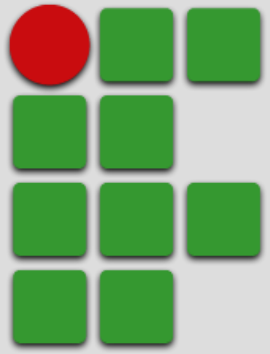


INSTITUTO FEDERAL
Paraná

DIAGRAMA DE CLASSES

PROFª MARCELA TURIM KOSCHEVIC

OBJETIVOS DA AULA

- Revisar conceitos básicos da Orientação a Objetos.
- Conhecer a fundamentação teórica para o diagrama de classes.
- Aprender sobre os relacionamentos (associações) entre as classes.
- Iniciar um trabalho prático sobre diagrama de classes.

Obs.: Para essa aula, retirei recortes dos livros de LIMA (2014) e PAGE-JONES (2001)

CLASSES

Uma **classe** é a definição dos atributos e das ações de um tipo de objeto; ela descreve um conjunto de objetos individuais em qualquer contexto. É obtida pela classificação de objetos com a mesma estrutura de dados e o mesmo comportamento.

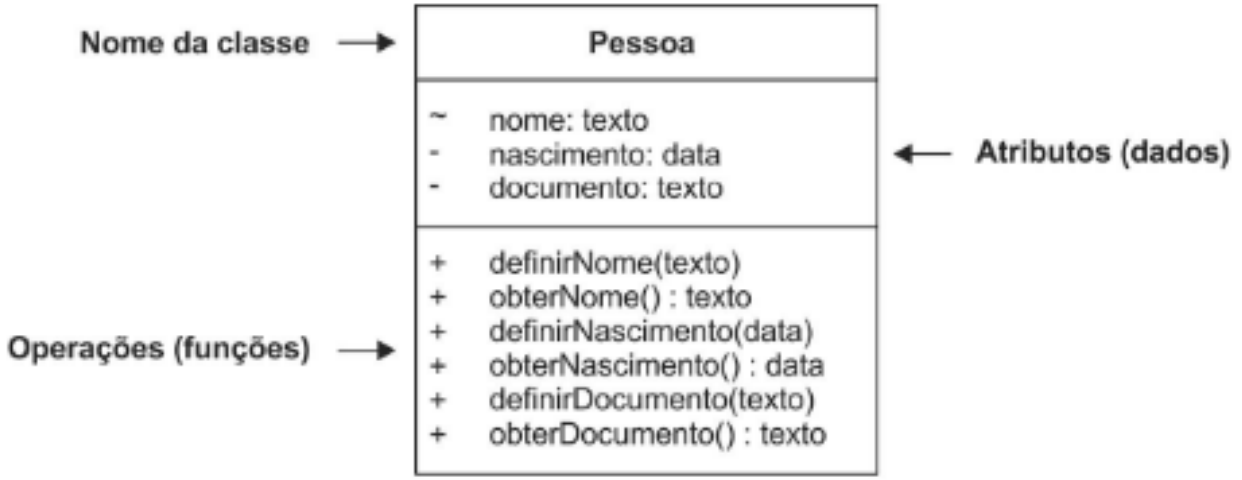


Figura 1.3. Representação de classe no modelo.

ATRIBUTOS

Os atributos, como vimos, são recursos de uma classe ou qualquer outro elemento que represente propriedades ou elementos de dados. Em algumas linguagens, os atributos são denominados **variável de instância de classe** ou **membro de dado**.

Os atributos possuem algumas características importantes, como visibilidade (escopo), nome, tipo de dado e valor inicial.

- **pública:** o atributo é acessível (pode "ser enxergado") por outras classes; a visibilidade pública é representada no modelo por um sinal de adição (+).
- **privada:** o atributo é acessível somente pela própria classe; é representada no modelo por um sinal de subtração (–).
- **protegida:** o atributo é acessível somente pela própria classe e suas subclasses; é representada no modelo pelo símbolo de sustenido (#).
- **de pacote:** o atributo de uma classe é acessível somente pelas classes do pacote que a contém; é representada no modelo pelo til (~) e disponível apenas em algumas linguagens, como Java e C# (onde o pacote é denominado namespace).

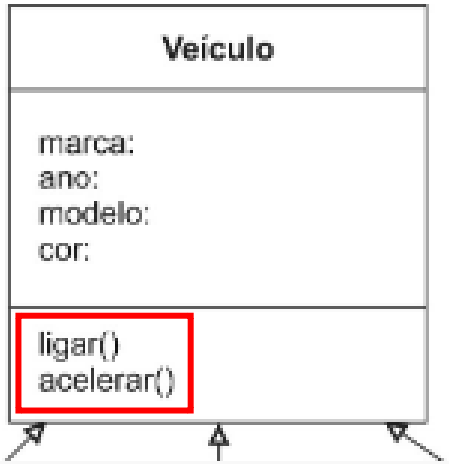
OPERAÇÕES

No mundo real, uma operação é apenas a abstração de um comportamento (ações) semelhante entre diferentes objetos.

