

Análise e Projeto de Sistemas

Universidade Federal do Ceará – UFC
Campus de Quixadá
Curso de Sistemas de Informação
Prof. Marcos Antonio de Oliveira
(deoliveira.ma@gmail.com)

“O engenheiro de software amador está sempre à procura de mágica, de algum método sensacional ou ferramenta cuja aplicação promete tornar trivial o desenvolvimento de software. É uma característica do engenheiro de software profissional saber que tal panacéia não existe.”
(GRADY BOOCH)

MODELAGEM DE CLASSES DE ANÁLISE

Esses slides são uma adaptação das notas de aula do professor Eduardo Bezerra autor do livro Princípios de Análise e Projeto de Sistemas com UML

Índice

- Introdução
- Estágios do modelo de classes
- Diagrama de classes
- Diagrama de objetos
- Técnicas para identificação de classes
- Construção do modelo de classes
- Modelo de classes no processo I&I
- Estudo de caso

INTRODUÇÃO

Introdução

- O modelo de casos de uso fornece uma perspectiva do sistema a partir de um ponto de vista **externo**
- De posse da visão de casos de uso, os desenvolvedores precisam prosseguir no desenvolvimento do sistema

Introdução

- A funcionalidade externa de um sistema orientado a objetos é fornecida através de colaborações entre objetos

Externamente: os atores visualizam resultados de cálculos, relatórios produzidos, confirmações de requisições realizadas, etc.

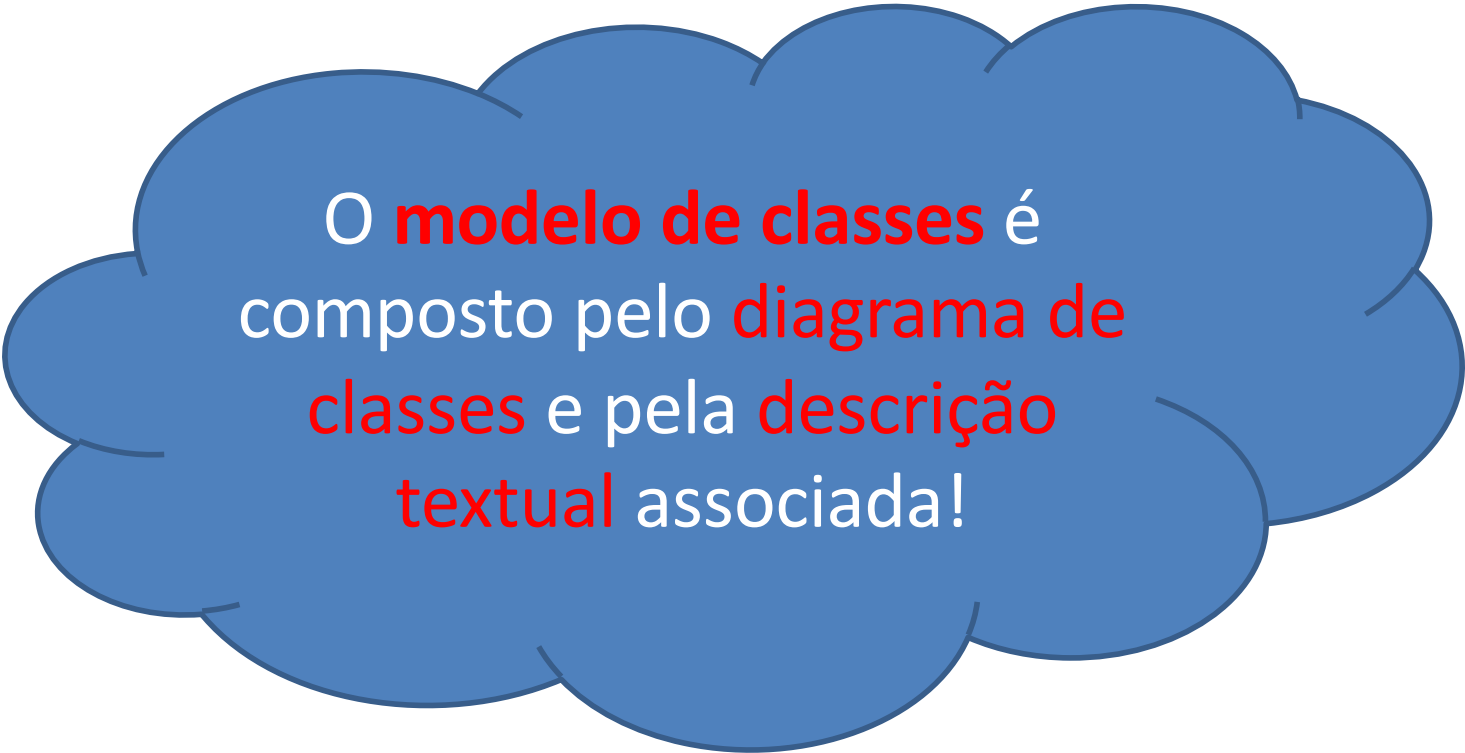
Internamente: os objetos ***colaboram*** uns com os outros para produzir os resultados.

Essa **colaboração** pode ser vista sob o **aspecto dinâmico** e sob o **aspecto estrutural estático**.

