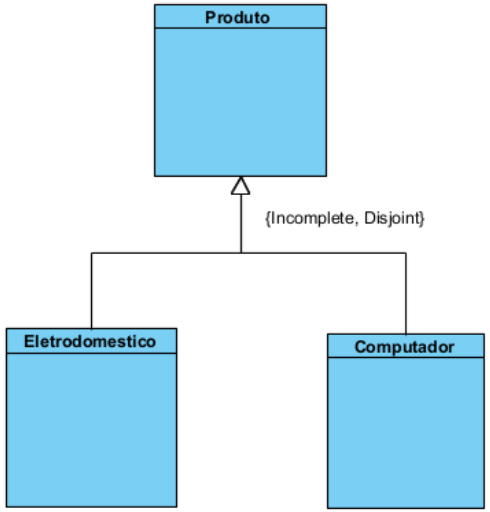
  Todo cliente pertence, necessariamente, a um desses dois tipos. Um cliente pode ter de nenhuma até várias vendas associadas a ele, mas uma venda está associada a somente um cliente.
- **Nota Fiscal**: O sistema também deverá armazenar informações das notas fiscais emitidas. Sobre uma nota, deverá ser armazenado: código único, número da nota, data de emissão e valor total. Uma nota fiscal está associada a somente uma Venda e uma Venda está associada a somente uma nota fiscal;
- **Item de nota**: Elementos que serão associados a uma nota fiscal, descrevendo quais produtos associados. Deverá possuir informações de: código único, descrição do item e quantidade. Todo item de nota pertence, necessariamente, a uma nota fiscal e uma nota fiscal terá no mínimo um e no máximo diversos itens.

- Crie um diagrama de classe que atenda o contexto acima, deixando claro quais os aspectos de herança e hierarquia em relações todo-parte, quando houver.

