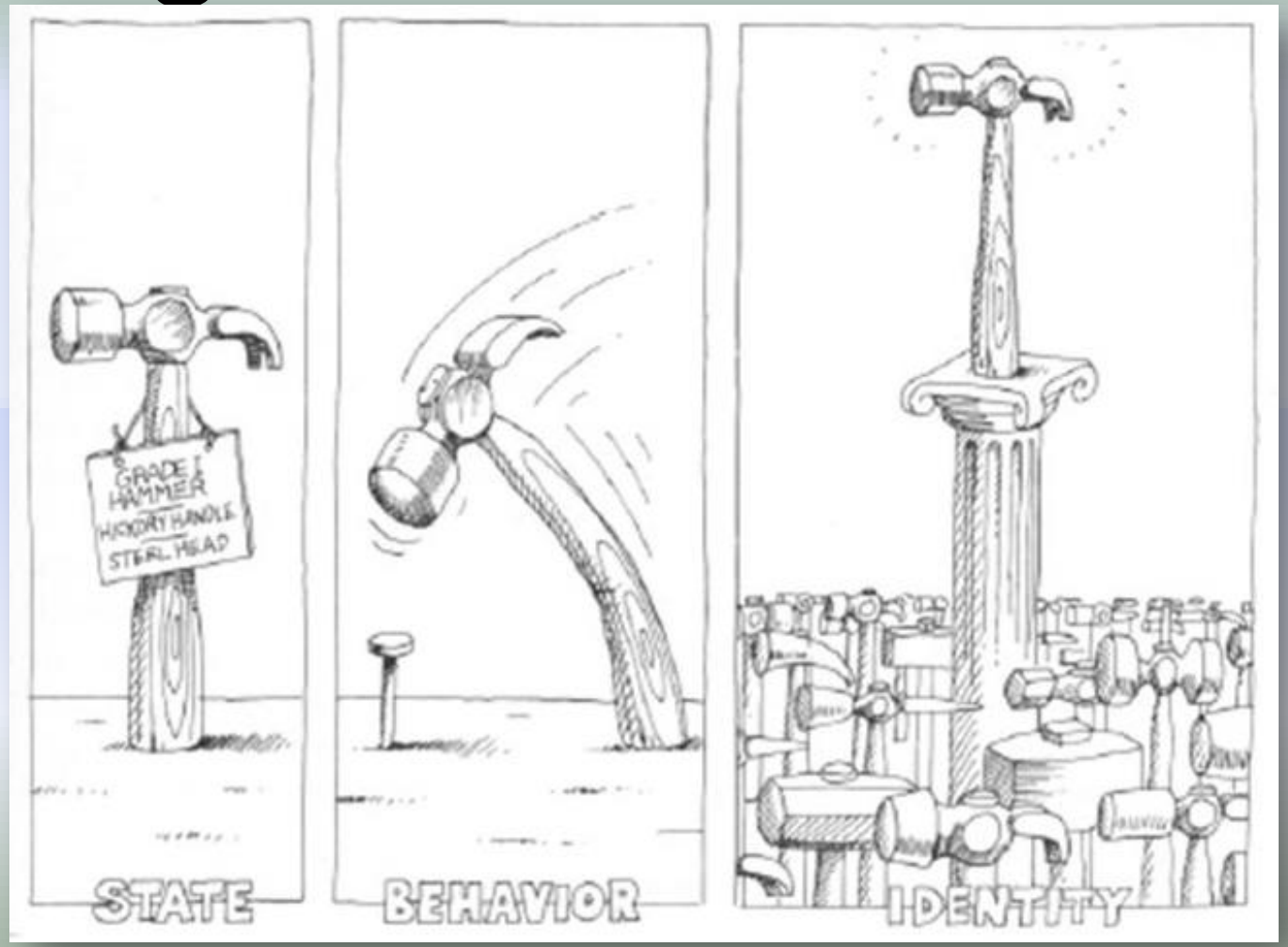
Diagramas de Classes



Conceitos Básicos

- O caso de uso fornece uma perspectiva do sistema de um ponto de vista externo (do ator)
- Internamente os objetos colaboram para atender às funcionalidades do sistema
- Demonstra a estrutura estática dessa colaboração, mostra as classes de um sistema, seus atributos e operações, e como as classes se relacionam.
- O diagrama de objetos (que pode ser visto como uma instanciação do diagrama de classes) também representa a estrutura estática

Perspectivas de um Diagrama de Classes

- O ***modelo conceitual*** (análise) 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*** (projeto) 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.

Definição de Classes

Uma classe de objetos, ou simplesmente classe, descreve :

- Um conjunto de objetos com propriedades semelhantes (atributos);
- O mesmo comportamento (operações);
- Os mesmos relacionamentos; e
- A mesma semântica.

São os blocos de construção mais importantes de qualquer sistema orientado a objetos;

Conceitos Básicos

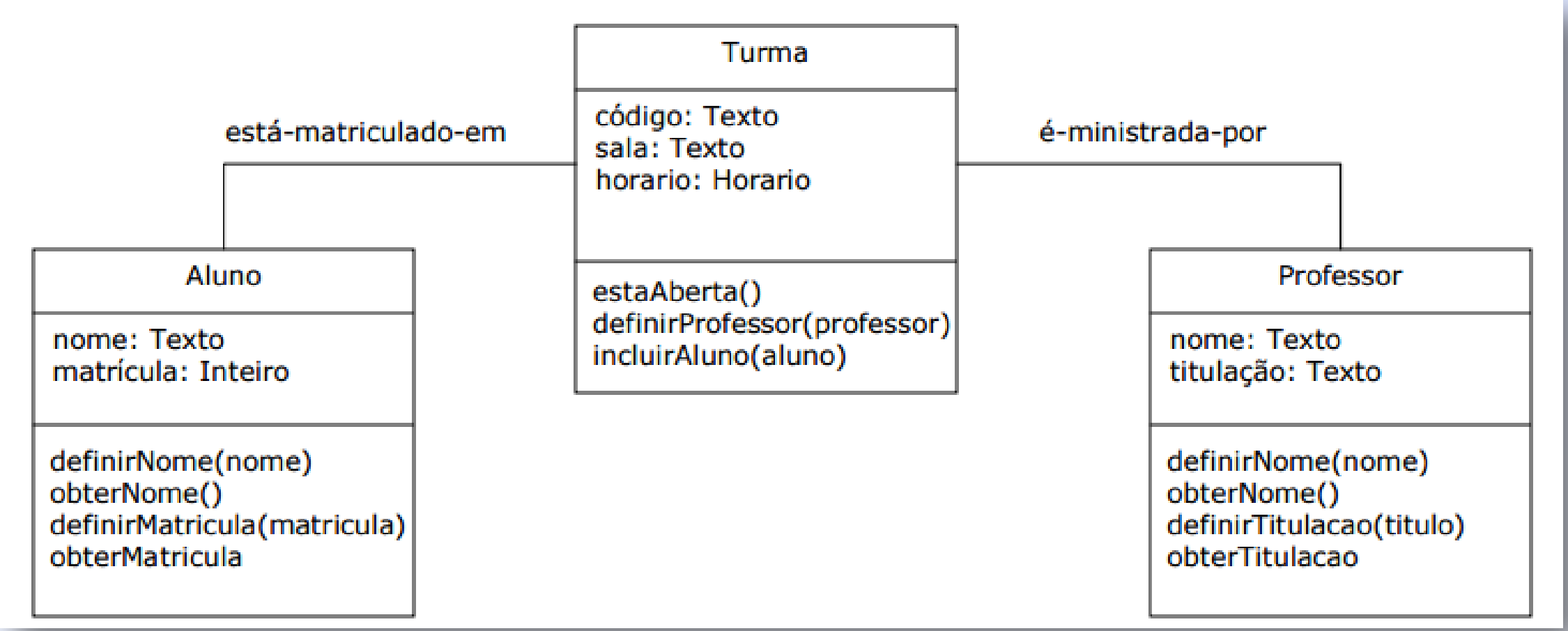
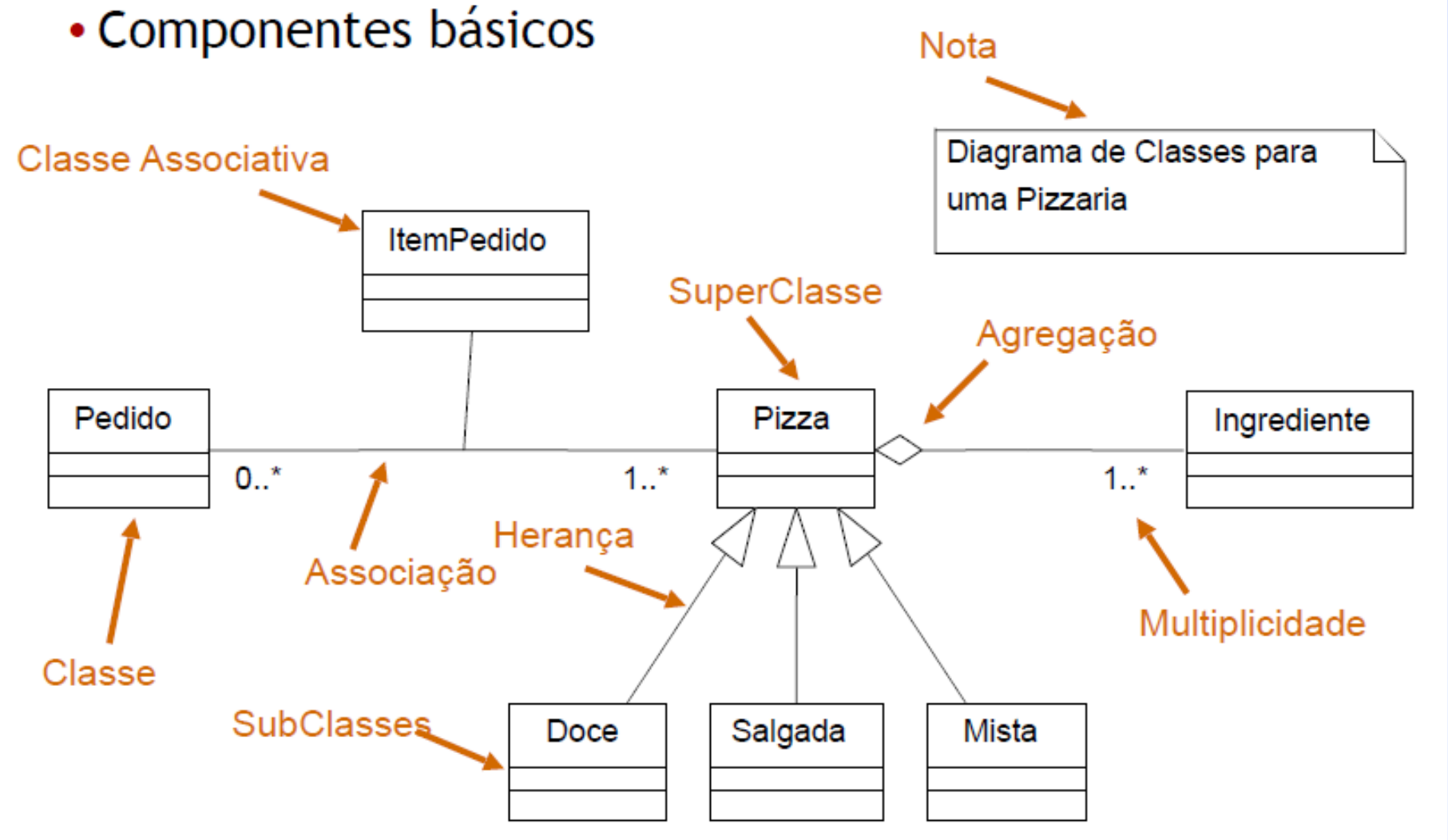


Diagrama de Classes



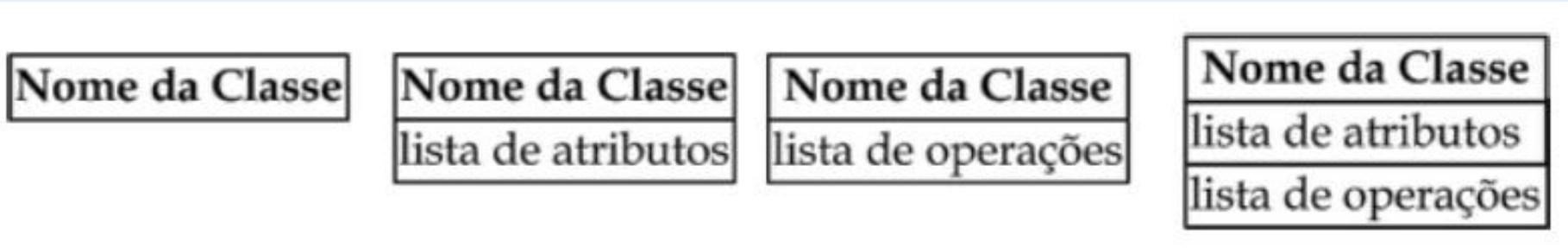
Conceitos Básicos

- Elementos de um diagrama de classes
  - Classes
  - Relacionamentos
    - Associação
    - Agregação
    - Composição
  - Generalização
  - Dependência

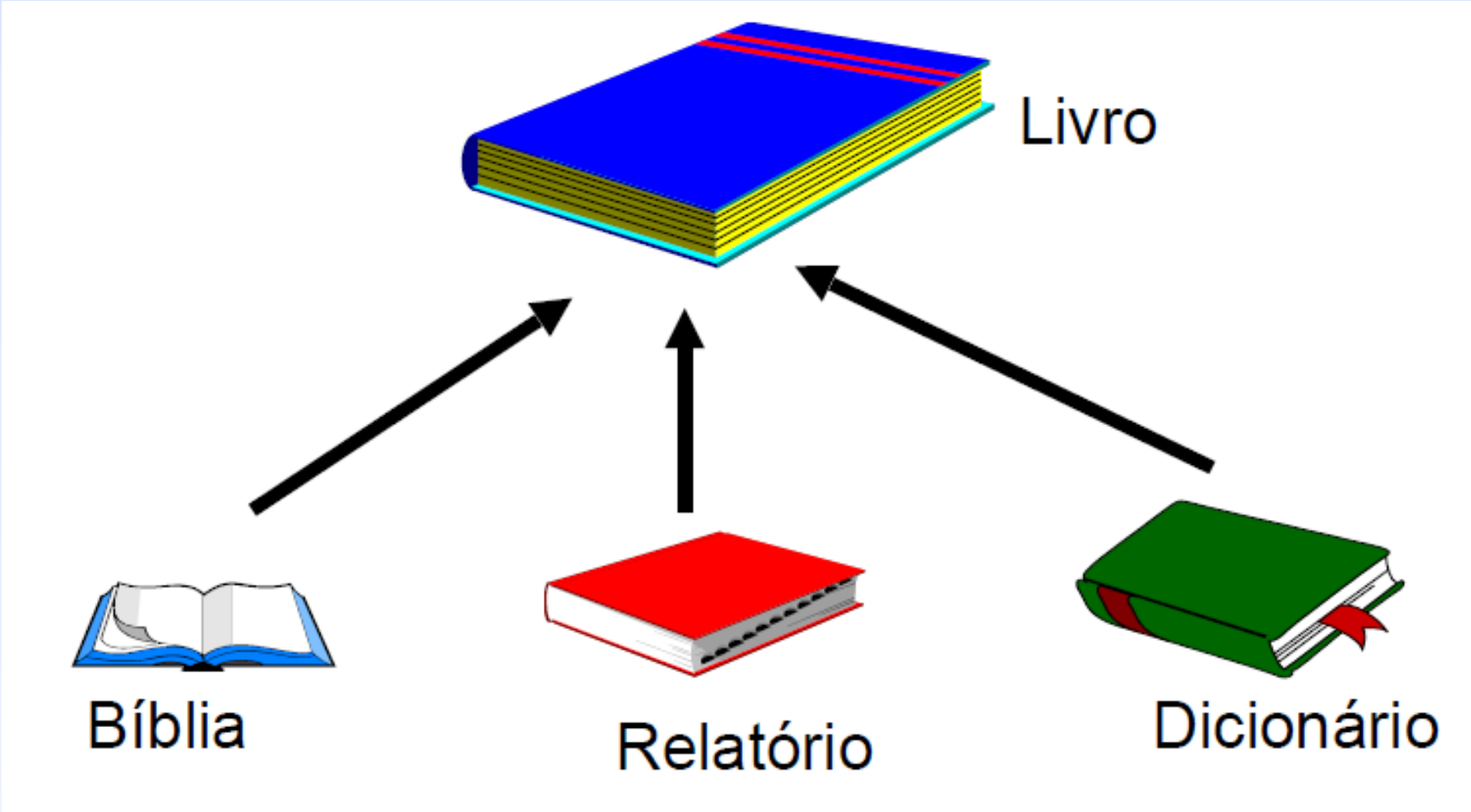
Conceitos Básicos: Classes

- Representada através de uma “caixa” com no máximo três compartimentos exibidos (nome, atributos e métodos).
- Devem receber nomes de acordo com o vocabulário do domínio do problema.
- É comum adotar um padrão para nomeá-las

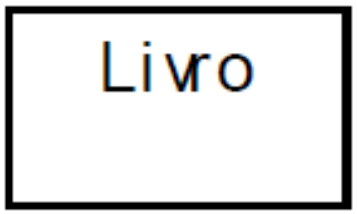
Ex: todos os nomes de classes serão substantivos singulares com a primeira letra maiúscula.