ESTÁGIOS DO MODELO DE CLASSES

Modelo de Classes

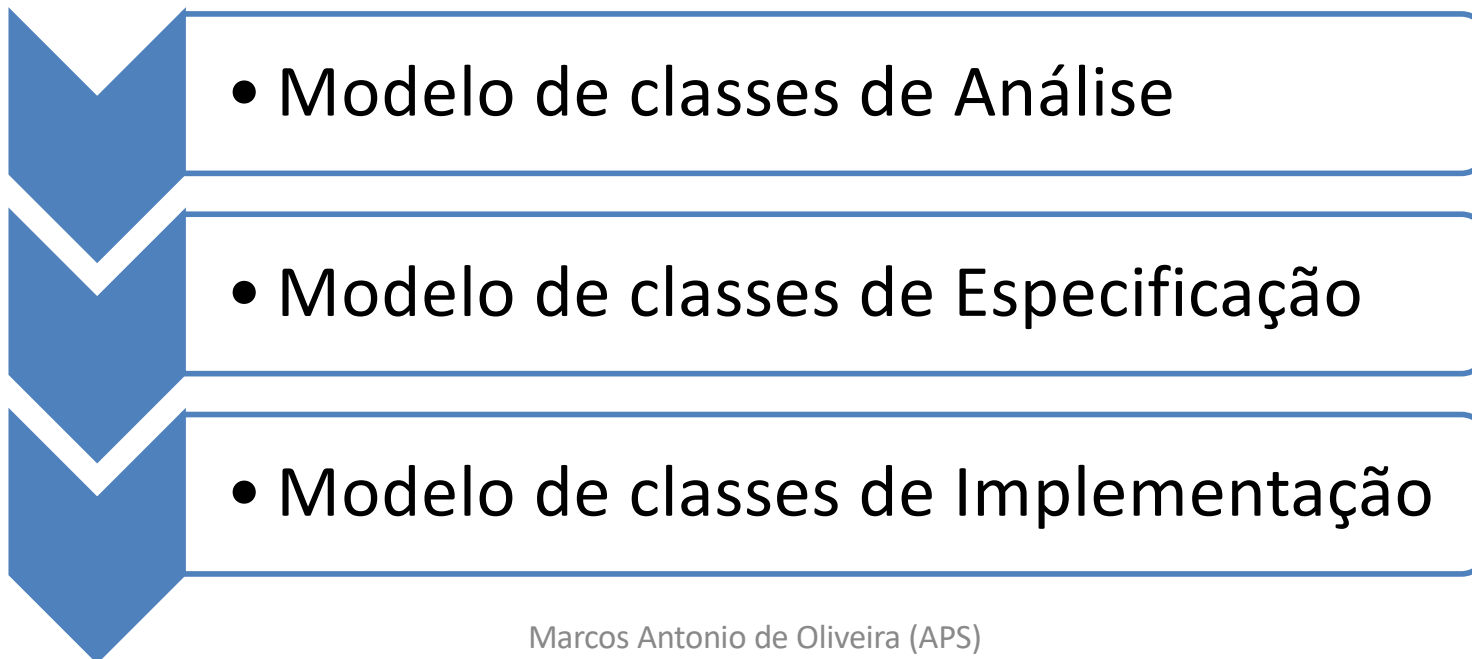
- O diagrama da UML utilizado para representar o aspecto estático é o **diagrama de classes**



O **modelo de classes** é composto pelo **diagrama de classes** e pela **descrição textual** associada!

Modelo de Classes

- O modelo de classes evolui durante o desenvolvimento do sistema
 - À medida que o sistema é desenvolvido, o modelo de classes é incrementado com novos detalhes



Modelo de Classes

O Modelo de Classes de 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

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.

DIAGRAMA DE CLASSES

Introdução

- O diagrama de classes é utilizado na construção do modelo de classes desde o nível de análise até o nível de especificação
- É o digrama mais **rico** da UML em termos de notação

Classes

- Uma classe representa um grupo de objetos semelhantes
- Uma classe descreve esses objetos através de **atributos e operações**
 - Os atributos correspondem às informações que um objeto armazena
 - As operações correspondem às ações que um objeto sabe realizar

Notação para uma Classe

- Representada através de uma “caixa” com no máximo três compartimentos exibidos
- Notação utilizada depende do nível de abstração desejado

Nome da Classe

Nome da Classe
lista de atributos

Nome da Classe
lista de operações

Nome da Classe
lista de atributos
lista de operações

Exemplo (classe ContaBancária)

ContaBancária

ContaBancária
número saldo dataAbertura

ContaBancária
número saldo dataAbertura
criar() bloquear() desbloquear() creditar() debitar()

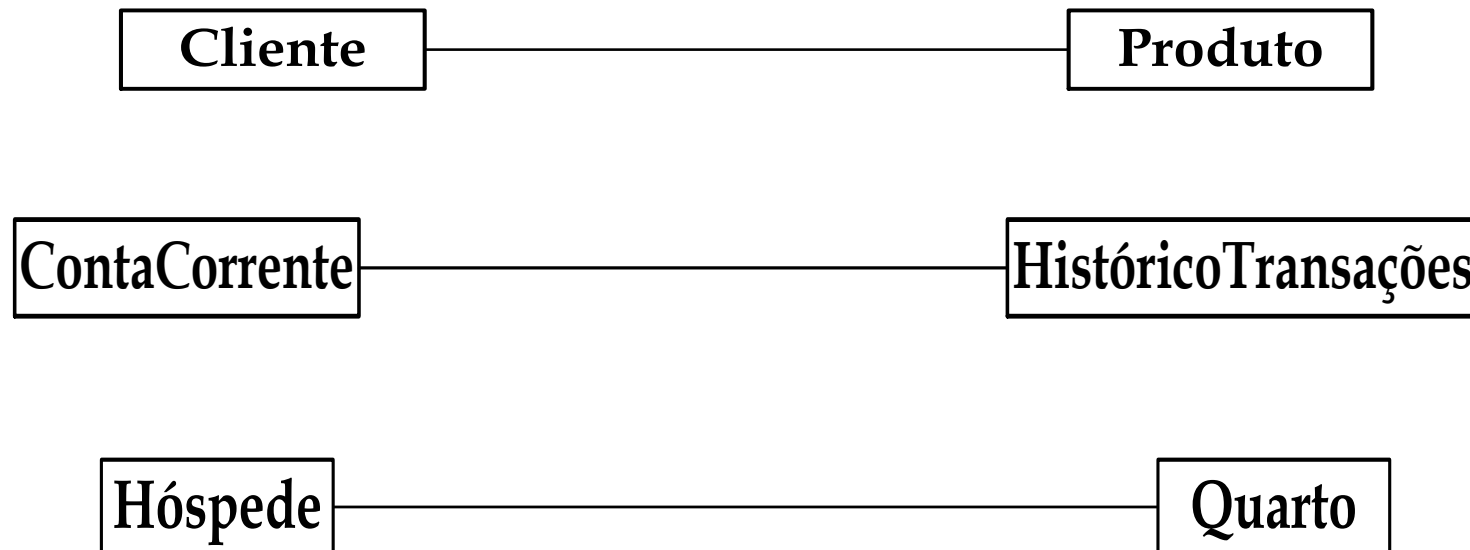
Associações

- Para representar o fato de que objetos podem se relacionar uns com os outros, utiliza-se a ***associação***
- Uma associação representa relacionamentos (ligações) que são formados entre objetos durante a execução do sistema

Embora as associações sejam representadas entre **classes** do diagrama, tais associações representam **ligações possíveis** entre **objetos** das classes envolvidas.