Depois de modelados os objetos, as operações são recursos de uma classe que representam comportamentos ou serviços que ela suporta. Os métodos implementam as operações de uma determinada classe.

Os objetos interagem e comunicam-se por meio de mensagens, que identificam os métodos a serem executados no objeto receptor.

Para invocar um método de um objeto, envia-se uma mensagem para ele especificando o nome do objeto, o método a ser executado e a lista de argumentos requeridos. Após a execução, o objeto pode ou não retornar um valor como resposta à mensagem recebida.



HERANÇA

- Herança é o mecanismo em que uma classe herda atributos e comportamento de sua classe imediatamente superior e também pode possuir atributos e operações (ações) específicos dela mesma.

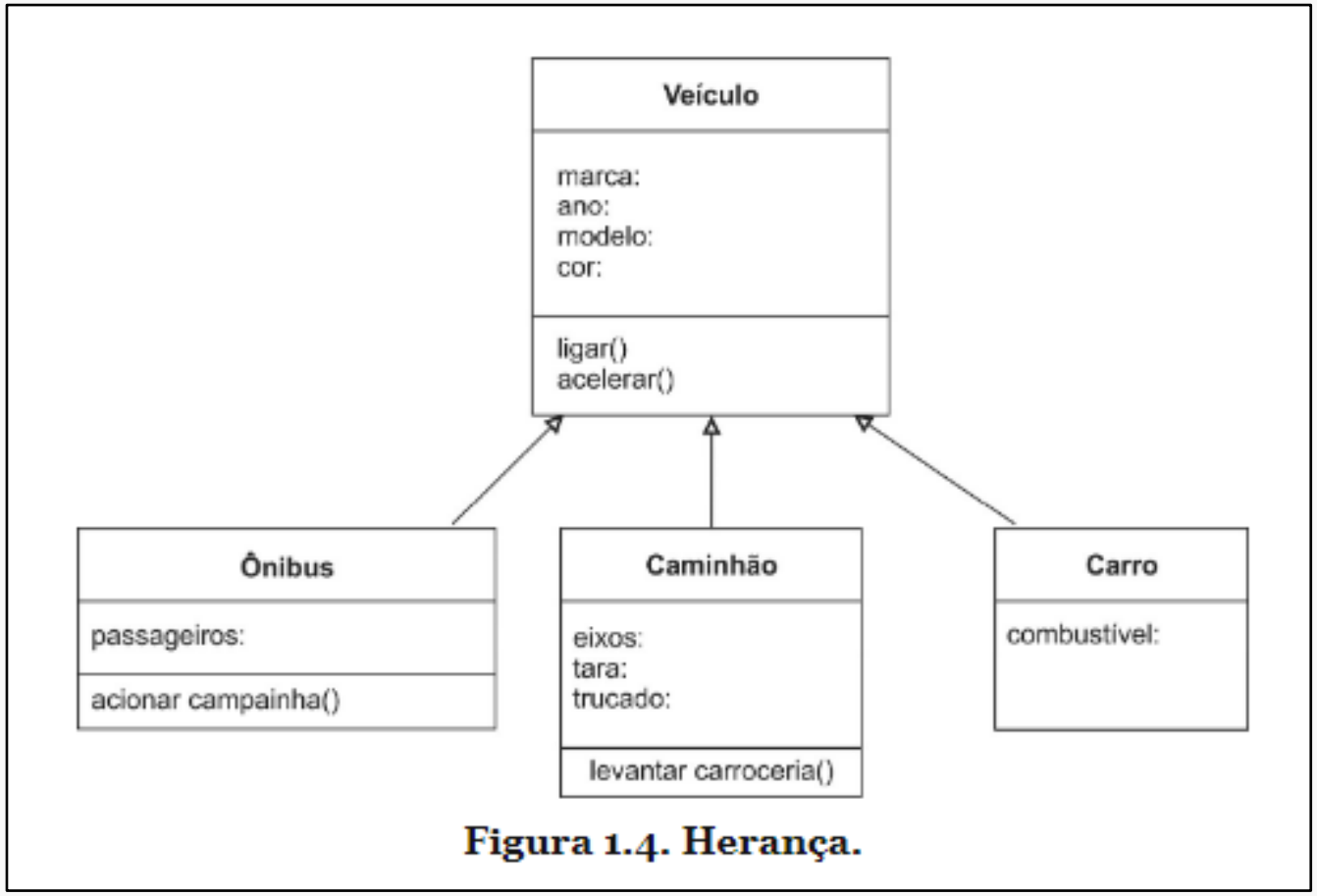


Figura 1.4. Herança.