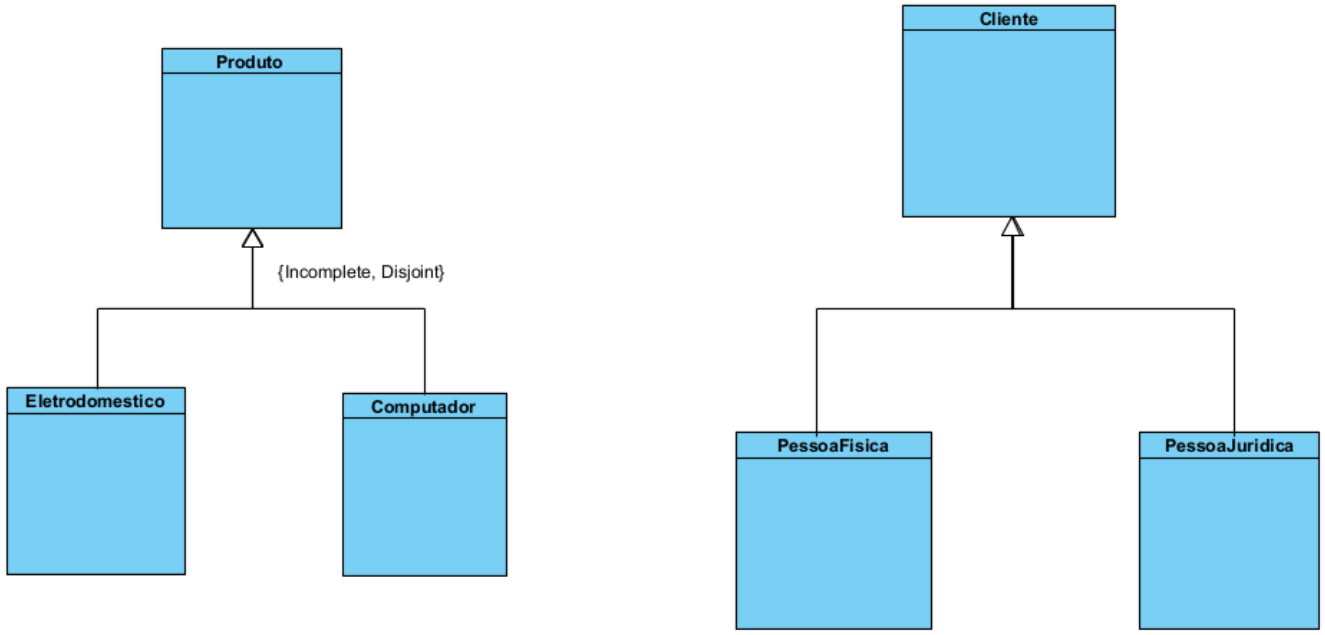
Resolução exercício



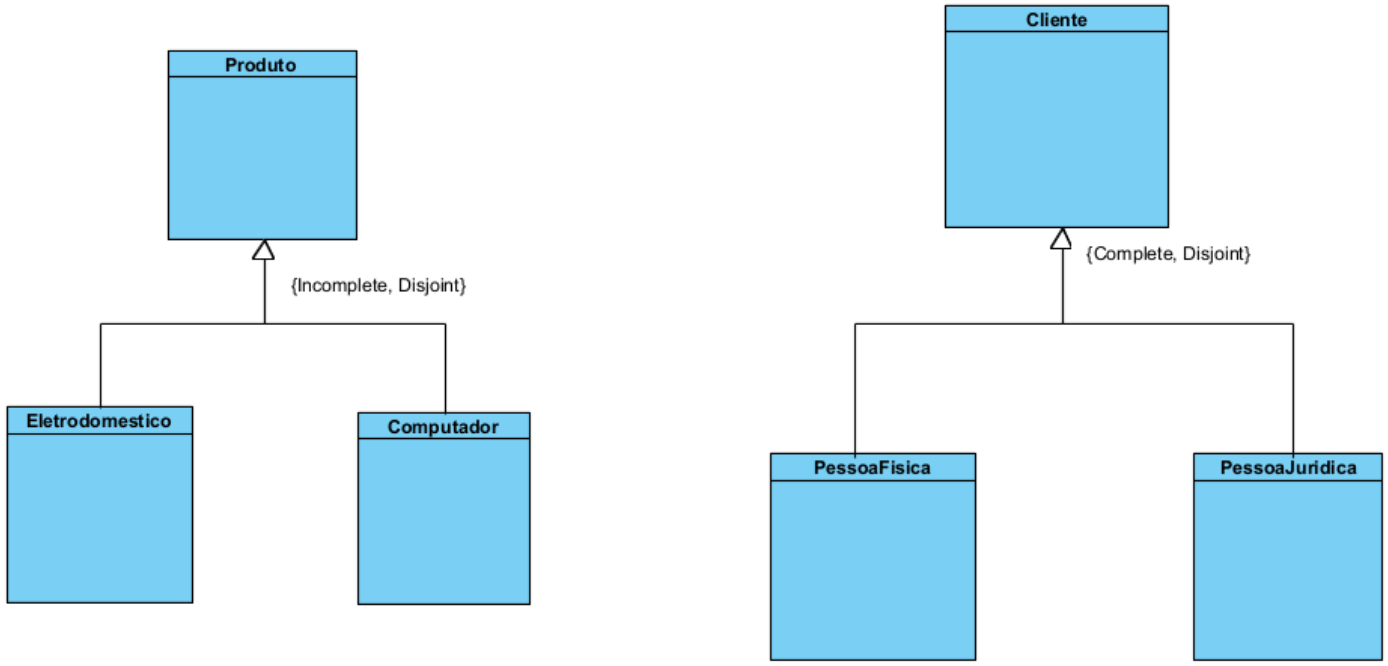
Resolução exercício



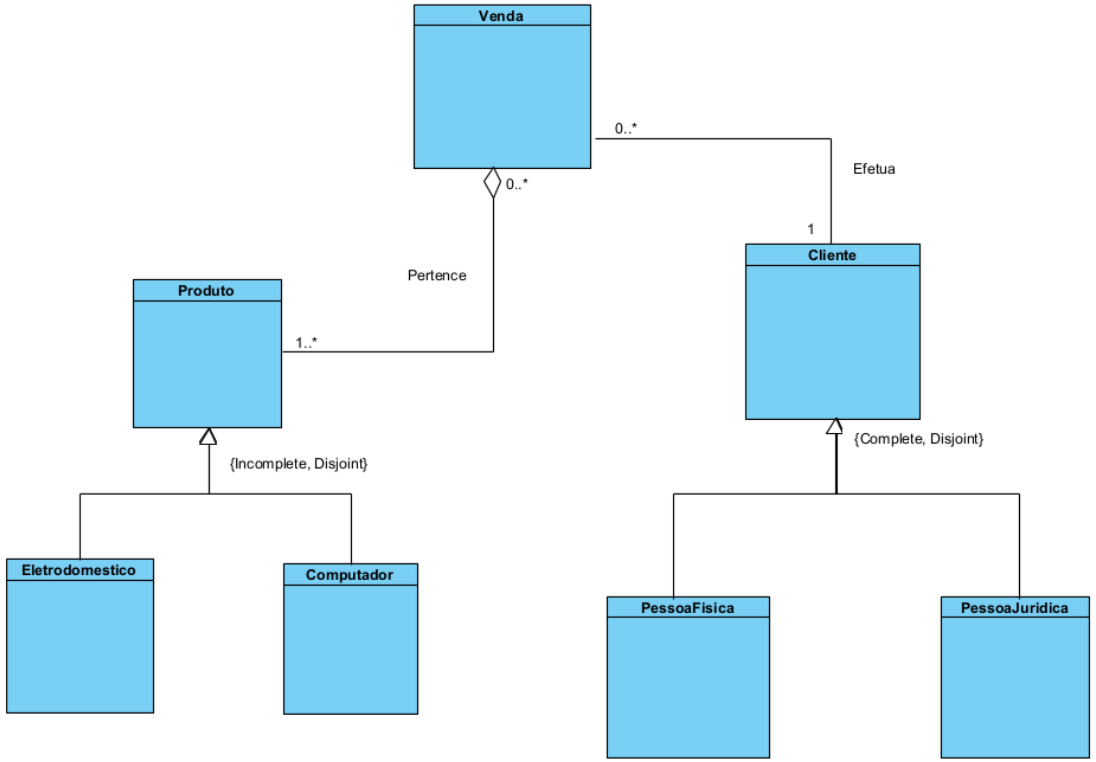