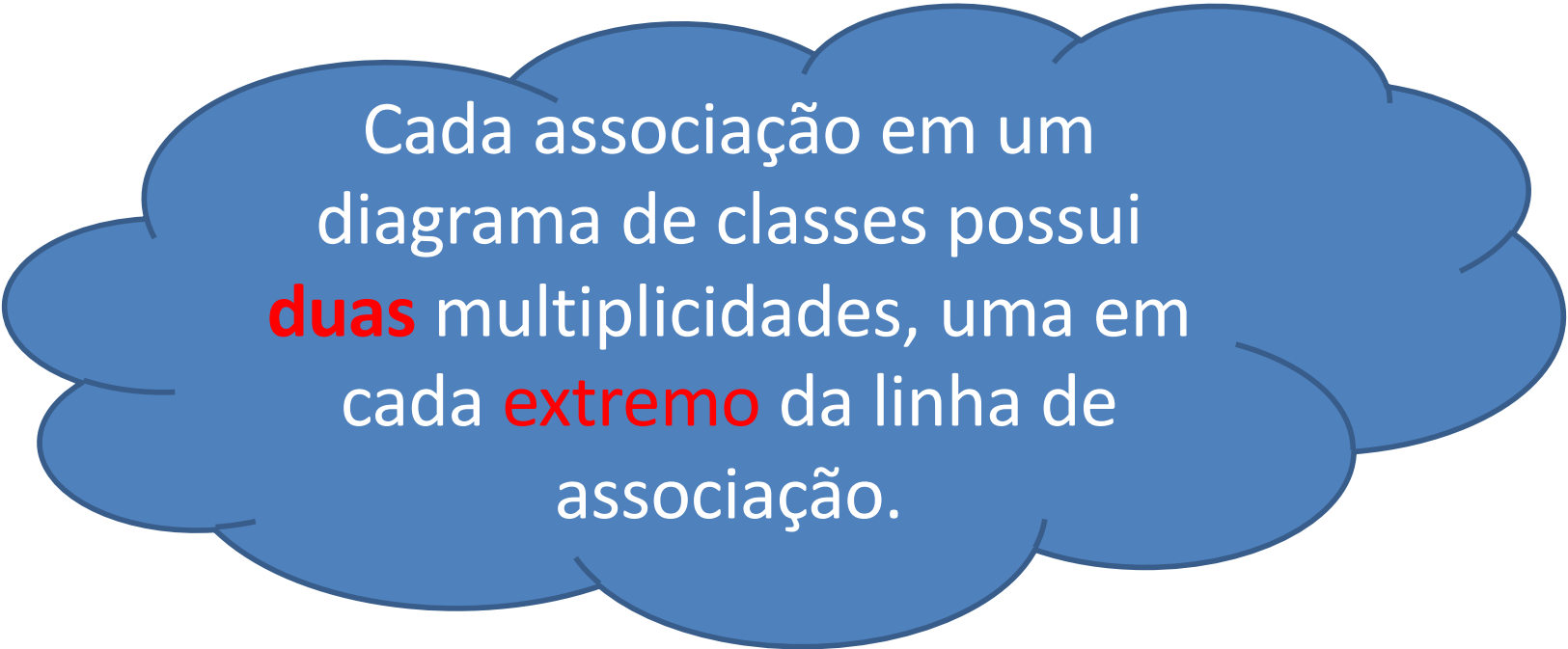
Notação para Associações

- Representada através de um segmento de reta ligando as **classes** cujos **objetos** se relacionam



Multiplicidade

- Representam a informação dos limites **inferior** e **superior** da quantidade de **objetos** aos quais um outro **objeto** pode estar associado



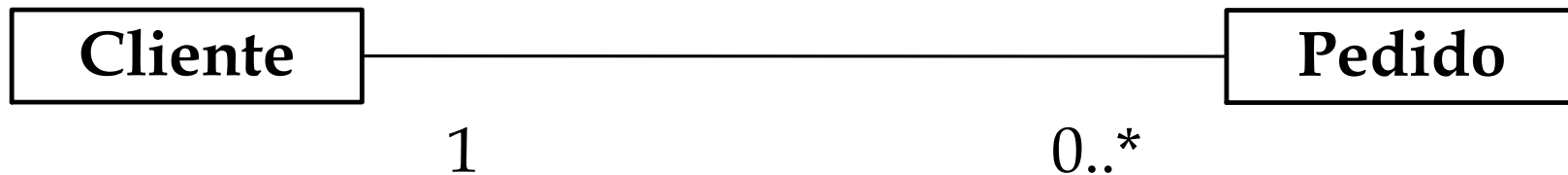
Cada associação em um diagrama de classes possui **duas** multiplicidades, uma em cada **extremo** da linha de associação.

Multiplicidades

Nome	Simbologia
Apenas Um	1..1 (ou 1)
Zero ou Muitos	0..* (ou *)
Um ou Muitos	1..*
Zero ou Um	0..1
Intervalo Específico	$l_i..l_s$

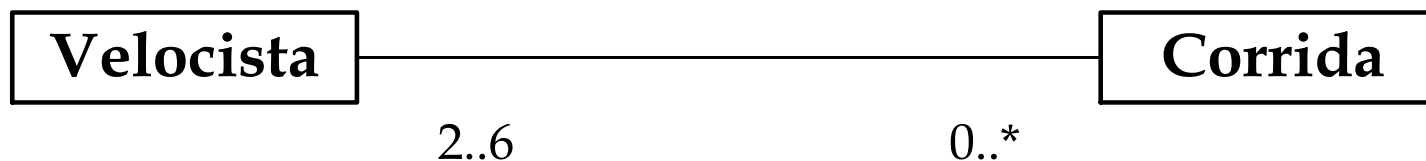
Exemplo (Multiplicidade)

- **Pode** haver um cliente que esteja associado a vários pedidos
- **Pode** haver um cliente que não esteja associado a pedido algum
- Um pedido está associado a um, e somente um, cliente



Exemplo (Multiplicidade)

- Uma corrida está associada a, no mínimo, dois velocistas
- Uma corrida está associada a, no máximo, seis velocistas
- Um velocista ***pode*** estar associado a várias corridas



Conectividade

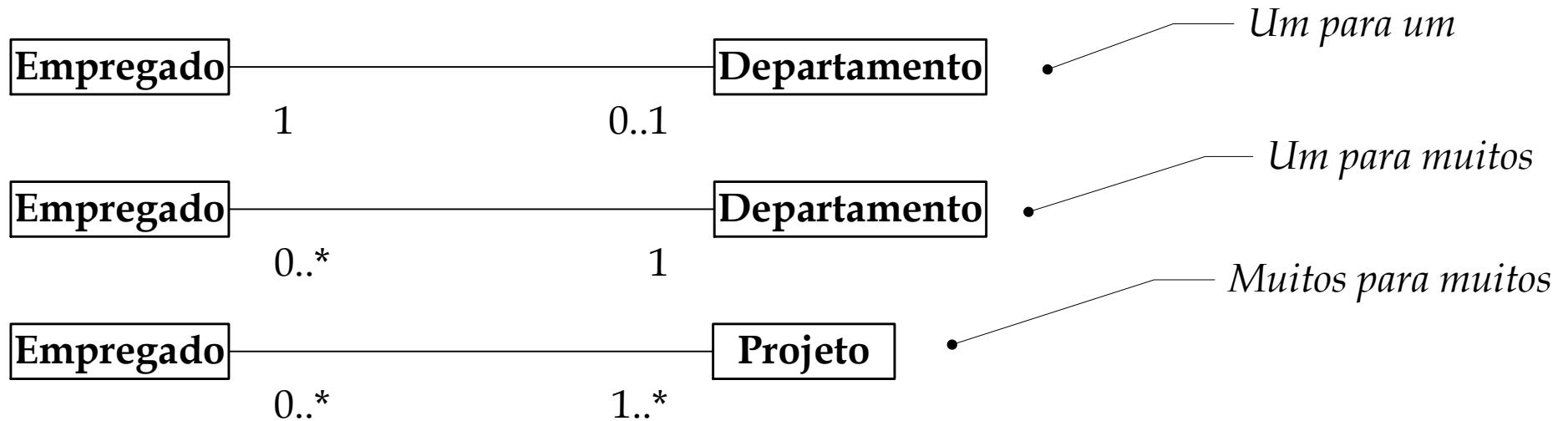
- A **conectividade** corresponde ao tipo de associação entre duas classes
 - “muitos para muitos”
 - “um para muitos”
 - “um para um”