HERANÇA (OUTRO EXEMPLO)

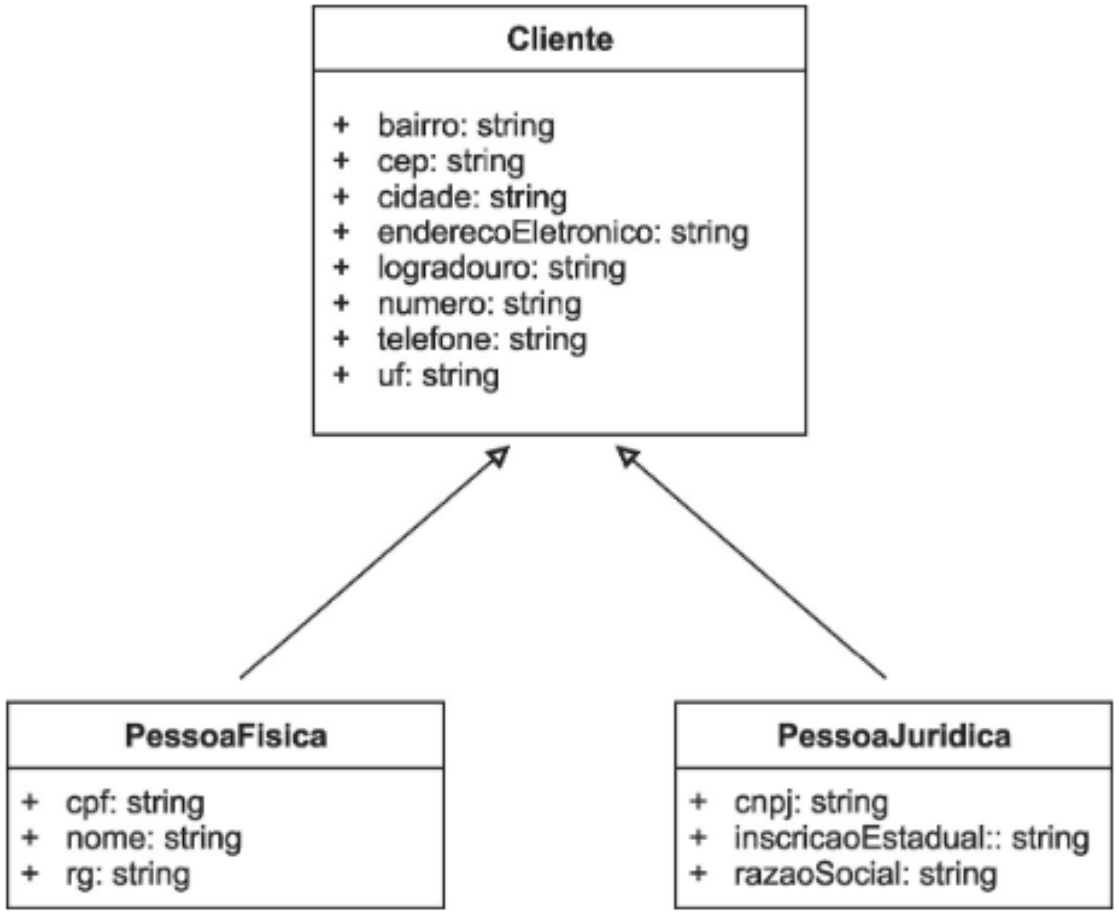
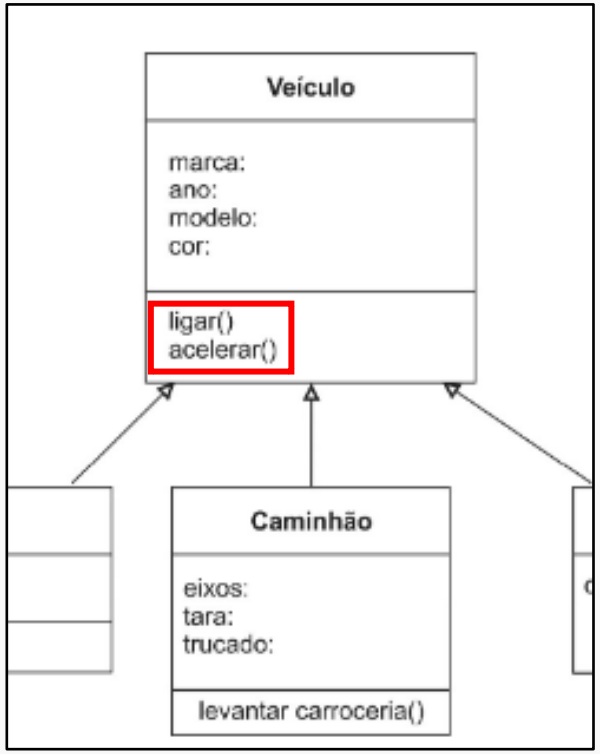


Figura 9.16. Generalização/Especialização da classe Cliente.