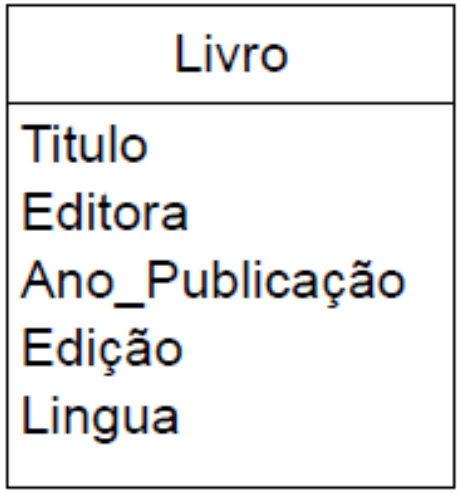
Conceitos Básicos: Classes



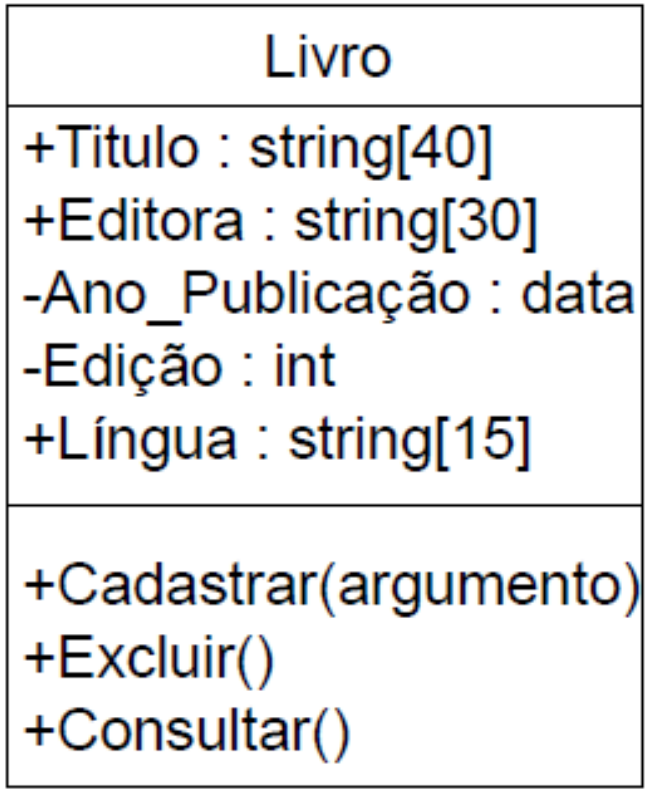
Conceitos Básicos: Classes



Classe com detalhes suprimidos



Classe com detalhes a nível de análise



Classe com detalhes a nível de implementação

Conceitos Básicos: Atributos

- Representam uma propriedade do objeto que está sendo modelado;
- Cada atributo possui um valor para cada instância de um objeto;
- Cada nome de atributo deve ser único dentro de uma classe;
- Podem ser representados apenas por seu nome, ou especificando seu tipo e valor inicial;

Conceitos Básicos: Atributos

– Representam o conjunto de características (estado) dos objetos daquela classe

– Visibilidade:

+ público: visível para qualquer classe

protegido: visível somente para classes derivadas

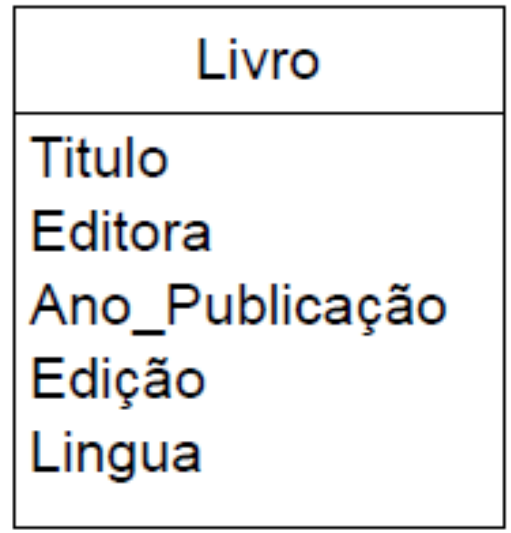
- privado: visível somente para classe

Exemplo:

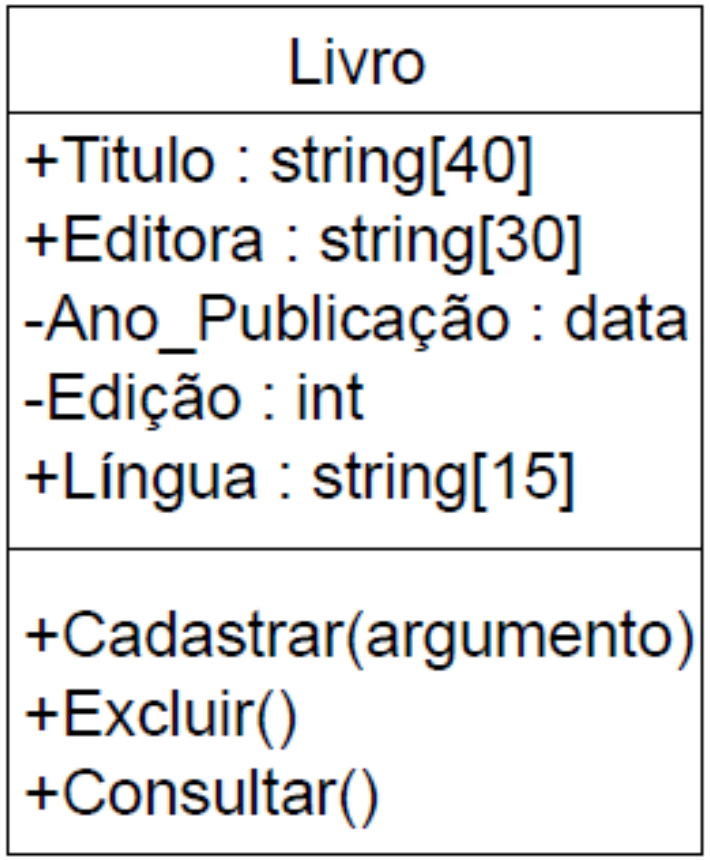
+ nome : String

Conceitos Básicos: Atributos

Conceitos Básicos: Atributos



Atributos Simples



Atributos Completos

Conceitos Básicos: Métodos

– Representam o conjunto de operações (comportamento) que a classe fornece

– Visibilidade:

+ público: visível para qualquer classe

protegido: visível somente para classes derivadas

- privado: visível somente para classe

Exemplo:

- getNome() : String

Classes: Relacionamentos

- Os relacionamentos possuem:

– Nome: descrição dada ao relacionamento (faz, tem, possui,...)

– Sentido de leitura

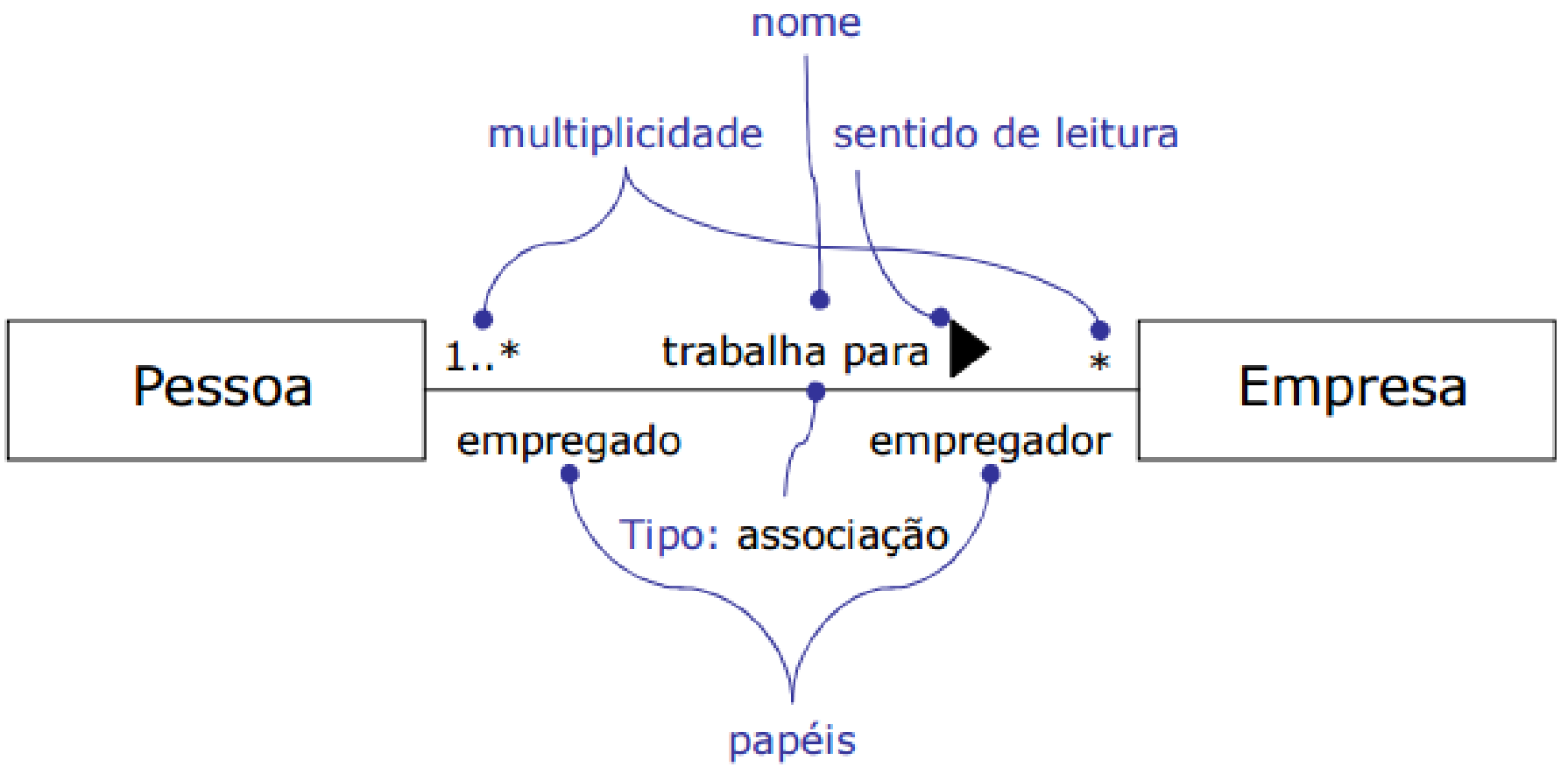
– Navegabilidade: indicada por uma seta no fim do relacionamento

– Multiplicidade: 0..1, 0..*, 1, 1..*, 2, 3..7

– Tipo: associação (agregação, composição), generalização e dependência

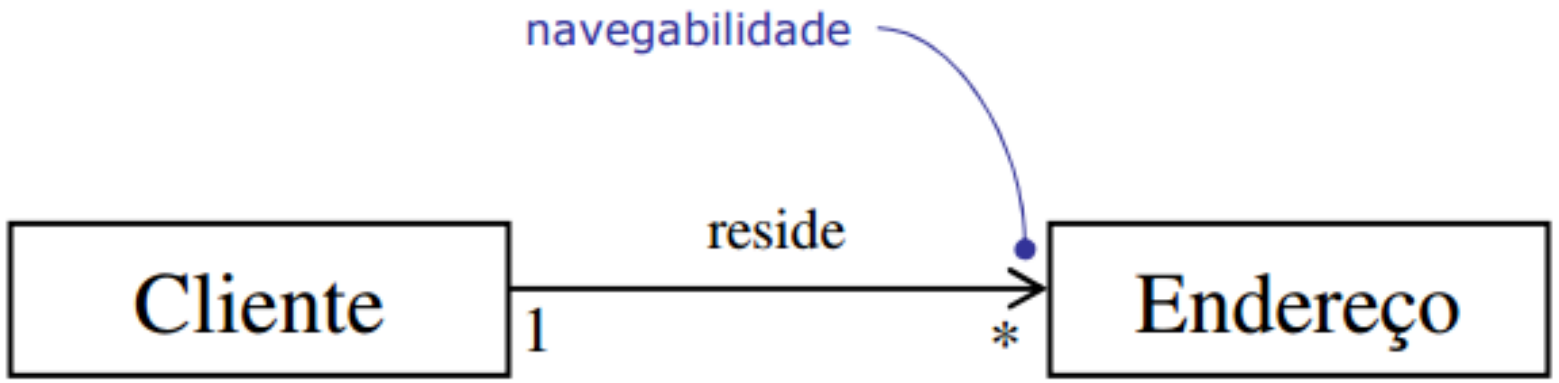
– Papéis: desempenhados por classes em um relacionamento

Classes: Relacionamentos



E a navegabilidade?

Classes: Relacionamentos



- O cliente sabe quais são seus endereços, mas o endereço não sabe a quais clientes pertence

Classes: Relacionamentos

Relacionamentos: **Associação**

- Uma associação é um relacionamento estrutural que indica que os objetos de uma classe estão vinculados a objetos de outra classe.
- Uma associação é representada por uma linha sólida conectando duas classes.