A conectividade da associação entre duas classes depende dos símbolos de multiplicidade que são utilizados na associação!

Conectividade e Multiplicidade

Conectividade	Multiplicidade	
	Em um extremo	Em outro extremo
Um para um	0..1 1	0..1 1
Um para muitos	0..1 1	* 0..* 1..*
Muitos para muitos	* 0..* 1..*	* 0..* 1..*

Exemplo (Conectividade)



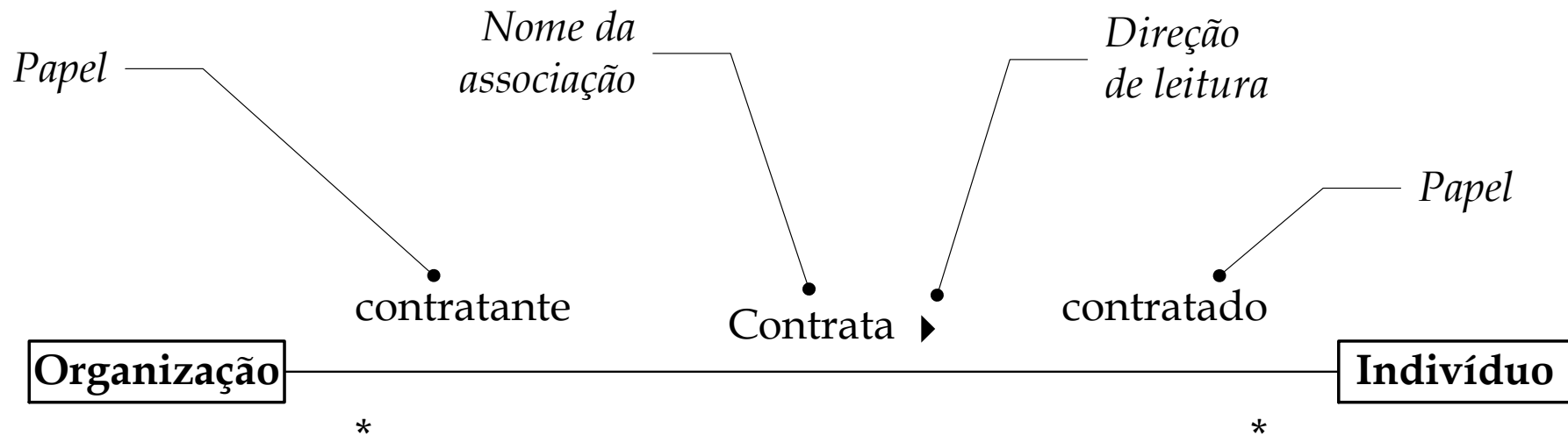
Participação

- Uma característica de uma associação que indica a necessidade (ou não) da existência desta associação entre objetos
- A participação pode ser **obrigatória** ou **opcional**
 - Se o valor mínimo da multiplicidade de uma associação é **igual a 1** (um), significa que a participação é **obrigatória**
 - Caso contrário, a participação é **opcional**

Nome de Associação, Direção de Leitura e Papéis

- Para melhor esclarecer o significado de uma associação no diagrama de classes, a UML define três recursos de notação
 - **Nome da associação:** fornece algum significado semântico a mesma
 - **Direção de leitura:** indica como a associação deve ser lida
 - **Papel:** para representar um papel específico em uma associação

Exemplo (Nome de Associação, Direção de Leitura e Papéis)

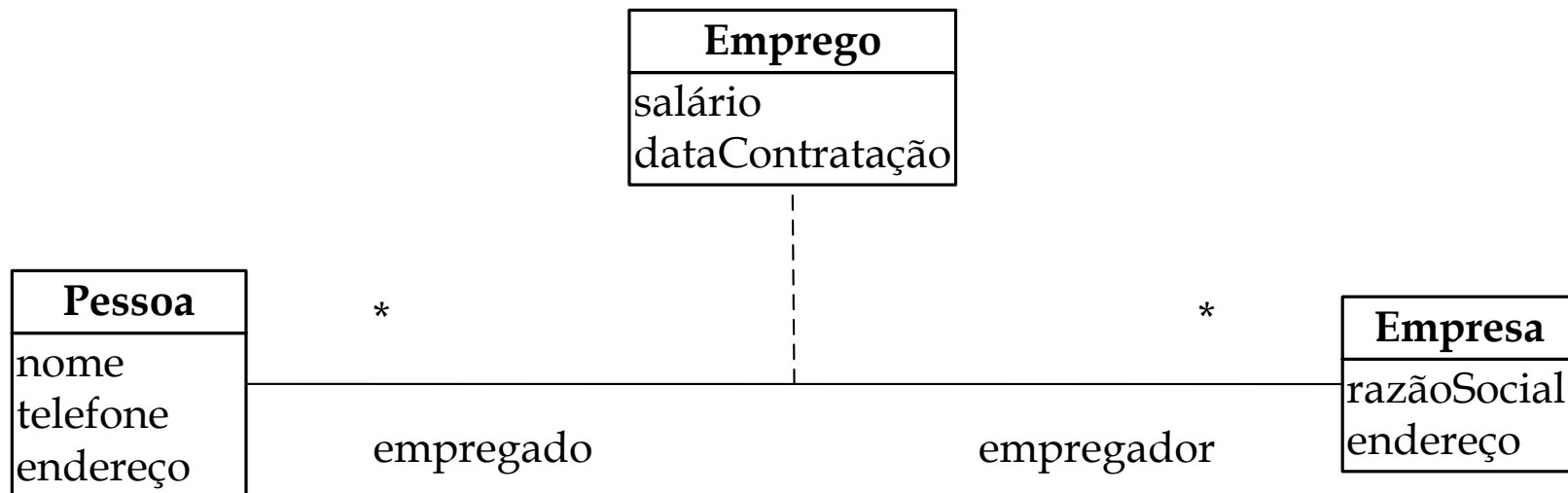


Classe Associativa

- É uma classe que está ligada a uma associação, ao invés de estar ligada a outras classes
- É normalmente necessária quando duas ou mais classes estão associadas, e é necessário manter informações sobre esta associação
- Uma classe associativa pode estar ligada a associações de qualquer tipo de conectividade