Resolução exercício



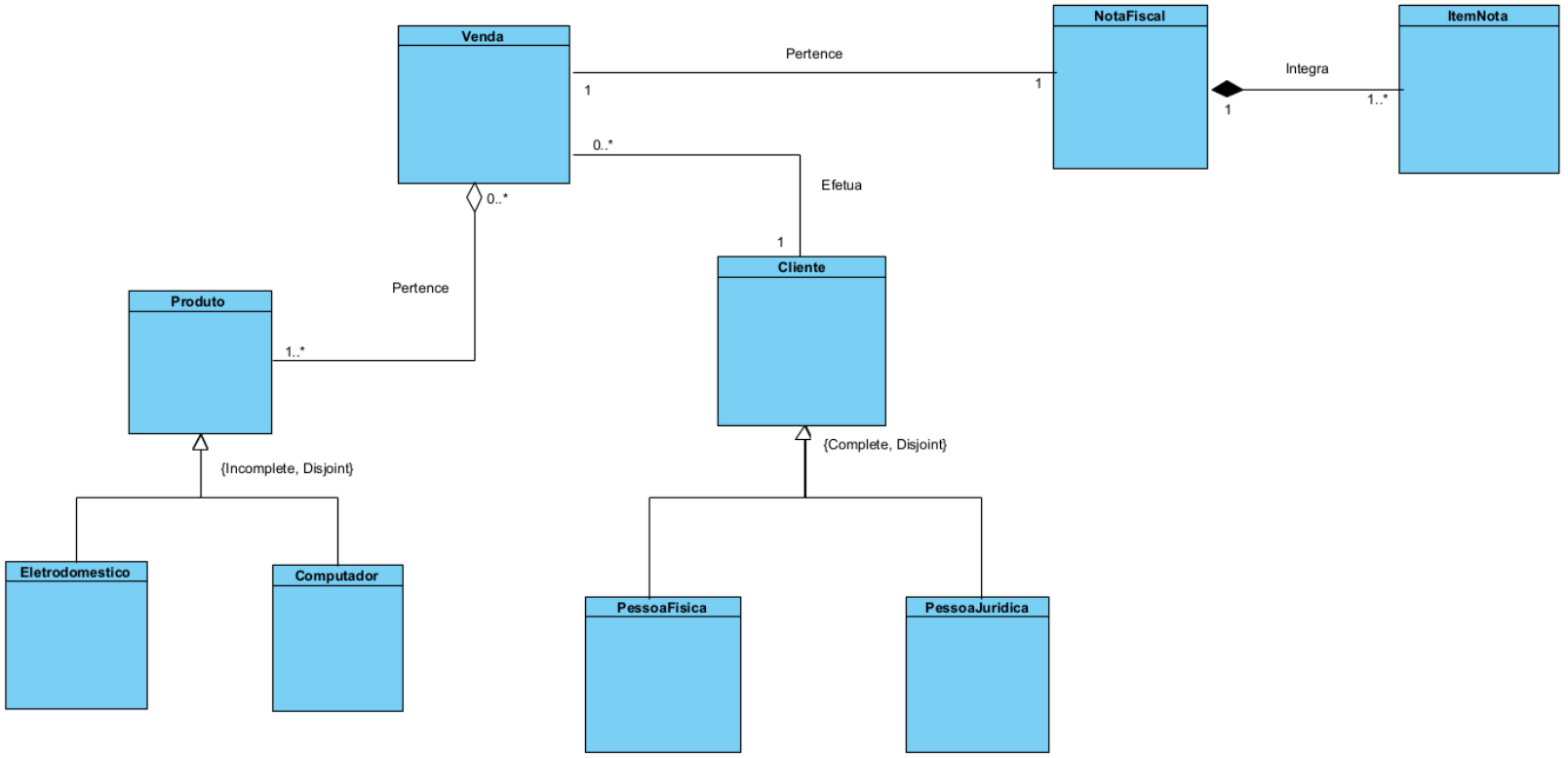
Resolução exercício



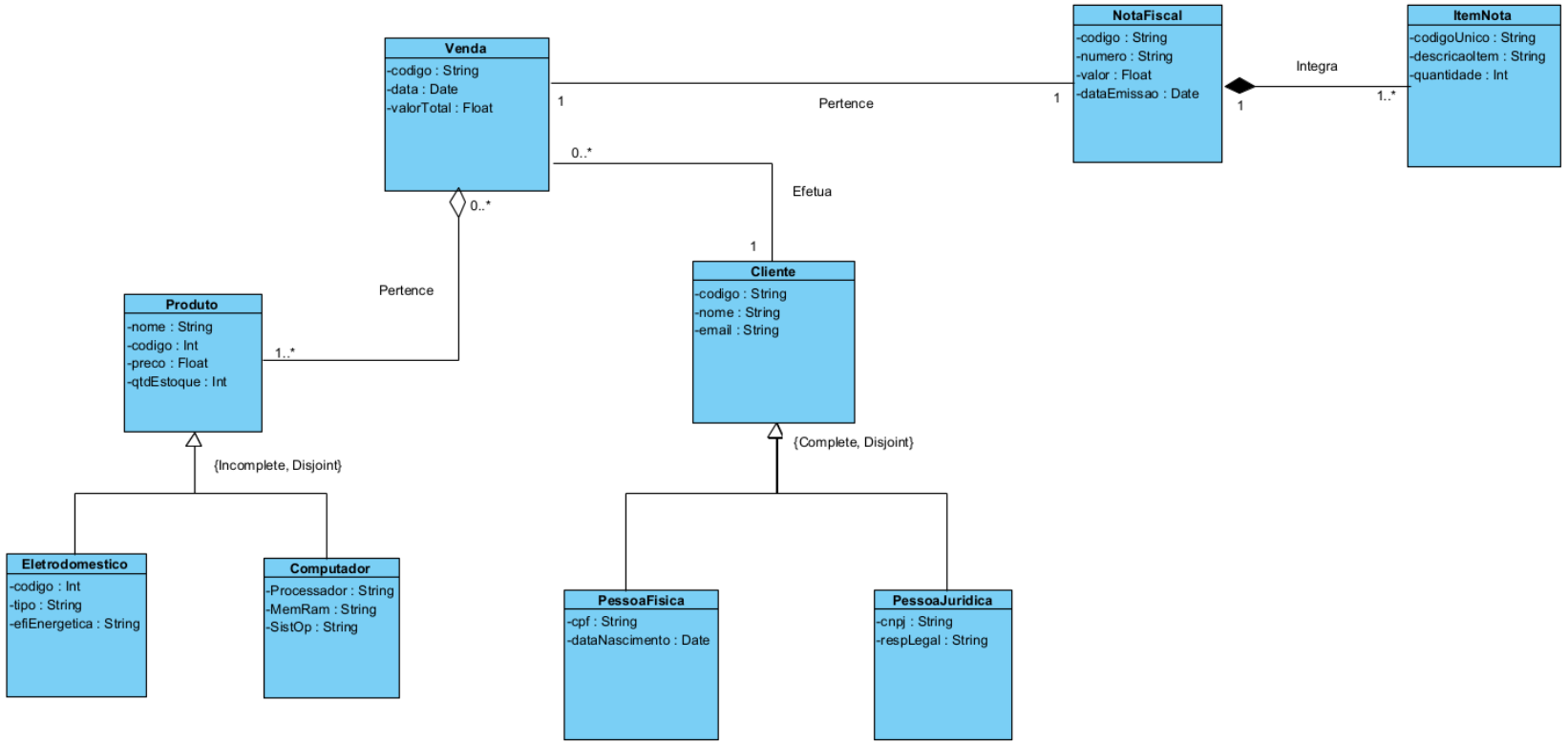
Resolução exercício



Resolução exercício



Resolução exercício



Referências

Este material foi baseado no produzido pelo professor Victorio Albani Carvalho, Tendo como base as notas de aula do professor Ricardo Falbo.