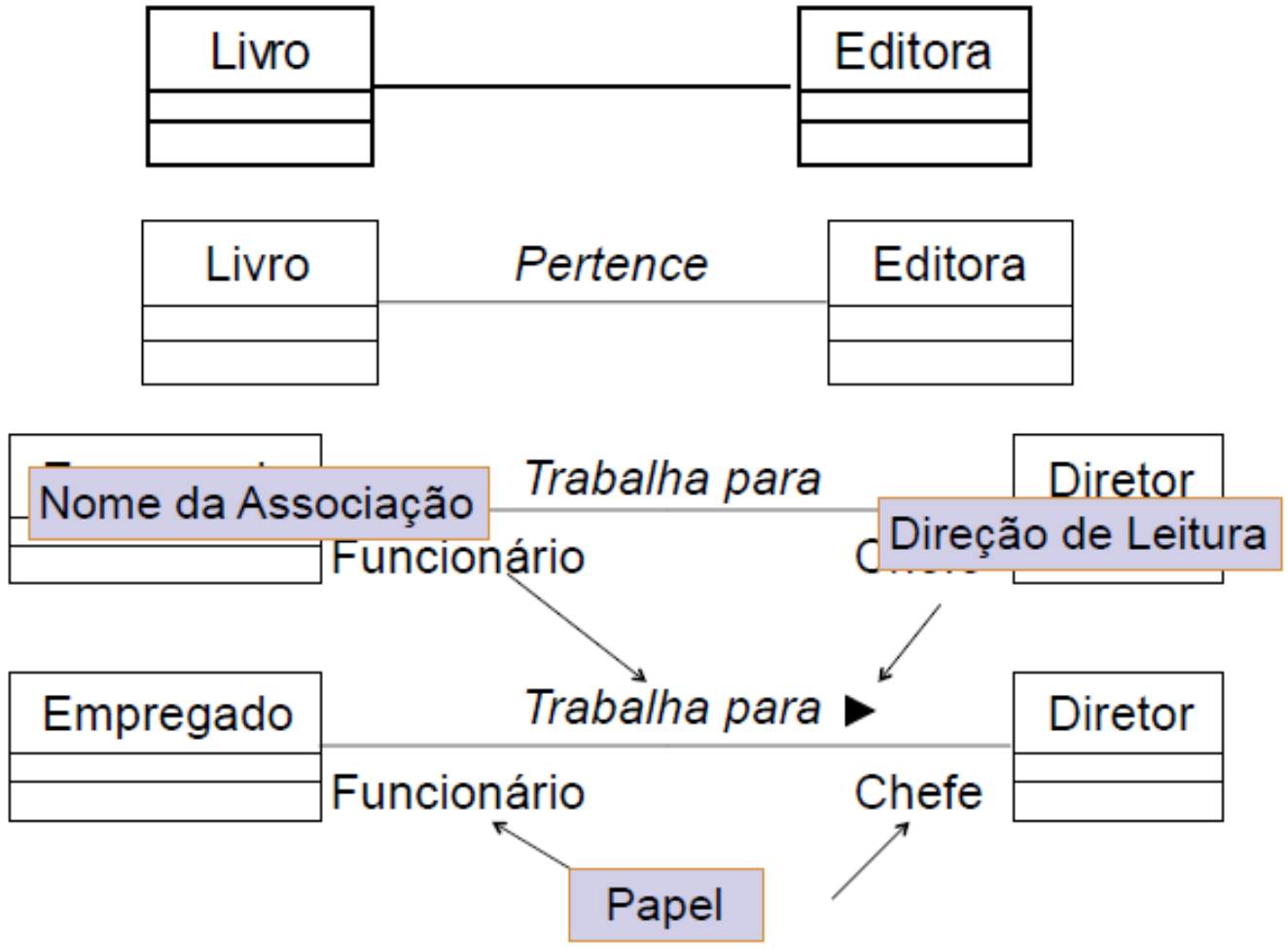
Classes: Relacionamentos

Relacionamentos: **Associação**

- Associações Normais (binárias)
- Tipo mais comum de associação, sendo uma conexão entre duas classes;
- Uma associação pode possuir um nome, que normalmente é um verbo (substantivos são aceitos);
- Cada terminação da associação pode possuir um papel, mostrando como uma classe é vista pela outra;

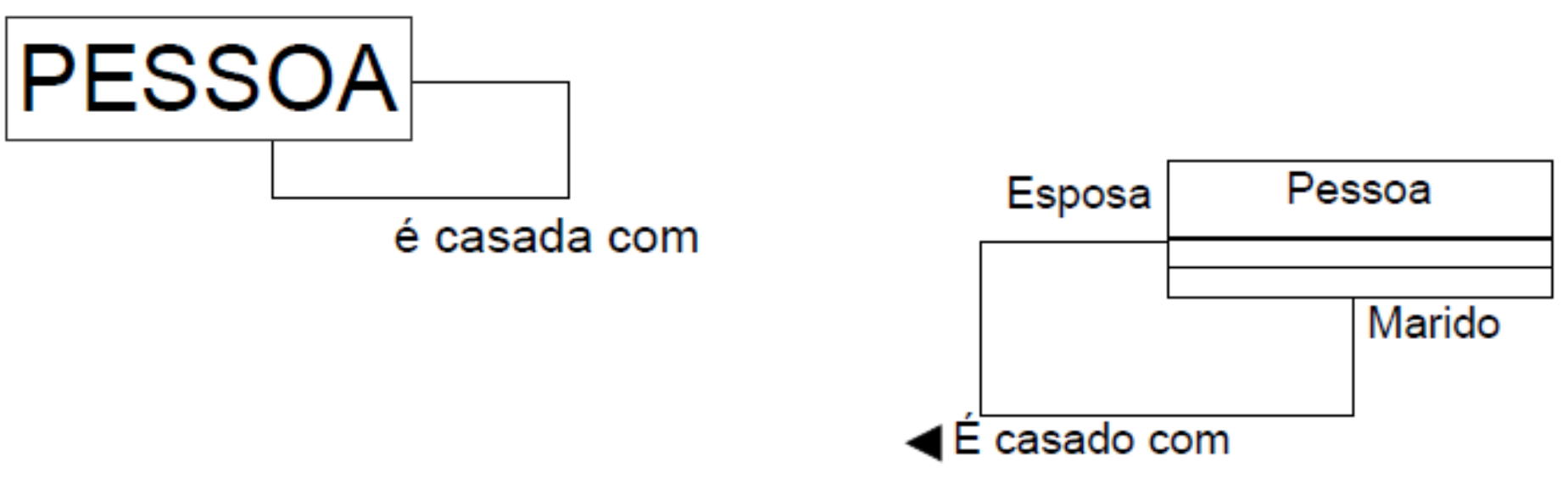
Classes: Relacionamentos

- Associações Normais



Classes: Relacionamentos

- Associações Binária Recursiva (reflexiva, autoassociação)
- Associação entre dois objetos, sendo que os mesmos pertencem a mesma classe;



Relacionamentos: Associação

A multiplicidade especifica quantos objetos de uma classe relacionam-se a um único objeto de classe associada;

Indicadores de multiplicidade:

– 1 Exatamente um

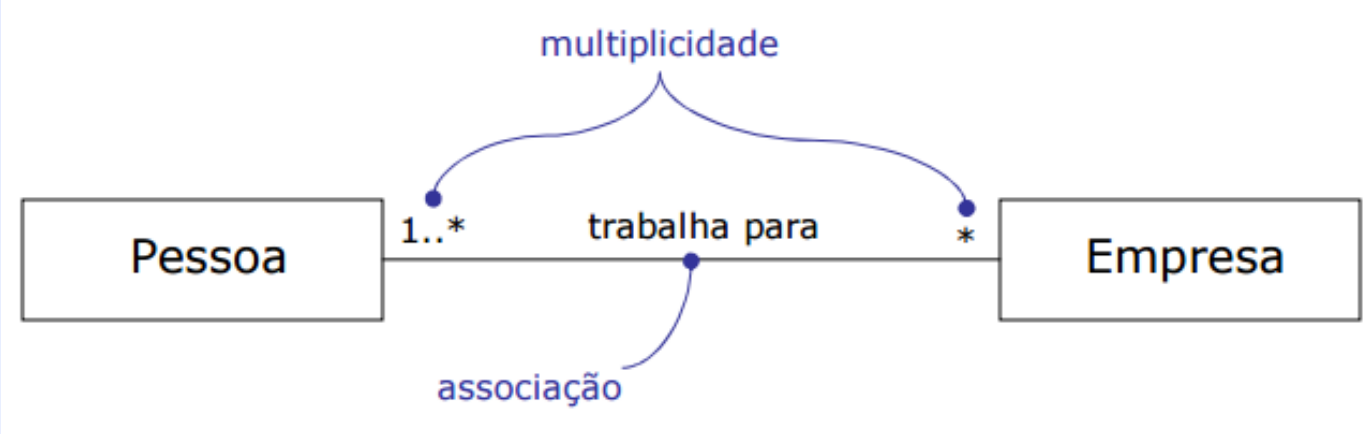
– 1..* Um ou mais

– 0..* Zero ou mais (muitos)

– * Zero ou mais (muitos)

– 0..1 Zero ou um

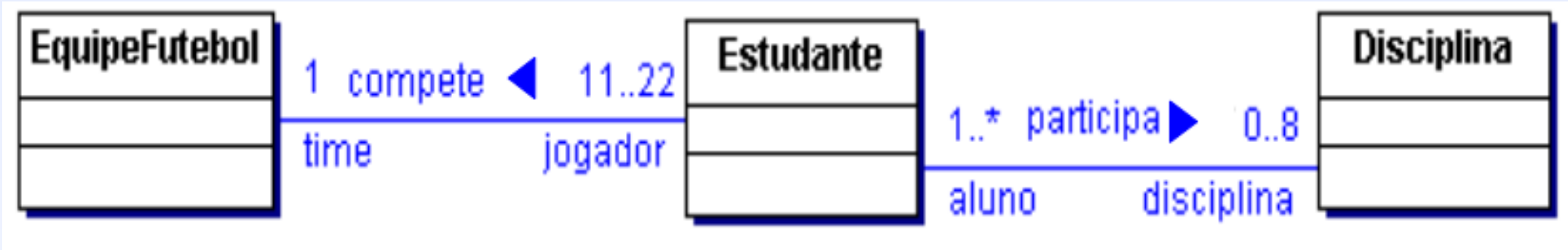
– m..n Faixa de valores (por exemplo: 4..7)



Classes: Relacionamentos

Relacionamentos: **Associação**

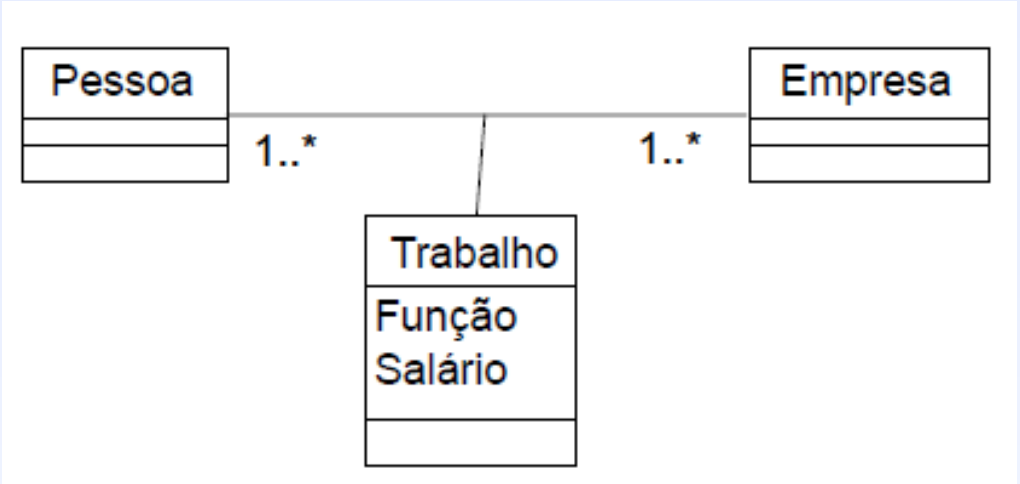
- Exemplo:
- Um Estudante pode ser um aluno de uma Disciplina e um jogador da Equipe de Futebol
- Cada Disciplina deve ser cursada por no mínimo 1 aluno
- Um aluno pode cursar de 0 até 8 disciplinas



Classes: Relacionamentos

Classe Associativa

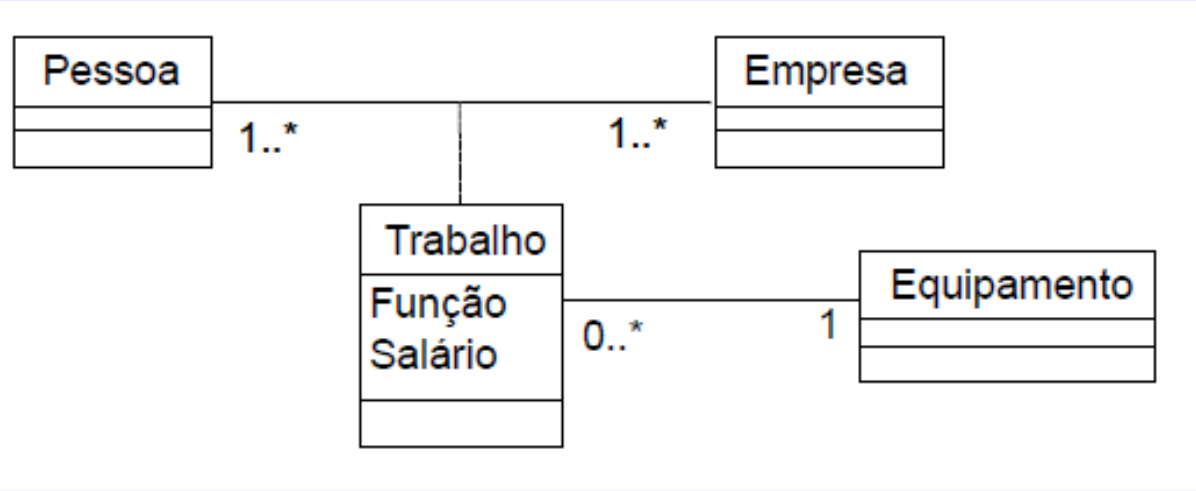
- Uma classe associativa é uma associação com propriedades de classe;
- Este tipo de classe surge quando duas ou mais classes estão associadas e é necessário manter informações sobre a associação;



Classes: Relacionamentos

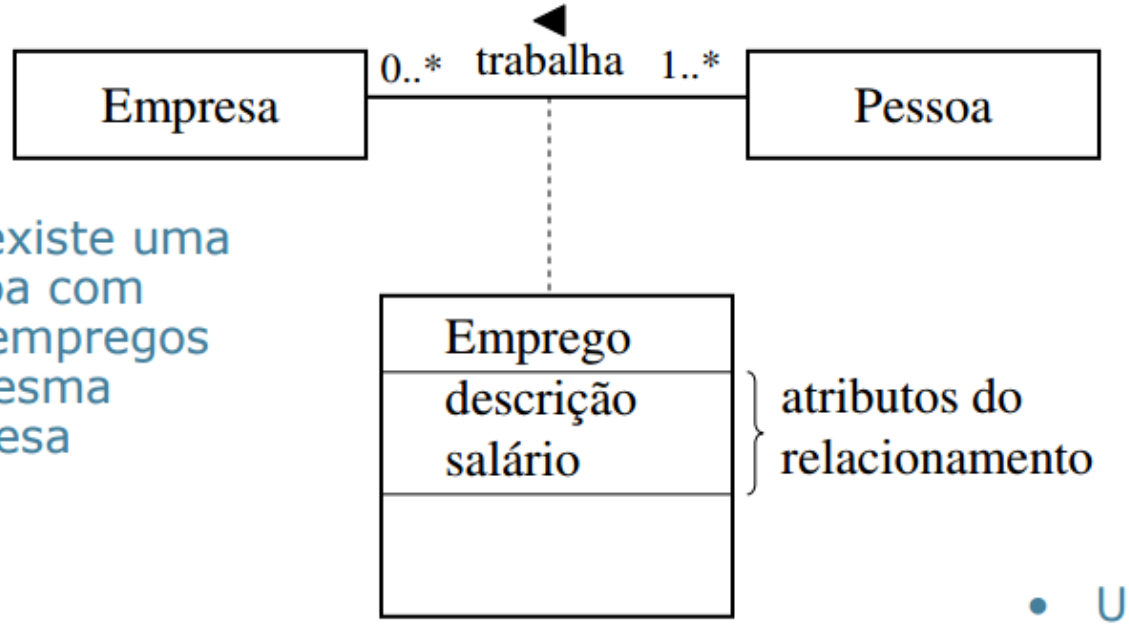
Classe Associativa

- Não se deve nomear a linha de associação de uma classe associativa;
- Somente o nome escolhido para a classe associativa deve ser suficiente para expressar o significado desejado;
- Uma classe associativa pode participar de outros relacionamentos;