Notação para uma Classe Associativa

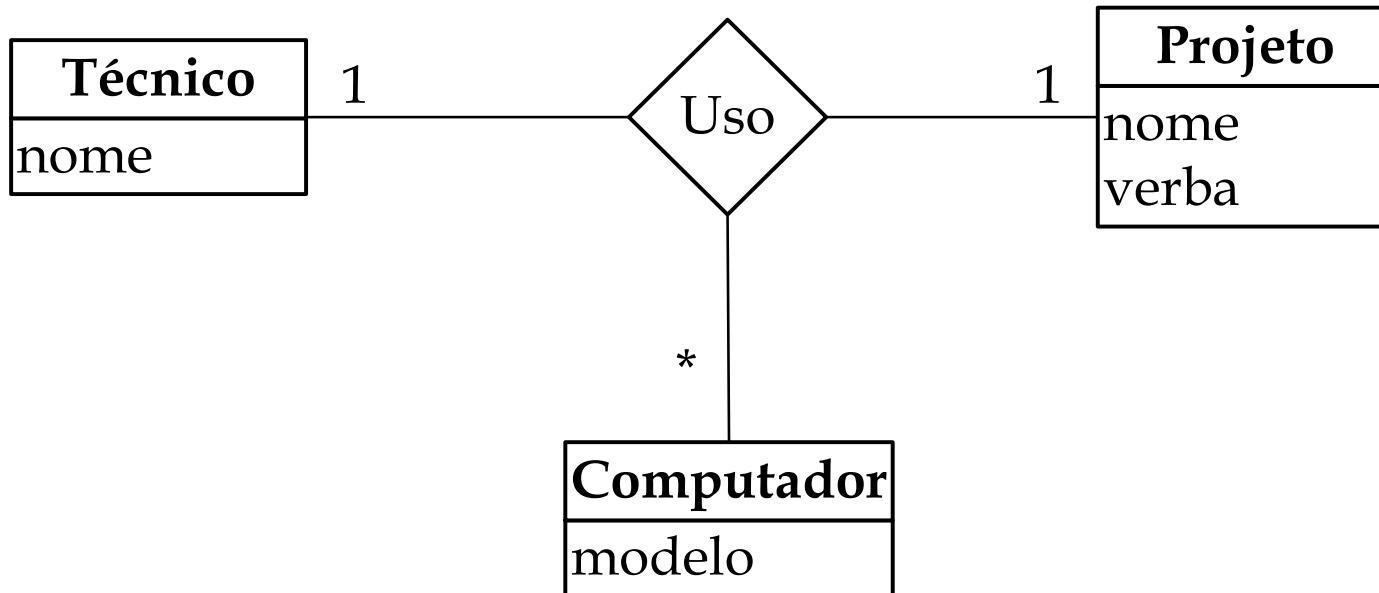
- Representada pela notação utilizada para uma classe. A diferença é que esta classe é ligada a uma associação



Associações N-árias

- São utilizadas para representar a associação existente entre objetos de n classes
- Uma associação ternária é o caso mais comum (menos raro) de associação n -ária ($n = 3$)
- Na notação da UML, as linhas de associação se interceptam em um losango

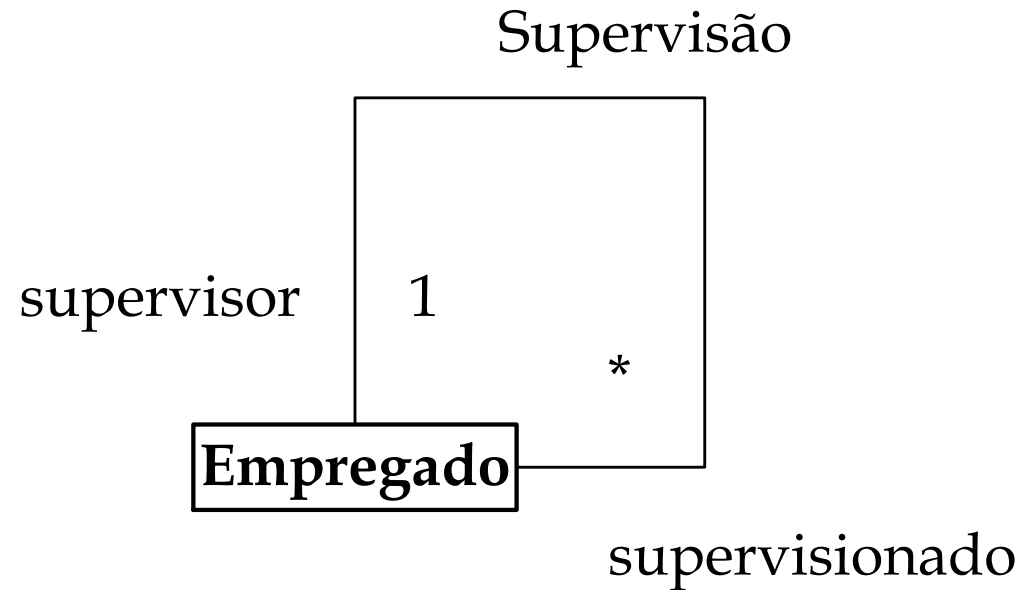
Exemplo (Associação Ternária)



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

Exemplo (Associação Reflexiva)



Agregações e Composições

- É um caso especial da associação...
...conseqüentemente, multiplicidades, participações, papéis, etc. podem ser usados igualmente
- Utilizada para representar conexões que guardam uma relação todo-parte entre si

Onde se puder utilizar uma agregação, uma associação também poderá ser utilizada!

Agregações e Composições

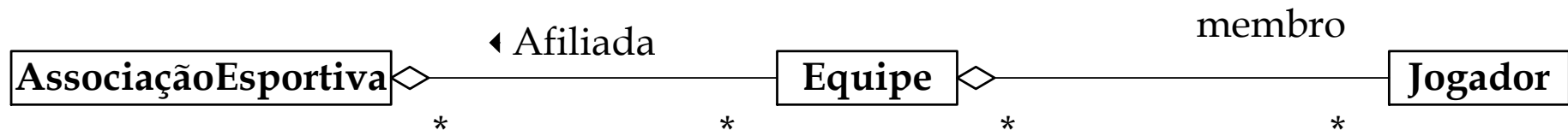
- Características particulares das agregações/composições
 - São assimétricas: se um objeto A é parte de um objeto B, B não pode ser parte de A
 - Propagam comportamento: um comportamento que se aplica a um todo automaticamente se aplica as suas partes
 - As partes são normalmente criadas e destruídas a pelo todo

Agregações e Composições

- Sejam duas classes associadas, X e Y. Se uma das perguntas a seguir for respondida com um sim, provavelmente há uma agregação onde X é todo e Y é parte
 - *X tem um ou mais Y?*
 - *Y é parte de X?*

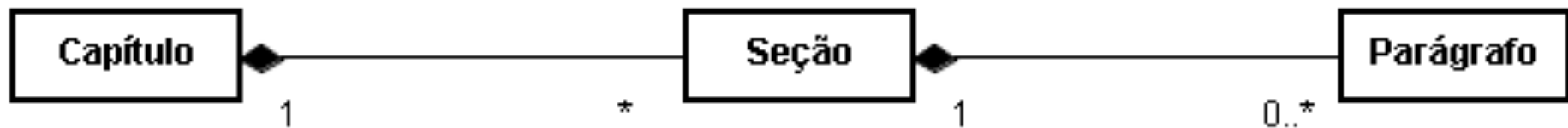
Notação para Agregação

- Uma agregação é representada como uma linha que conecta as classes relacionadas, com um diamante (losango) **branco** perto da classe que representa o todo



Notação para Composição

- Uma composição é representada como uma linha que conecta as classes relacionadas, com um diamante (losango) **negro** perto da classe que representa o todo



Agregações vs Composições

- As diferenças entre agregações e composições não são bem claras, mas essas são as mais visíveis
 - Na agregação, a destruição de um objeto **todo** não implica necessariamente a destruição do objeto **parte**
 - Na composição, os objetos **parte** pertencem a único objeto **todo**, por isso quando um objeto **todo** é destruído os objetos **parte** também são

Generalizações e Especializações

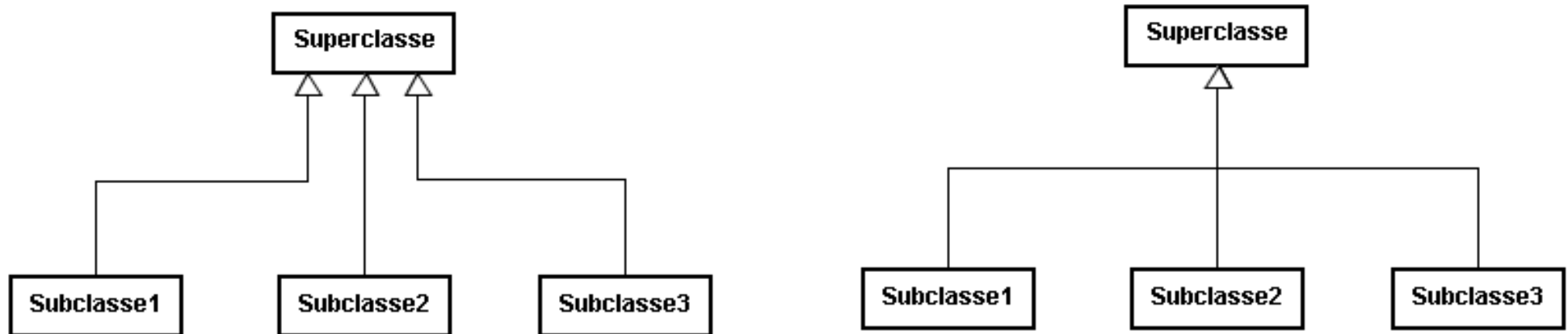
- O modelador também pode representar relacionamentos entre classes
 - Esses denotam relações de **generalidade** ou **especificidade** entre as classes envolvidas
 - Exemplo
 - O conceito *mamífero* é mais genérico que o conceito *ser humano*
 - O conceito *carro* é mais específico que o conceito *veículo*
- Esse é o chamado **relacionamento de herança**
 - Relacionamento de generalização/especialização
 - Relacionamento de gen/espec

Generalizações e Especializações

- Sejam A e B classes que se relacionam através da herança
 - Se A é uma **generalização** de B, então, B é uma **especialização** de A
- Os termos **subclasse** e **superclasse** são utilizados para denotar relações de herança
 - Subclasse é a classe que herda
 - Superclasse é a classe herdada

Generalizações e Especializações

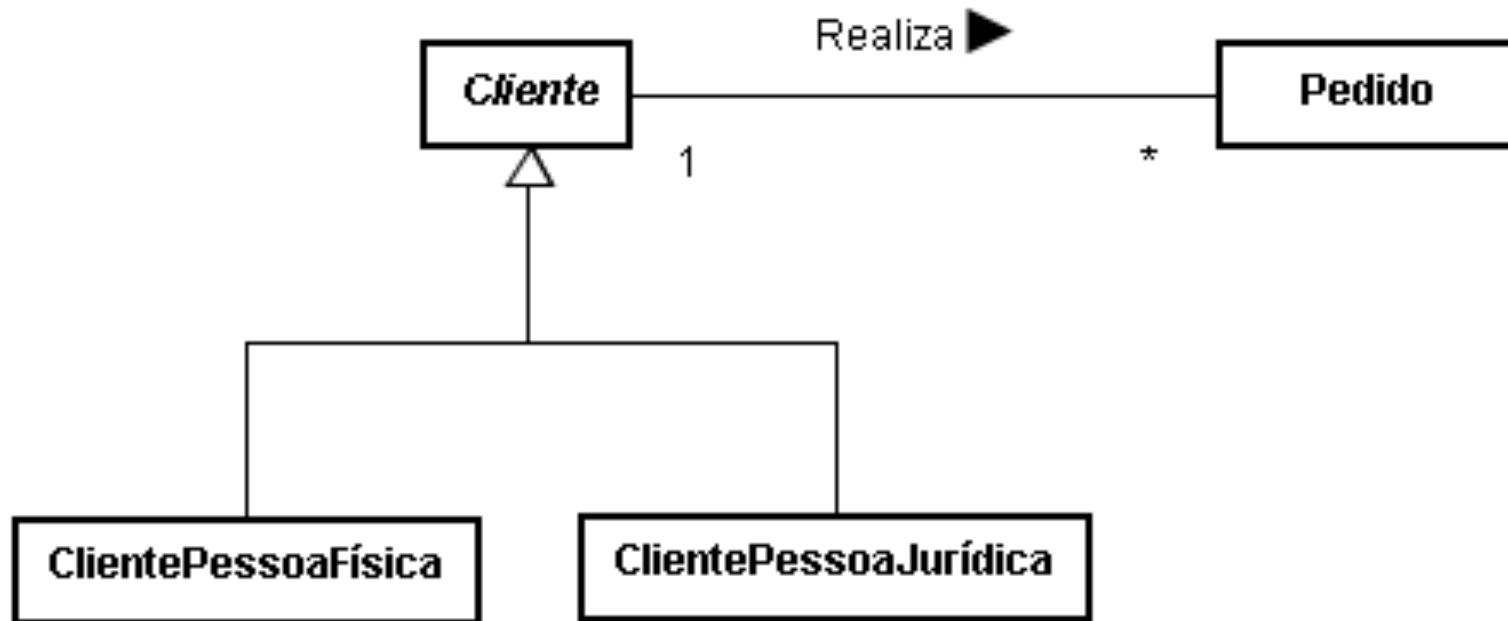
- No diagrama de classes, a herança é representada na UML por uma flecha partindo da **subclasse** em direção a **superclasse**