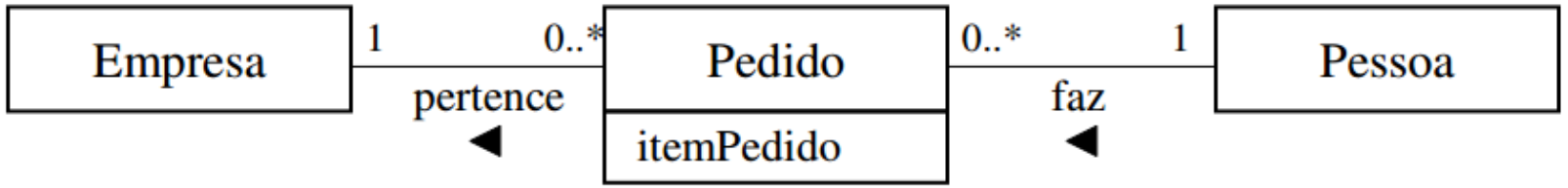
Classes: Relacionamentos

- Classe de associação



- Não existe uma pessoa com dois empregos na mesma empresa

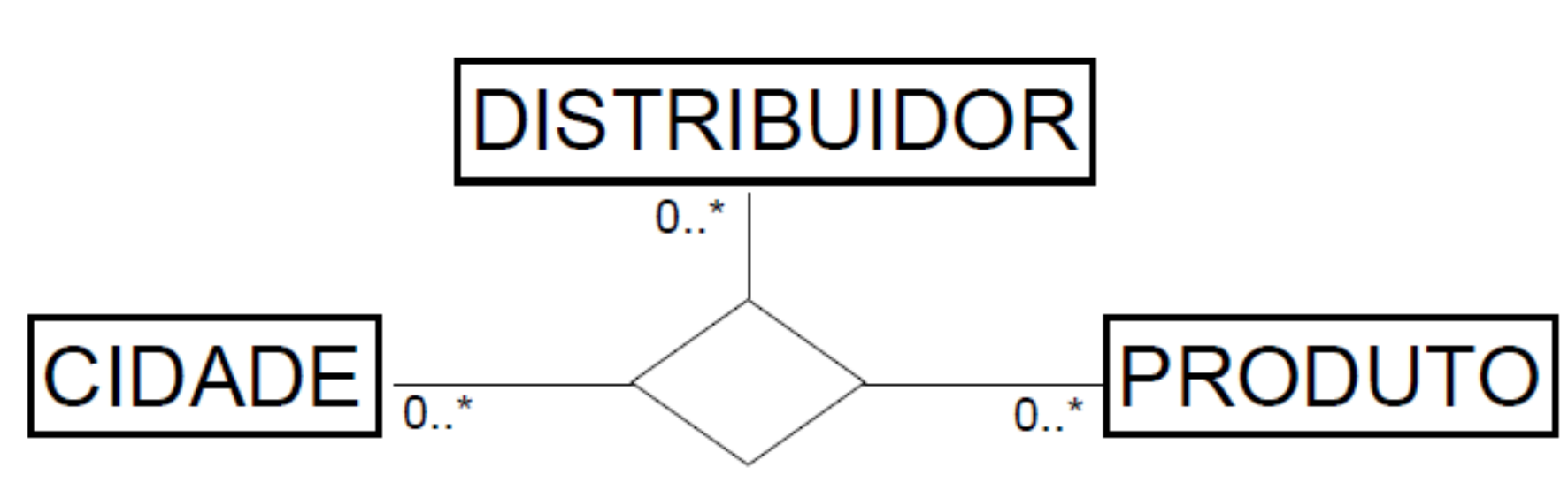
- Uma pessoa pode fazer mais de um pedido na mesma empresa



Classes: Relacionamentos

Associações Ternárias

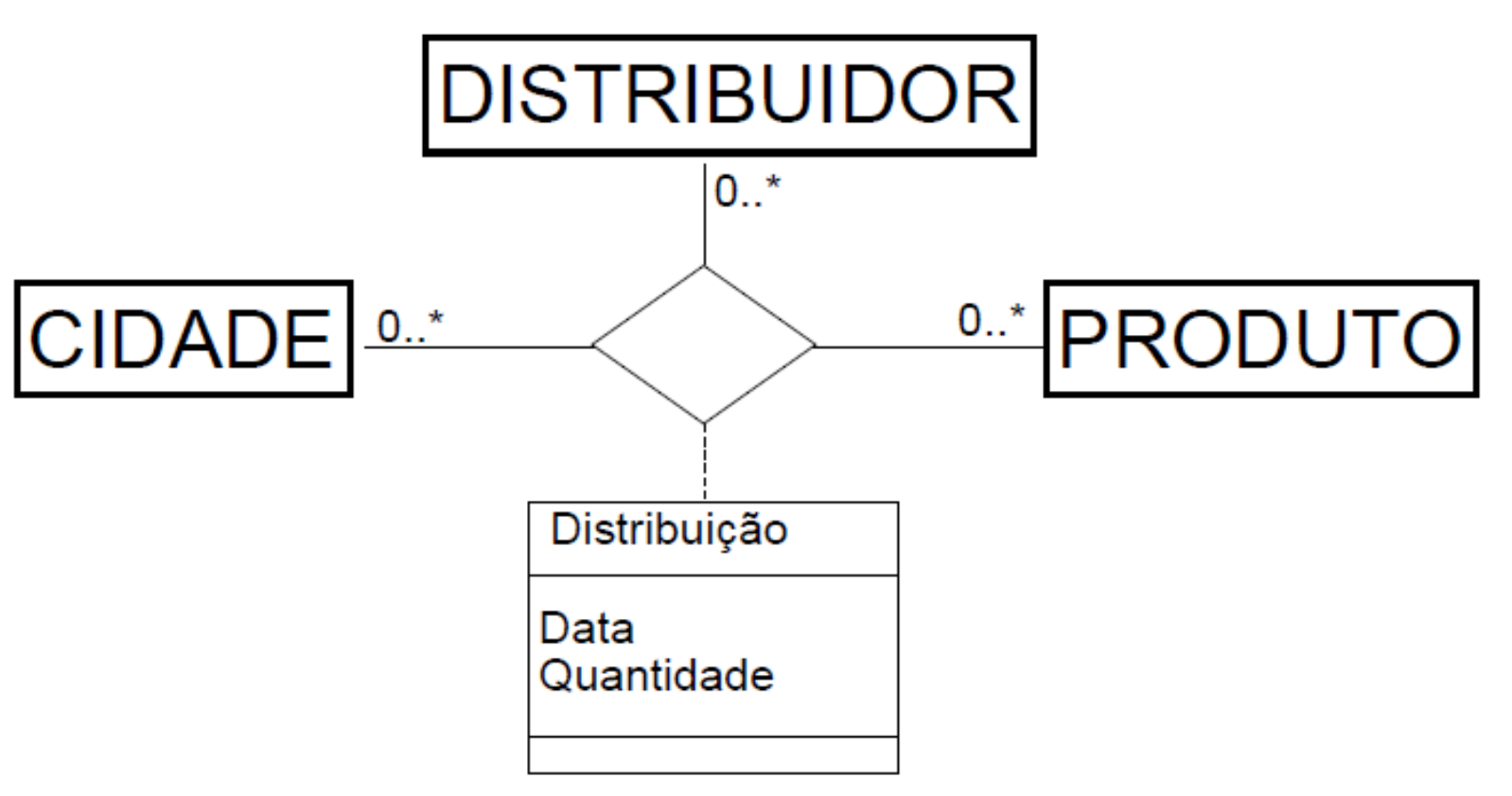
- Quando três classes estão associadas entre si;
- Representada por um losango, ligando cada classe do relacionamento;



Classes: Relacionamentos

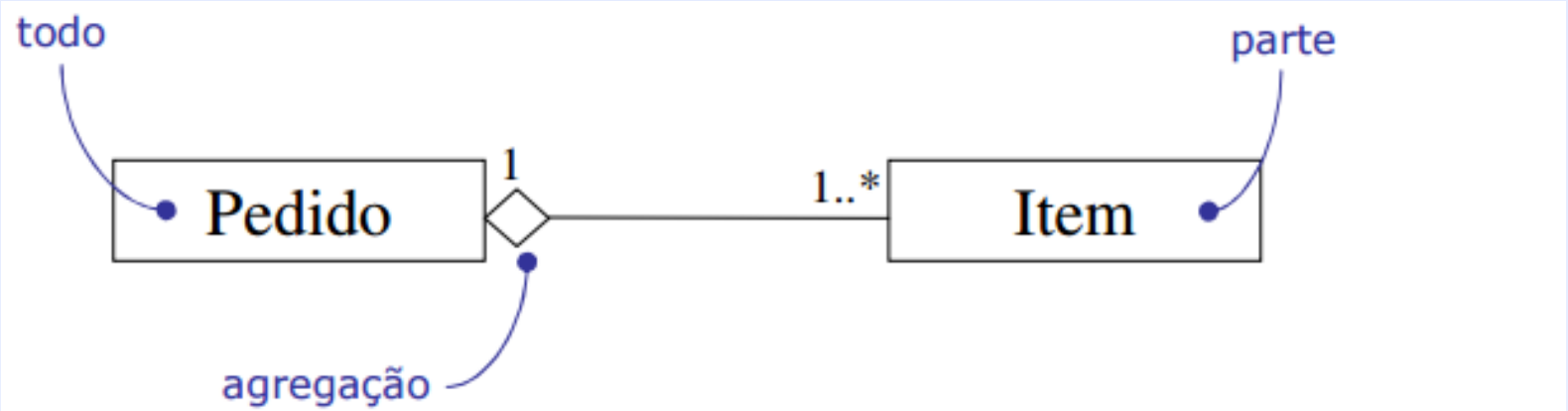
Associações Ternárias

- Pode ter uma classe associada na ternária



Relacionamentos: Agregação

- É um tipo especial de associação
- Indica que uma das classes do relacionamento é uma parte de outra classe – relação *todo-parte*;



– um objeto "parte" pode fazer parte de vários objetos "todo"

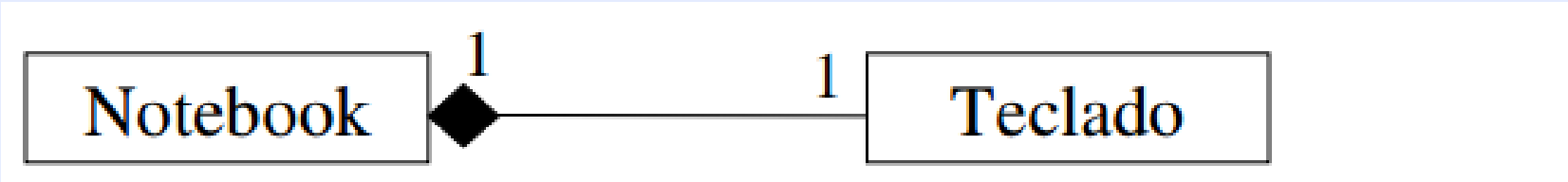
Relacionamentos: Agregação

- A diferença entre associação e agregação é puramente semântica - em uma agregação um objeto está contido no outro;
- Onde se puder utilizar uma agregação, pode-se usar também
- uma associação;
- Palavras Chave: *“consiste em”, “é parte de”;*
- Graficamente representada por uma linha conectando as classes relacionadas, com um diamante (losango) branco perto da classe que representa o todo.

Relacionamentos: Composição

– É uma variante semanticamente mais “forte” da agregação.

– Os objetos “parte” só podem pertencer a um <u>único</u> objeto “todo” e têm o seu tempo de vida coincidente com o dele.

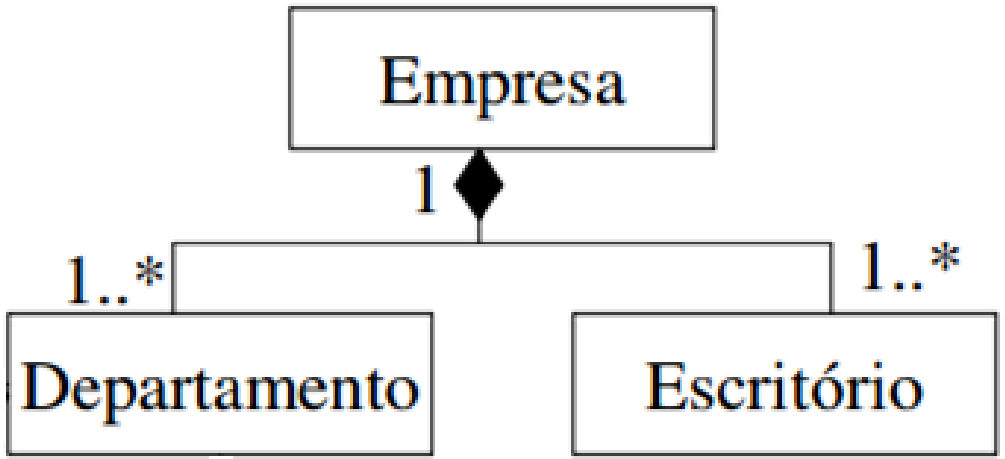


errado

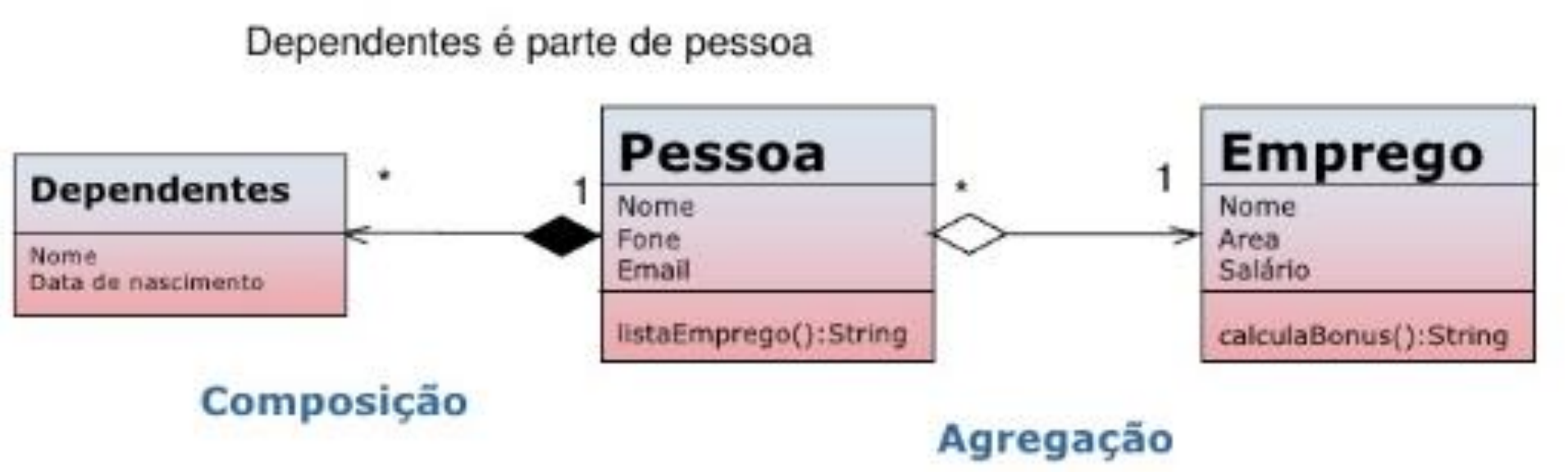
– Quando o “todo” *morre* todas as suas “partes” também *morrem*