Herança vs Associação

- A diferença semântica entre a herança e a associação
 - A primeira trata de um relacionamento entre **classes**, enquanto que a segunda representa relacionamentos entre **instâncias** de classes (**objetos**)
 - Na associação, **objetos específicos** de uma classe se associam entre si ou com objetos específicos de outras classes
 - Exemplo
 - Herança: "Gerentes são *tipos especiais* de funcionários"
 - Associação: "Gerentes *chefiam* departamentos"

Exemplo (Herança)

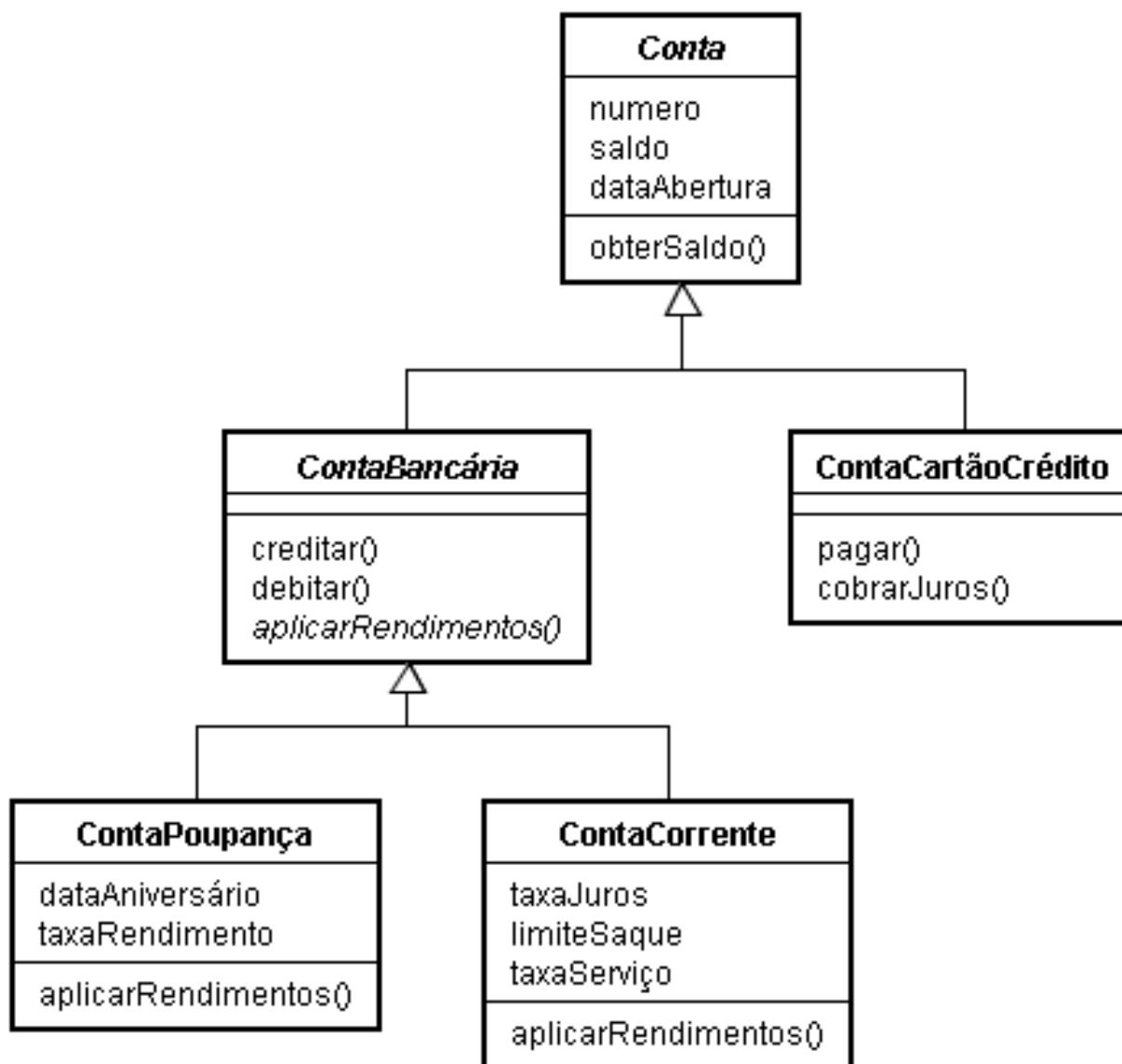


IMPORTANTE: Além das propriedades e operações, as subclasses herdam os relacionamentos!

Herança

- As principais propriedades do relacionamento de herança são
 - *Transitividade*
 - Uma classe herda tanto os **atributos**, **operações** e **relacionamentos** da super classe imediata quanto das superclasses **não imediatas**
 - *Assimetria*
 - Dadas duas classes A e B, se B é subclasse de A, então A não pode ser subclasse de B

Exemplo (Herança)



Classes Abstratas

- Usualmente, a existência de uma classe se justifica pelo fato de haver a possibilidade de gerar instâncias da mesma
 - Essas são as *classes concretas*
- No entanto, podem existir classes que não geram instâncias diretas
 - Essas são as *classes abstratas*

Classes Abstratas

- Classes abstratas são utilizadas para organizar e simplificar uma hierarquia de generalização
 - Propriedades comuns a diversas classes podem ser organizadas e definidas em uma classe abstrata a partir da qual as primeiras herdam
- Subclasses de uma classe abstrata também podem ser abstratas, mas a hierarquia deve terminar em uma ou mais classes concretas