Relacionamentos: Composição

Ex:

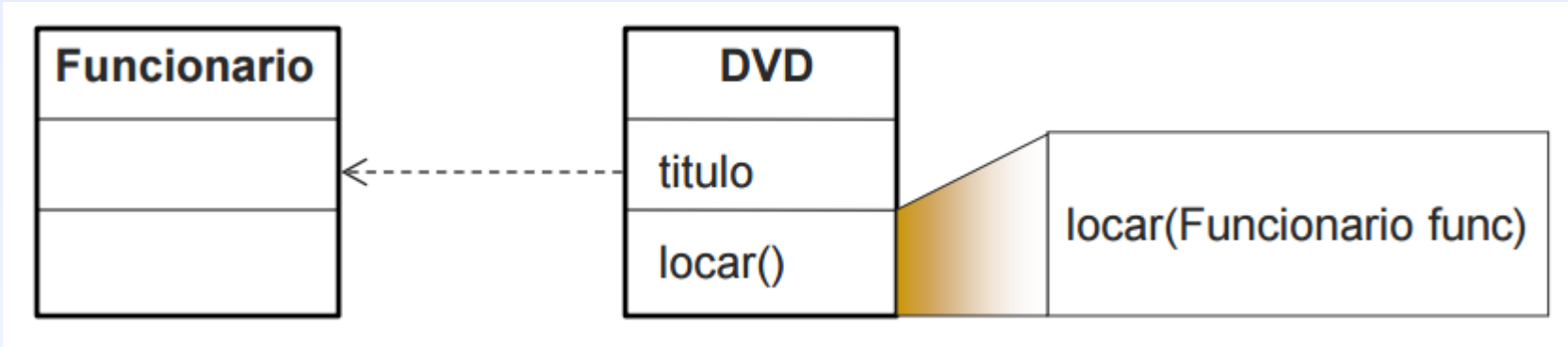


Agregação e Composição



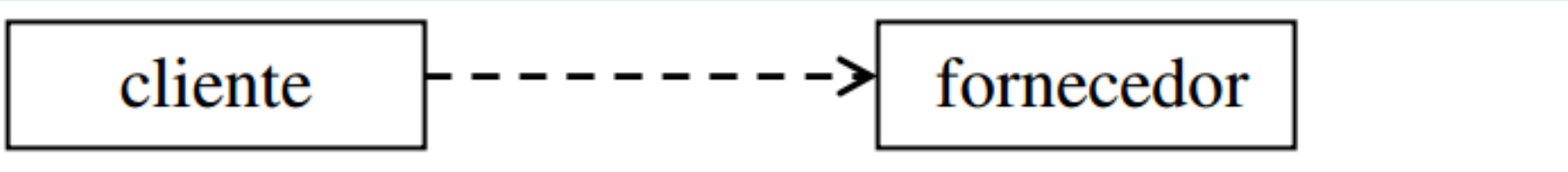
Relacionamentos: Dependência

- Tipo menos comum de relacionamento
- Identifica uma ligação fraca entre objetos de duas classes
- Representado por uma reta tracejada entre duas classes
- Uma seta na extremidade indica o dependente



Relacionamentos: Dependência

- Representa que a alteração de um objeto (o objeto independente) pode afetar outro objeto (o objeto dependente)

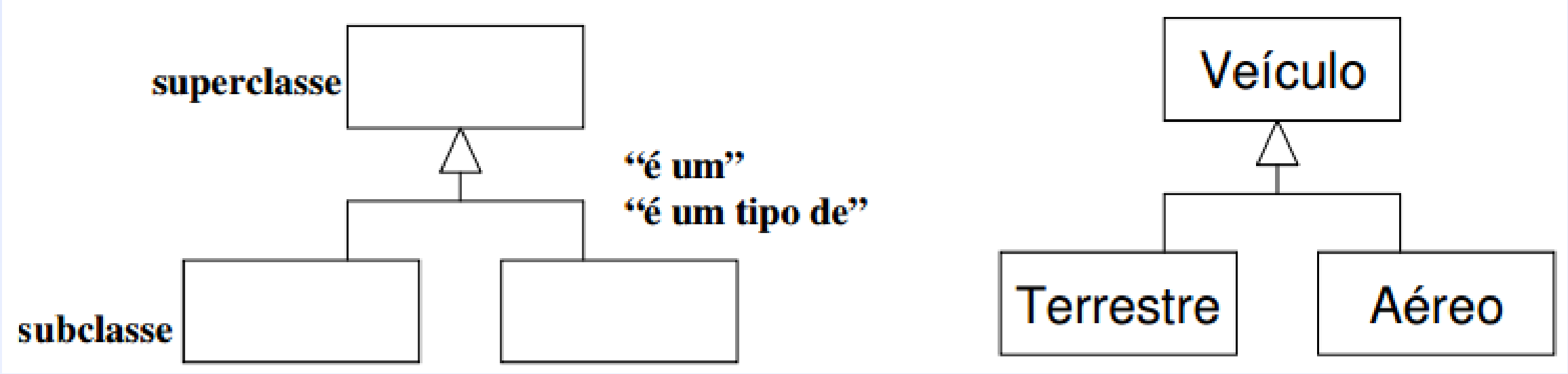


Obs:

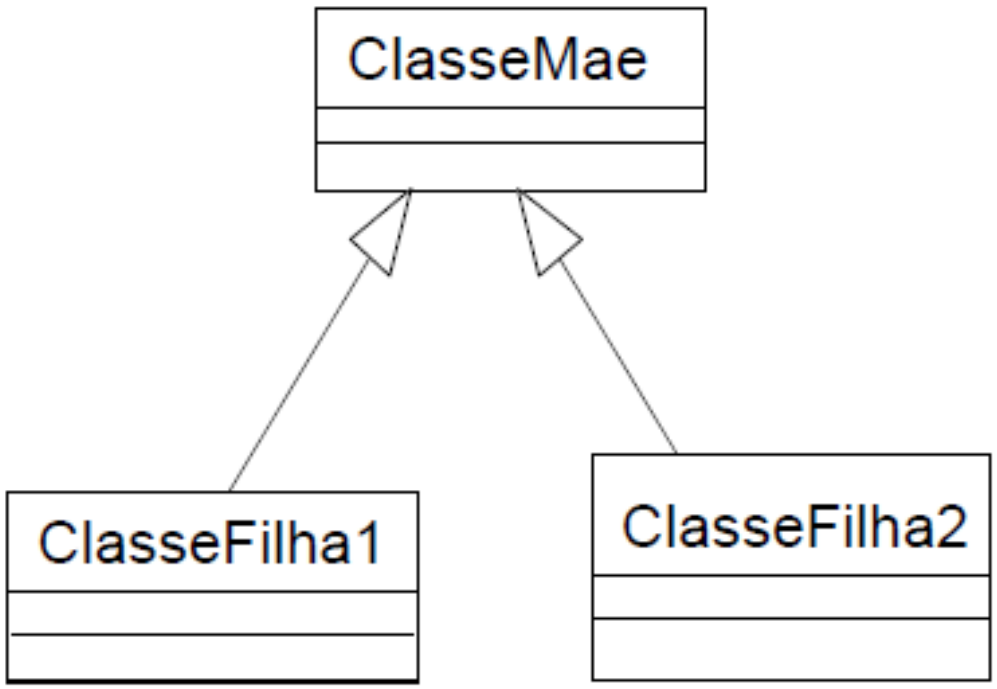
- A classe cliente depende de algum serviço da classe fornecedor
- A mudança de estado do fornecedor afeta o objeto cliente
- A classe cliente não declara nos seus atributos um objeto do tipo fornecedor
- Fornecedor é recebido por parâmetro de método

Relacionamentos: Generalização

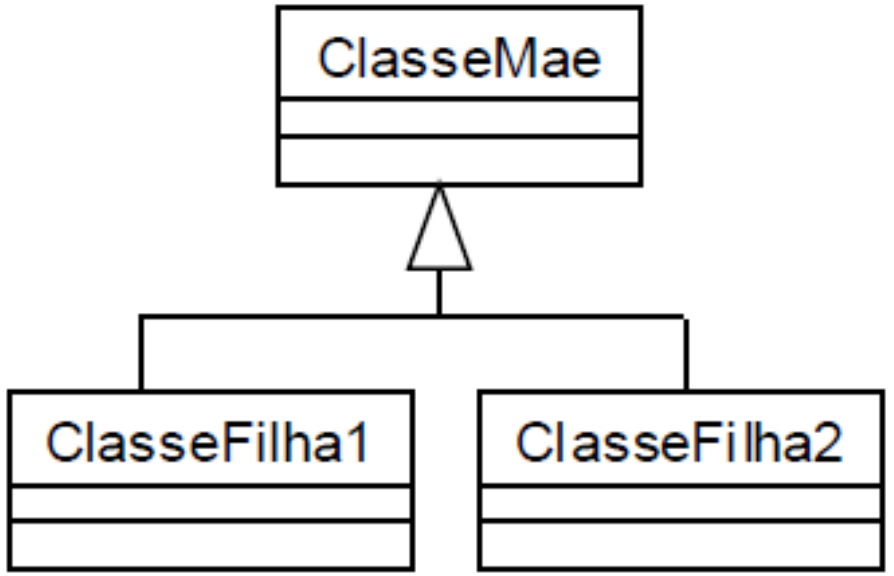
- É um relacionamento entre itens gerais (superclasses) e itens mais específicos (subclasses)



Relacionamentos: Generalização



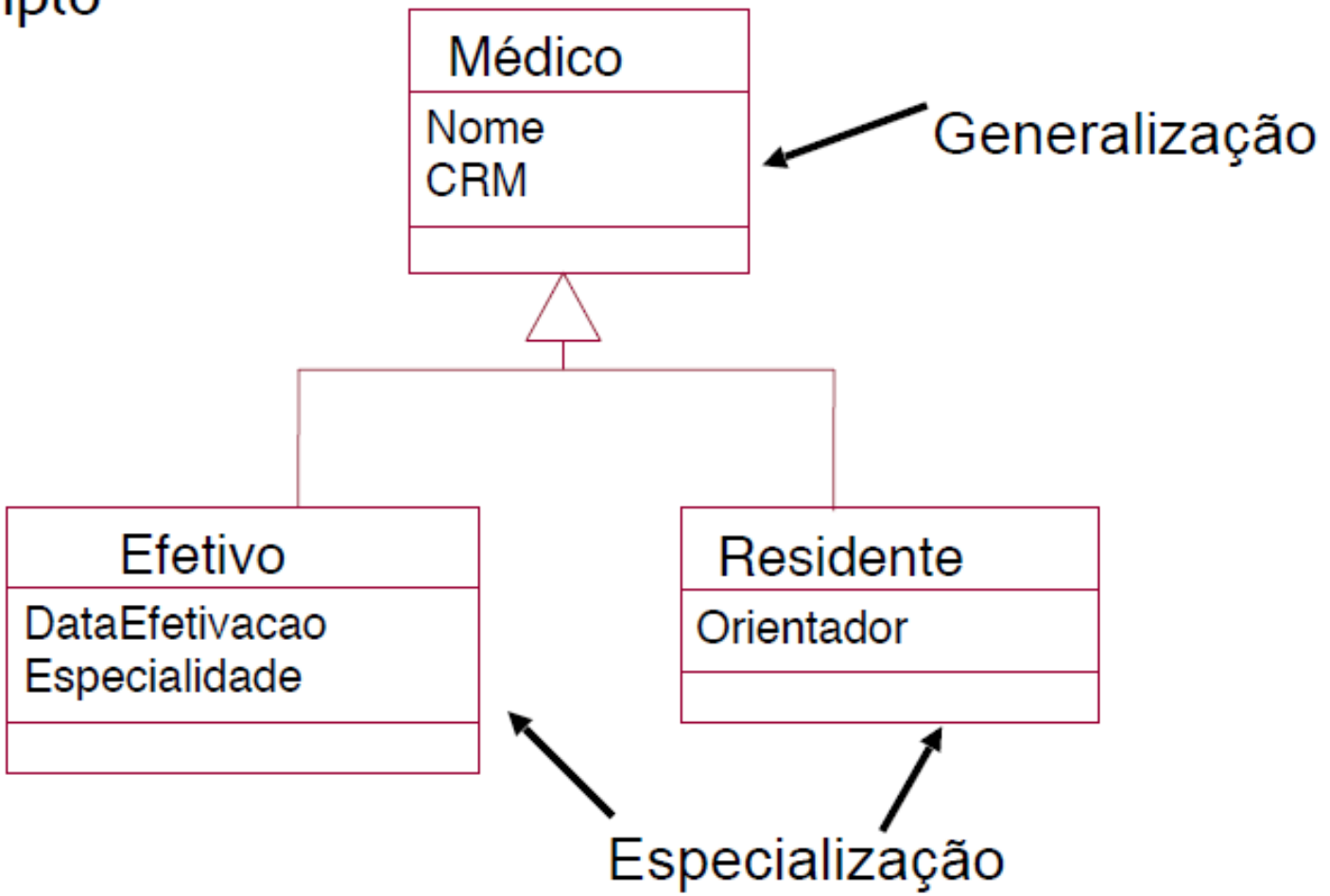
Estilo com setas separadas



Estilo com setas Agrupadas

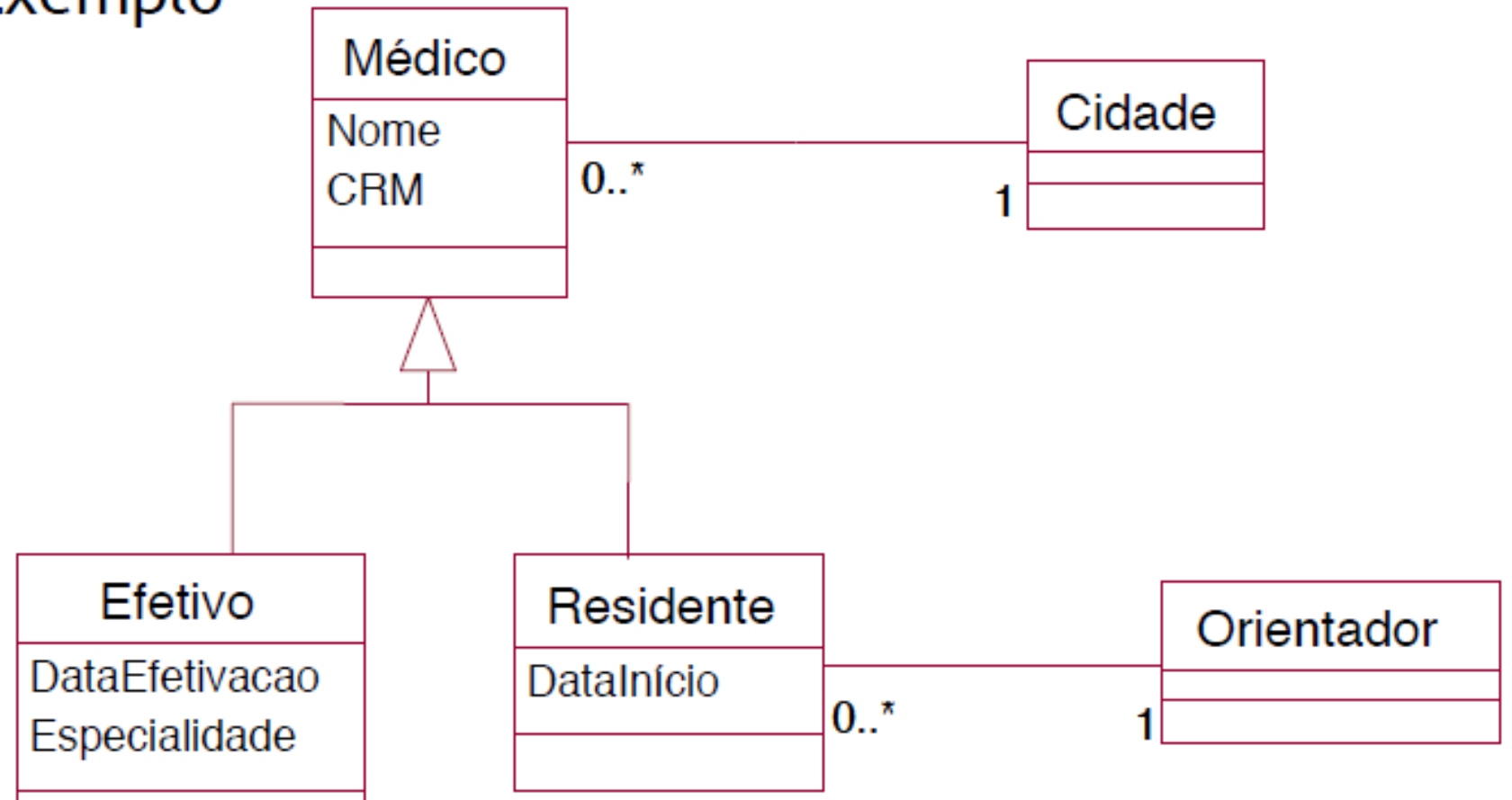
Relacionamentos: Generalização

- Exemplo



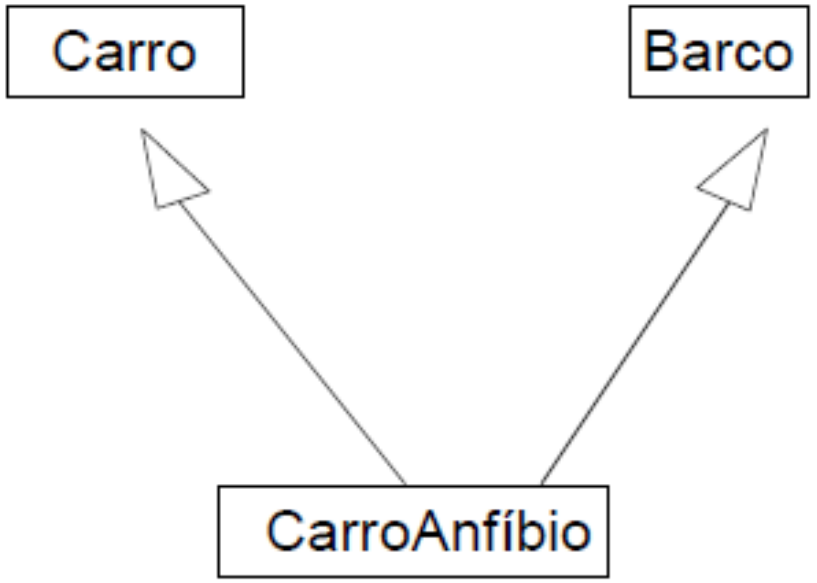
Relacionamentos: Generalização

- Exemplo



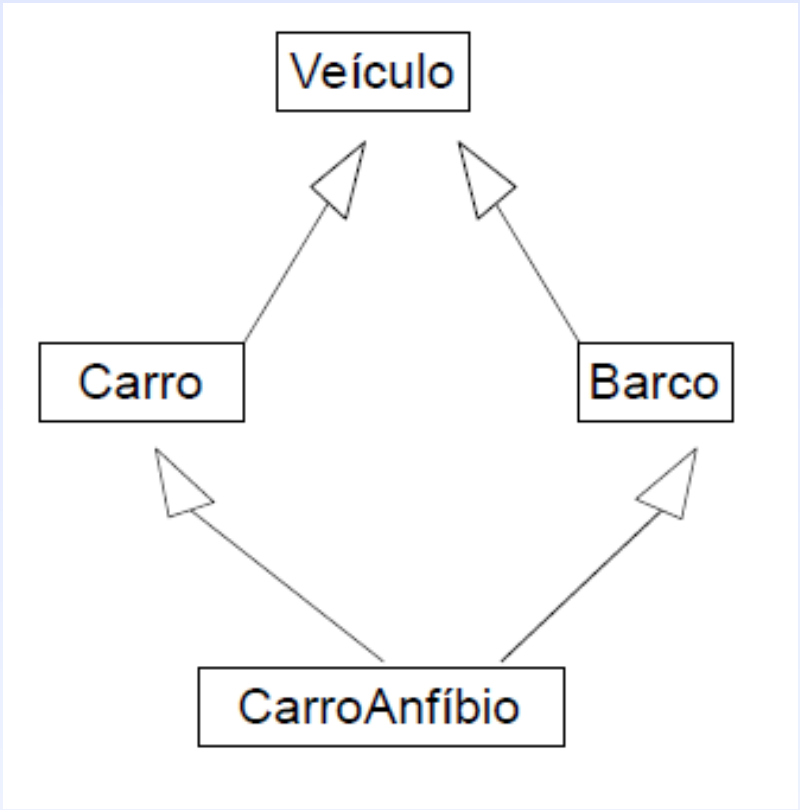
Relacionamentos: Generalização

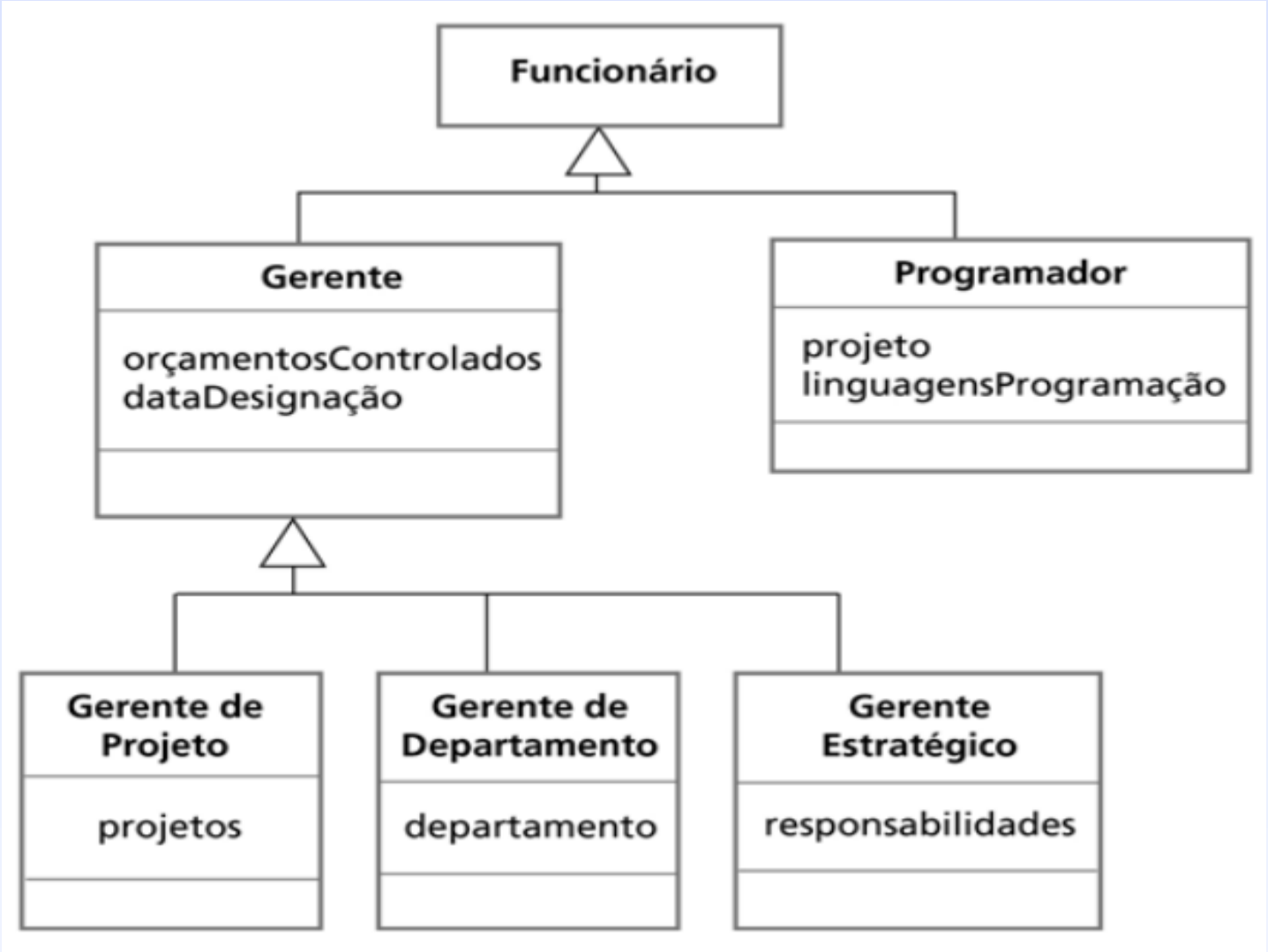
- Uma classe pode ter mais de uma superclasse (herança múltipla);



Relacionamentos: Generalização

Níveis de especialização





Relacionamentos: Generalização

Generalização – Restrições

- A UML permite que determinadas restrições sejam associadas a elementos de um modelo
- As restrições sobre generalizações são representadas no diagrama de classes, próximas à linha do relacionamento
- Apresentadas entre chaves, com um pontilhado;

Relacionamentos: Generalização

Generalização – Restrições

- **Sobreposta:** um objeto pode pertencer a mais de um tipo de classe. Posteriormente, podem ser criadas subclasses que herdem de mais de uma subclasse (herança múltipla)
- **Disjunta:** um objeto pertence a apenas um tipo de classe. Quaisquer subclasses criadas posteriormente poderão herdar de somente uma subclasse
- **Completa:** todas as subclasses possíveis foram enumeradas na hierarquia
- **Incompleta:** nem todas as subclasses foram enumeradas na hierarquia

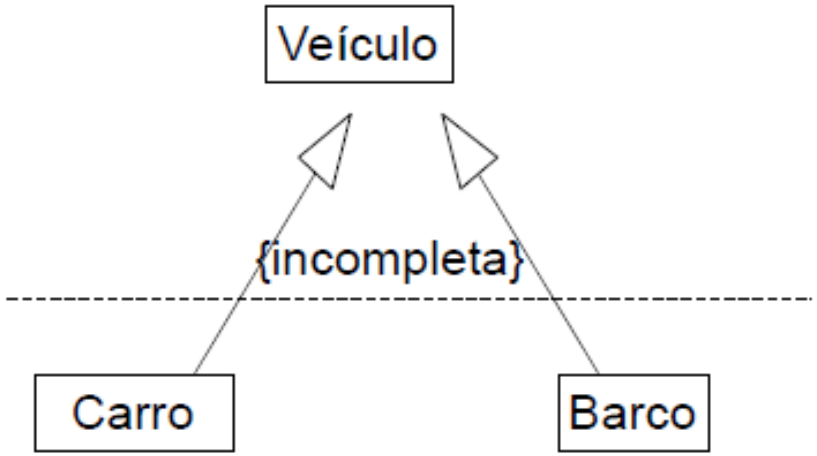
Relacionamentos: Generalização

Generalização – Restrições

- As duas primeiras restrições (sobreposta e disjunta) e as duas últimas (completa e incompleta) são exclusivas entre si;
    - Não há sentido em definir subclasses que sejam sobrepostas e disjuntas ao mesmo tempo;
    - Não há sentido em definir subclasses que sejam completas e incompletas ao mesmo tempo;

Relacionamentos: Generalização

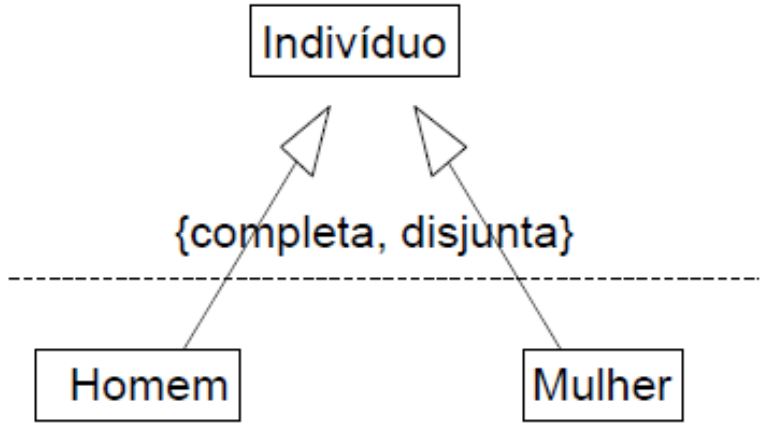
- Generalização - Restrições - Exemplos



Há outras subclasses de *Veículo* além das que foram enumeradas na hierarquia

Relacionamentos: Generalização

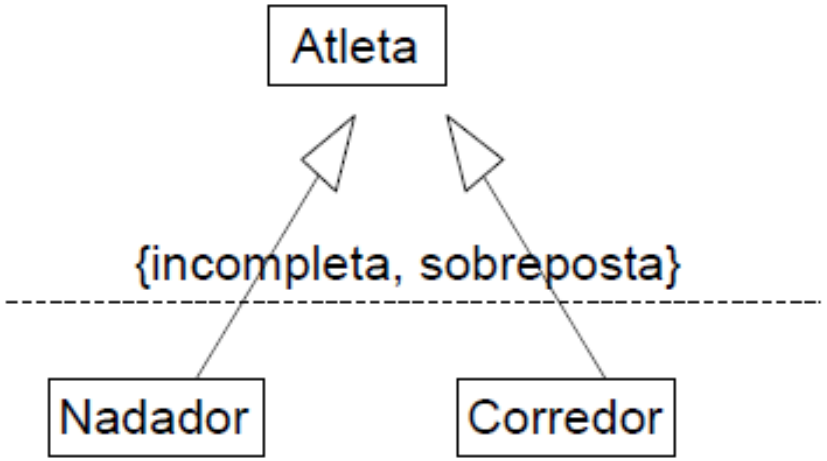
- Generalização - Restrições - Exemplos



1) não há objetos dentro desta hierarquia que não seja homem ou mulher (completa) e 2) não há objeto nesta hierarquia que seja homem e mulher ao mesmo tempo (disjunção)