Notação para Classe Abstrata

- Na UML, uma classe abstrata é representada com o seu nome em **itálico**

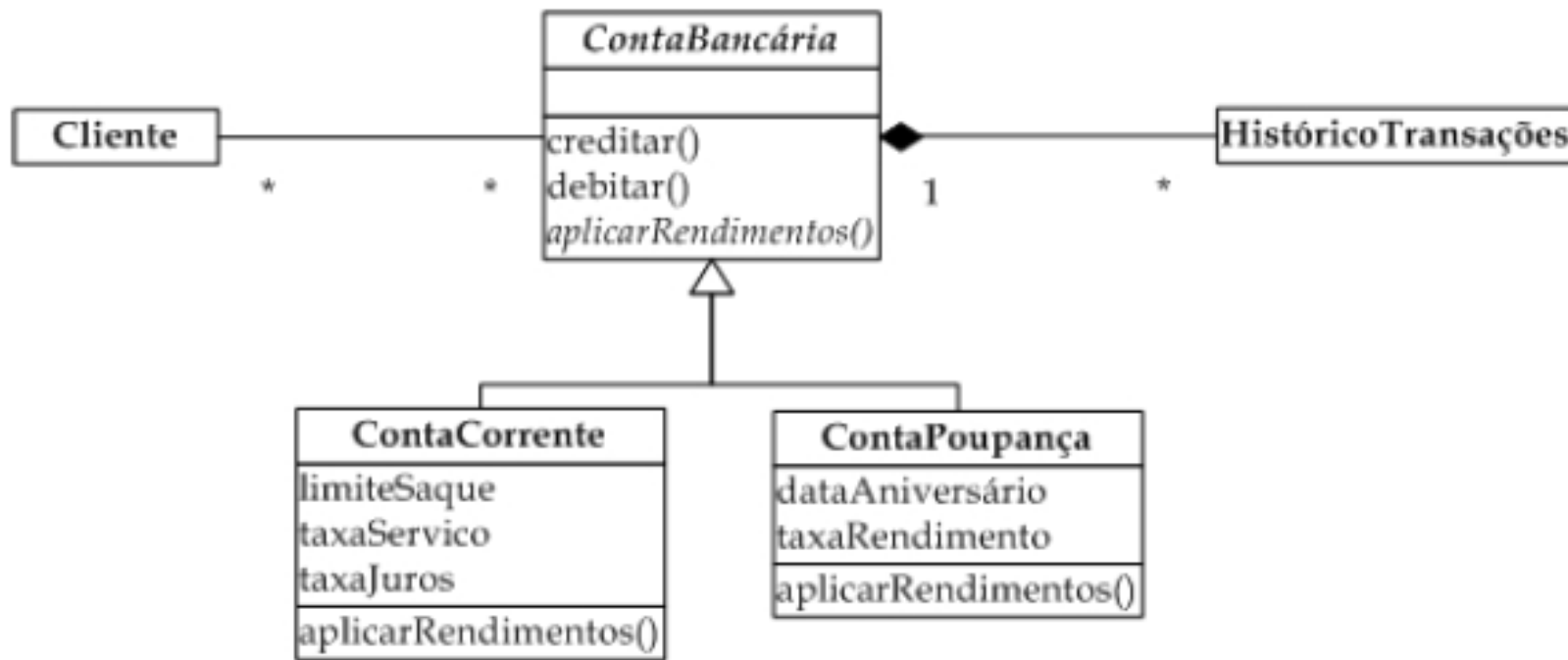


DIAGRAMA DE OBJETOS

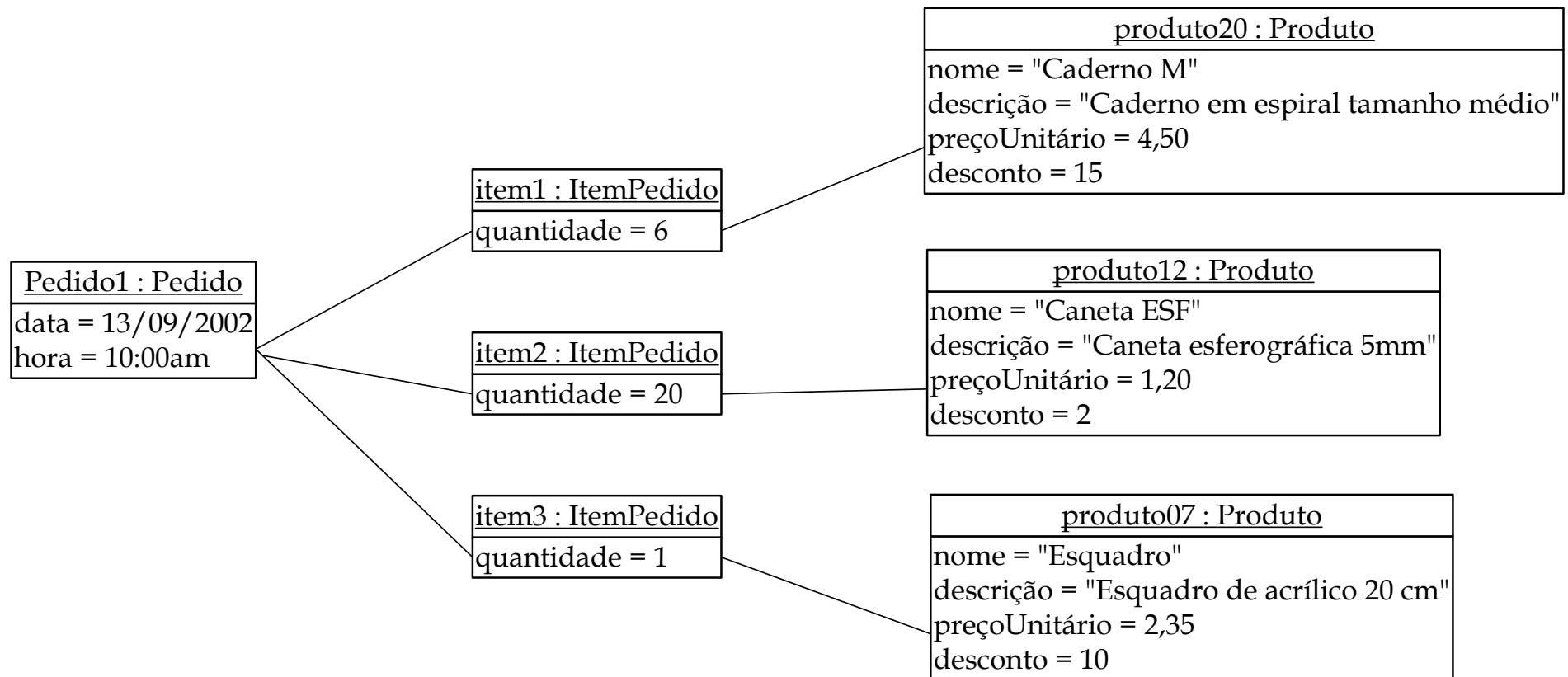
Diagrama de Objetos

- 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

Diagrama de Objetos

Formato	Exemplo
<u>nomeClasse</u>	<u>Pedido</u>
<u>nomeObjeto: NomeClasse</u>	<u>umPedido: Pedido</u>

Exemplo (Diagrama de Objetos)



Exemplo (Diagrama de Objetos)