Relacionamentos: Generalização

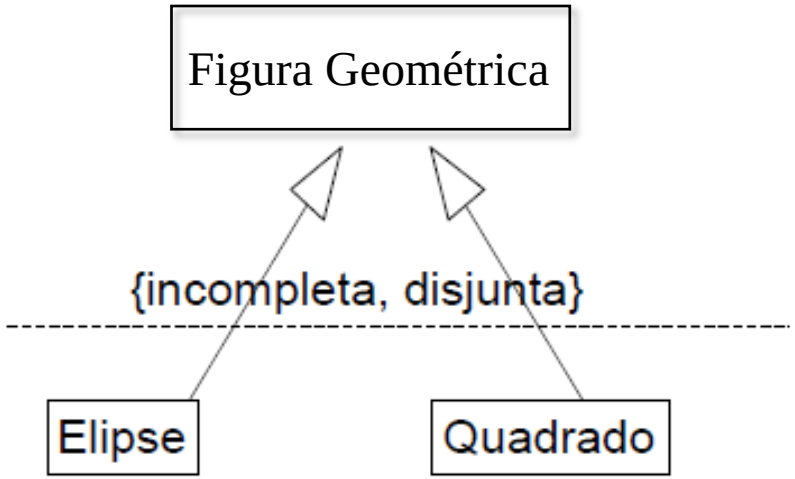
- Generalização - Restrições - Exemplos



1) há outras subclasses de *Atleta* não enumeradas na hierarquia (incompleta) e 2) um *Atleta* pode ser tanto *Nadador* quanto *Corredor* (sobreposta)

Relacionamentos: Generalização

- Generalização - Restrições - Exemplos

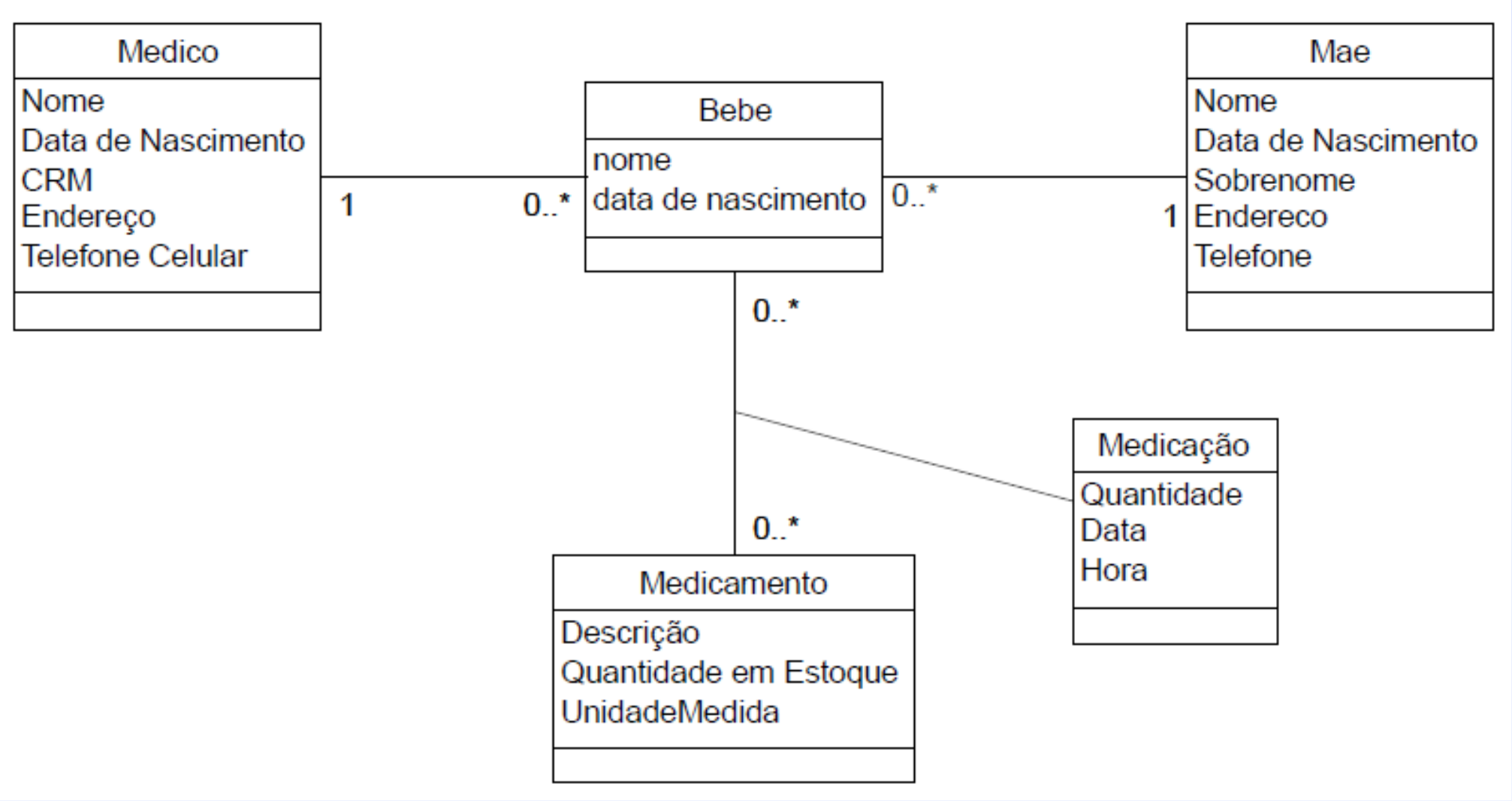


1) há outras subclasses de *FiguraGeométrica* não enumeradas na hierarquia (incompleta) e 2) uma *FiguraGeométrica* não pode ser do tipo *Elipse* e *Quadrado* ao mesmo tempo.

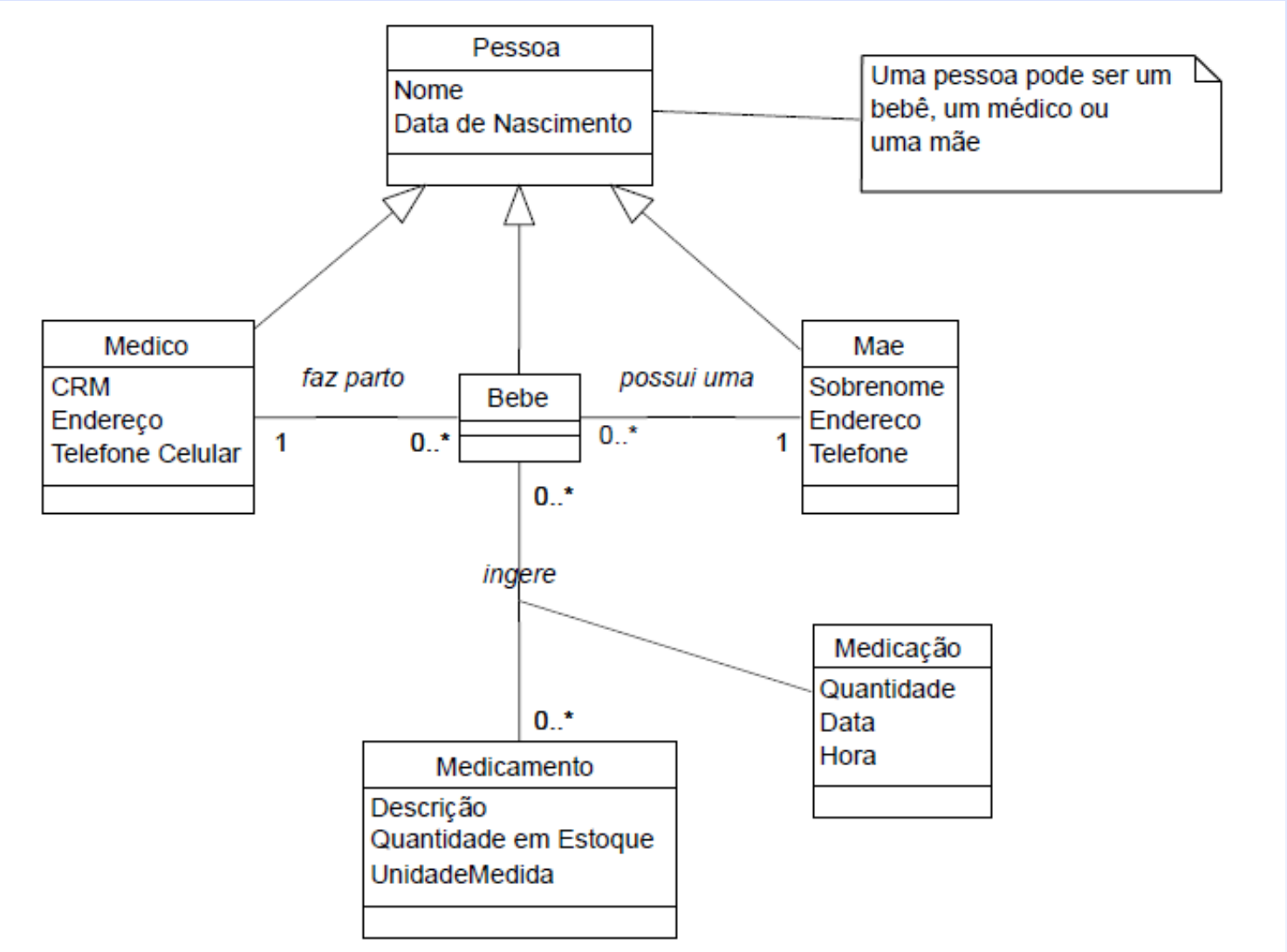
Exemplo Berçário

- Um berçário deseja controlar suas tarefas. Para isso deseja manter um cadastro para os **bebês**, constando **nome, médico que fez seu parto, sua mãe e data de nascimento.** Para os **médicos** é necessário saber: **o nome, data de nascimento, CRM, endereço e telefone celular**. Para as **mães** dos bebês é necessário manter informações como: **nome, sobrenome, data de nascimento, endereço e telefone**. Ainda, é necessário manter um controle dos **medicamentos** ingeridos pelos bebês no berçário. Sobre os medicamentos é necessário manter: **descrição, quantidade em estoque e unidade de medida**. Um bebê pode tomar vários medicamentos, assim como um medicamento pode ser dado para vários bebês. Quando um bebê toma uma **medicação** ainda é importante saber a **quantidade, o dia e a hora do medicamento**.

Exemplo Berçário



Exemplo Berçário



Exercício

Construa um diagrama de classes para os seguintes objetos. Se quiser, pode incluir novas classes. Utilize herança, agregação e demais conceitos.

- Escola, computador, playground, livro, porta, laboratório, salas, biblioteca, régua, alunos, giz, balanço, revista, janela, sala de aula, professor, carteira, sala dos professores, cadeira, teclado, monitor, escorregador, bibliotecário, quadro-negro, secretária, boletim
- Casa, cozinha, sofá, banheiro, panela, armário, lâmpada, geladeira, janela, sala, cama, pia, sala de estar, fogão, tampa da panela, sistema de alarme, abajur, sala de jantar, porta, mesa, sistema de incêndio, pão, leite, sabonete, cabo da panela

Diagrama de Objetos

Além do diagrama de classes, A UML define um segundo tipo de diagrama estrutural, o diagrama de objetos.

- Pode ser visto com uma instância de diagramas de classes
- Representa uma “fotografia” do sistema em um certo momento.
- Exibe as ligações formadas entre objetos conforme estes interagem e os valores dos seus atributos.
- Os diagramas de objetos são uma contribuição do método de Booch;

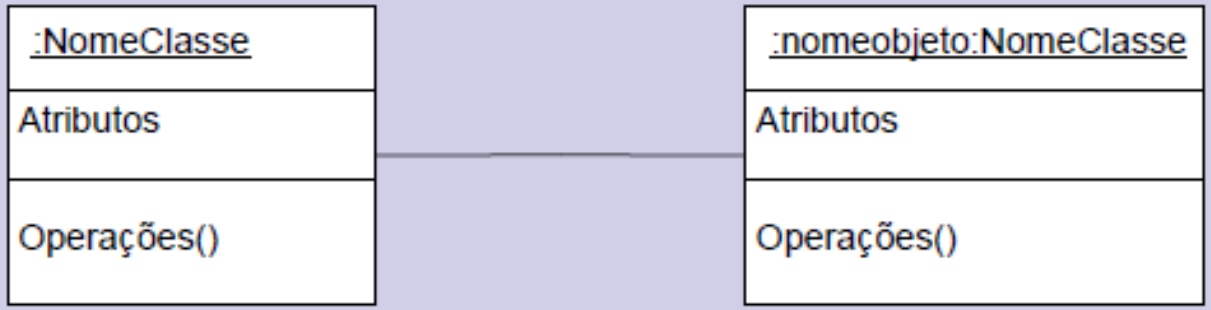
Diagrama de Objetos

“Um objeto é qualquer coisa, real ou abstrata, sobre a qual armazenamos dados e operações que manipulam os dados.”

Uma fatura, uma organização, uma forma em um programa para desenhar, uma tela com a qual o usuário interage, um mecanismo em um dispositivo robótico, uma planta inteira de engenharia, um componente em uma planta de engenharia, um avião, uma reserva de passagem aérea, um ícone em uma tela, um processo de preenchimento de pedidos, entre outros..

Diagrama de Objetos

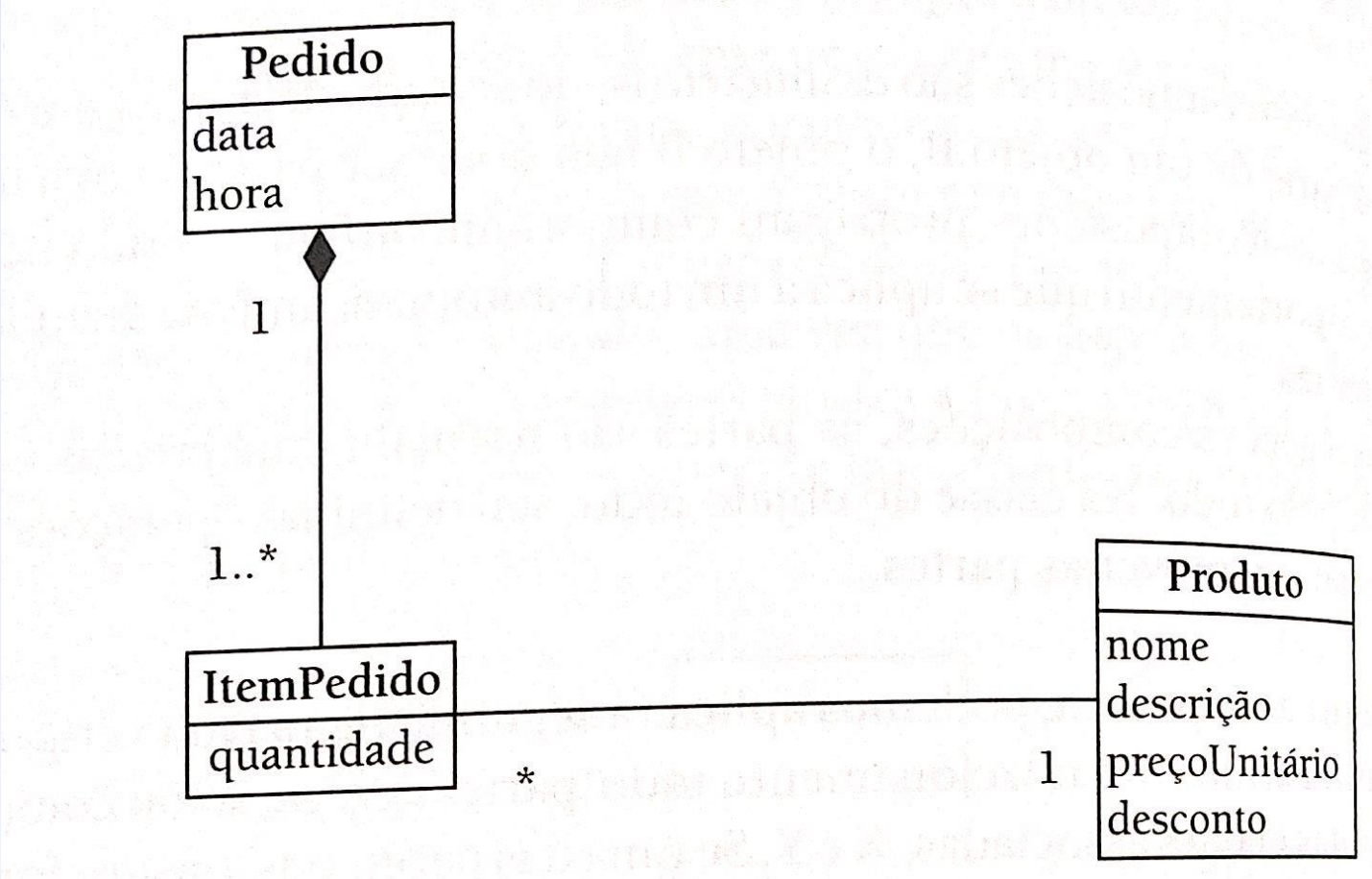
O Diagrama de Objetos é um gráfico de instâncias, incluindo objetos e valores de dados. É uma instância de um diagrama de classes. É considerado limitado, servindo de exemplo de estruturas de dados



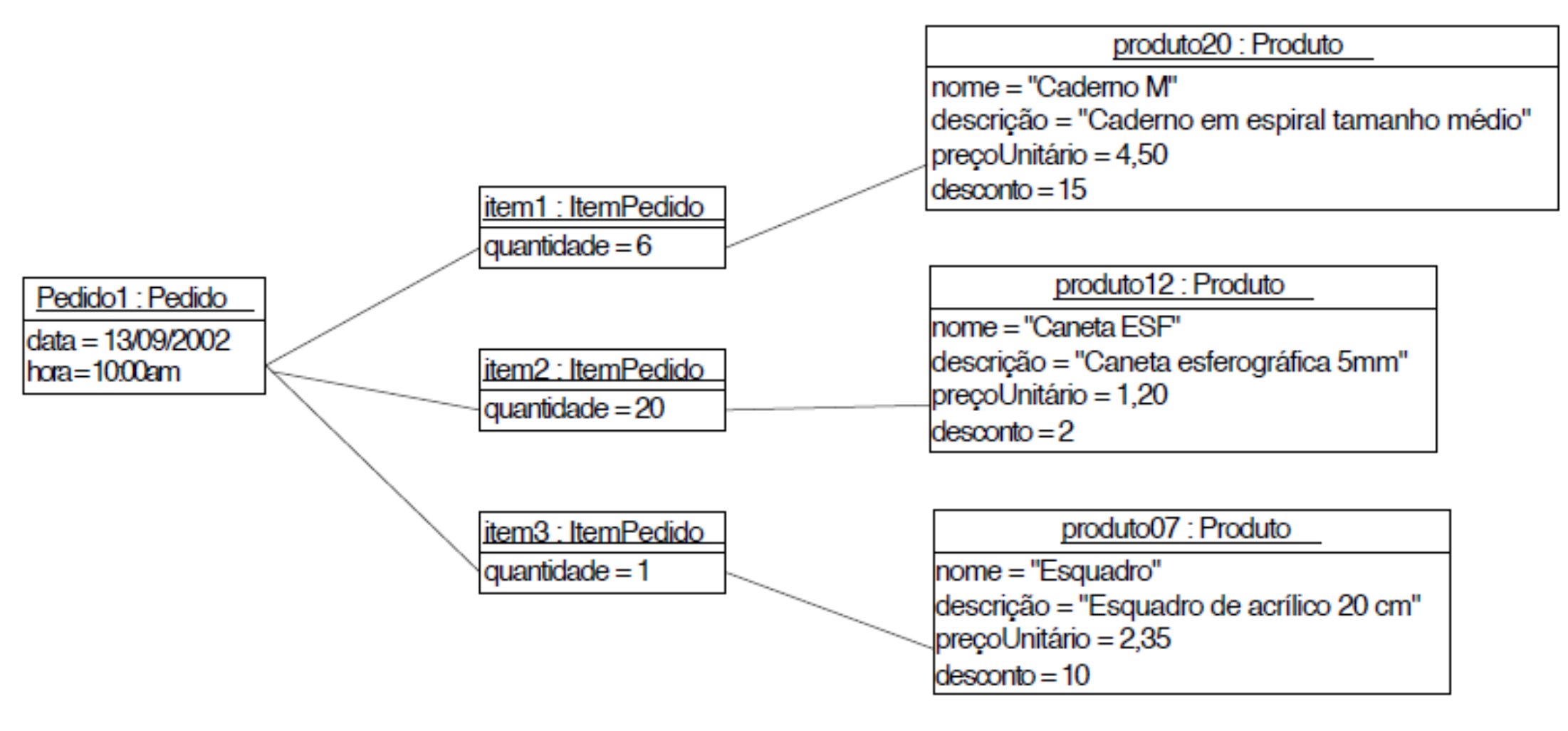
Notação do Diagrama de Objetos

Formato	Exemplo
nomeClasse	Pedido
nomeObjeto: NomeClasse	umPedido: Pedido

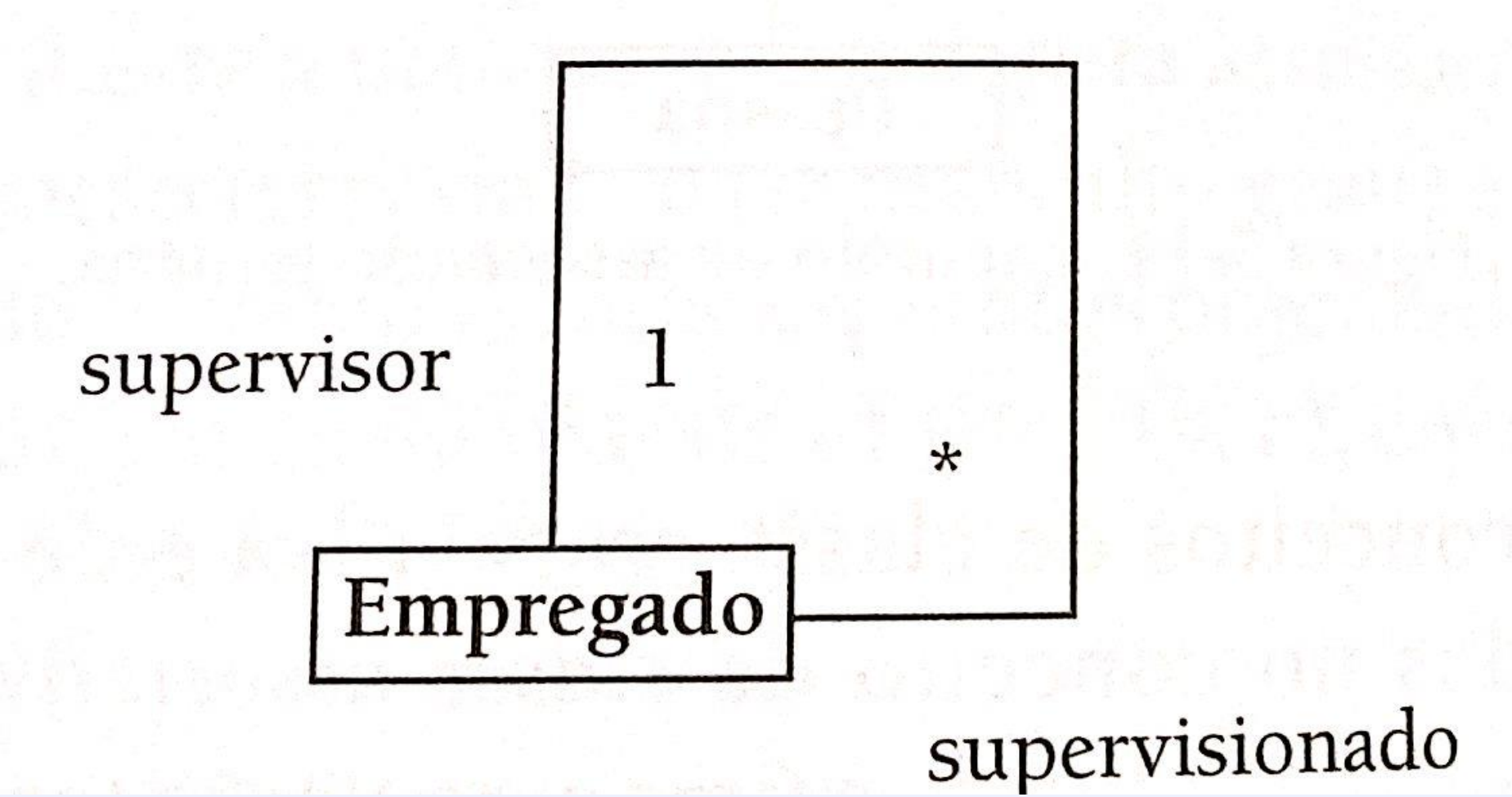
Exemplo de Diagrama de Objetos



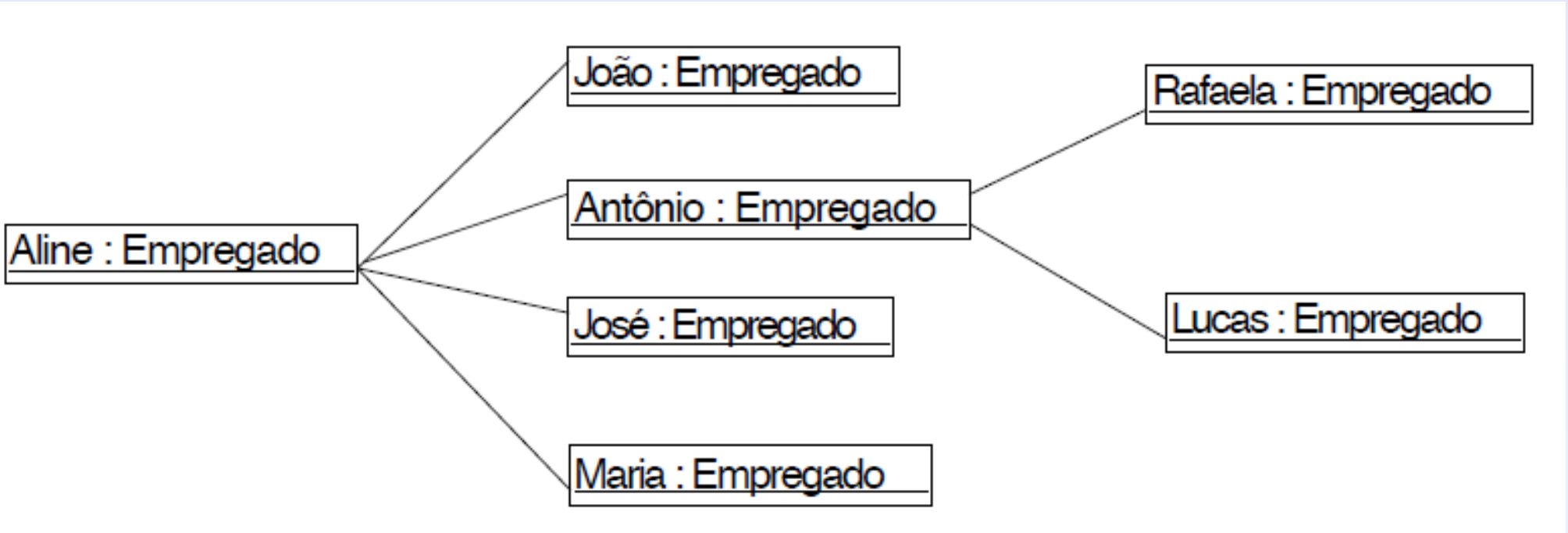
Exemplo de Diagrama de Objetos



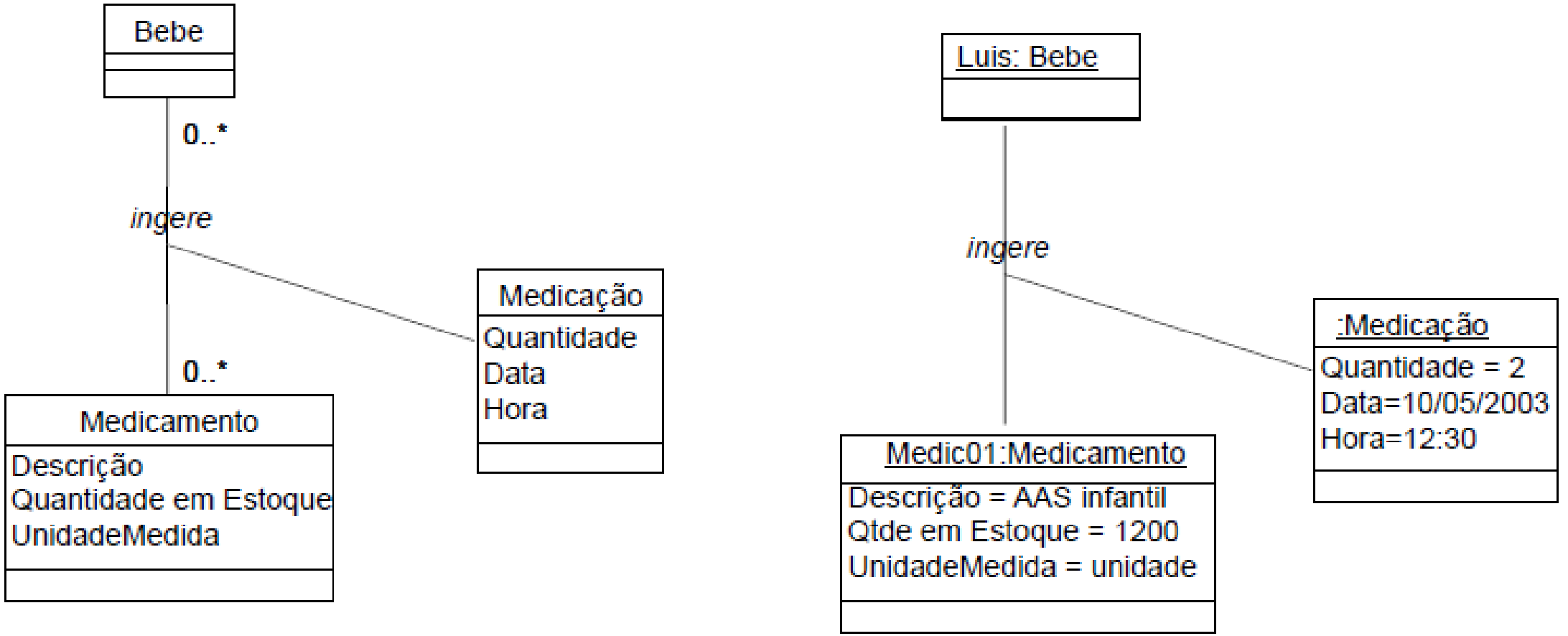
Exemplo de Diagrama de Objetos



Exemplo de Diagrama de Objetos



Exemplo de Diagrama de Objetos



Assistir ao vídeo

Tutorial para criação de diagrama de classes no Astah

https://www.youtube.com/watch?v=yLe4rhTG59I