POLIMORFISMO

Polimorfismo é o princípio em que classes derivadas de uma mesma superclasse podem invocar operações que têm a mesma assinatura, mas comportamentos diferentes em cada subclasse, produzindo resultados diferentes, dependendo de como cada objeto implementa a operação. Em outras palavras, é a capacidade de objetos diferentes possuírem operações com o mesmo nome e a mesma lista de argumentos, mas que executam tarefas de formas diferentes.

A herança permite pôr em prática o polimorfismo, desde que partes da superclasse não sejam eliminadas nas subclasses.



ENCAPSULAMENTO

O **encapsulamento**, ou **ocultação de informações**, é uma técnica que consiste em separar aspectos externos dos internos da implementação de um objeto, isto é, determinados detalhes ficam ocultos aos demais objetos e dizem respeito apenas ao próprio objeto.

No exemplo do veículo, como funciona o processo de aceleração "não é de nossa conta". Não interessa ao motorista saber como é o funcionamento interno do motor; isso é de responsabilidade do projetista do veículo. O que conhecemos do objeto é apenas o que se apresenta externamente, como ignição, câmbio, acelerador, embreagem etc. que solicitam serviços internos e acionam operações internas do veículo.

A grande vantagem do encapsulamento é o isolamento dos dados. Usando o exemplo da classe Pessoa, Figura 1.3, qualquer aplicação pode usar a classe para instanciar objetos, mas jamais terá acesso à forma como é definido o nome da pessoa - o cliente (ou usuário da classe) limitar-se-á apenas a solicitar o serviço invocando a operação definirNome, passando o parâmetro (argumento) esperado: o nome.

Os atributos estão encapsulados, isto é, envolvidos por código de forma que só são visíveis na operação em que foram criados. Isso também vale para as operações, quando algumas delas são visíveis apenas pelo próprio objeto.

Vale lembrar que o encapsulamento estabelece uma dependência com a relação de herança, pois objetos da subclasse herdam todas as definições de atributos e operações da superclasse.

CONCEITOS IMPORTANTES

- Objetos são coisas do mundo real que descobrimos estudando suas características (atributos) e seu comportamento (ações).
- Abstração é o processo de separar mentalmente os objetos observados e estudados da realidade.
- Domínio da aplicação é o modelo obtido dentro de um contexto de negócio após o estudo e a observação da realidade.
- Classe é a definição dos atributos e do comportamento de um conjunto de objetos individuais em qualquer contexto.
- Cada objeto pertencente a uma classe é denominado instância de classe.
- Atributos são a representação das propriedades ou elementos de dados de um objeto. Em algumas linguagens, são denominados variáveis de instância de classe ou membros de dado. Possuem algumas características importantes, como visibilidade (escopo), nome, tipo de dado e valor inicial.
- Operações são recursos de uma classe que representam um comportamento ou serviço que ela suporta.
- Os objetos interagem e comunicam-se por meio de mensagens, que identificam os métodos a serem executados no objeto receptor.

CONCEITOS IMPORTANTES

- Uma operação possui visibilidade, nome, lista de argumentos e tipo de retorno. O conjunto dessas características é denominado assinatura de operação.
- A visibilidade pode ser pública (acessível por outras classes), privada (acessível somente pela própria classe), protegida (acessível somente pela própria classe e suas subclasses) e de pacote (acessível somente pelas classes do pacote que as contém).
- Herança é o mecanismo em que uma classe herda atributos e comportamento de sua classe imediatamente superior e também pode possuir atributos e operações (ações) específicos dela mesma.

- Polimorfismo é a capacidade de objetos diferentes possuírem operações com o mesmo nome e a mesma lista de argumentos, mas que executam tarefas de formas diferentes.
- Encapsulamento, ou ocultação de informações, é uma técnica que consiste em separar aspectos externos dos internos da implementação de um objeto - os detalhes ficam ocultos aos demais objetos e dizem respeito apenas ao próprio objeto.
- A orientação a objetos preocupa-se com o que um objeto é e não como ele deve ser utilizado. Antes de desenvolver orientado a objetos, é preciso pensar orientado a objetos.

DIAGRAMA DE CLASSES DA UML

Um diagrama de classe mostra a estrutura estática do modelo, em que os elementos são representados por classes, com sua estrutura interna e seus relacionamentos. Os diagramas de classe também podem ser organizados em pacotes, mostrando somente o que é relevante em um pacote específico, considerando, por exemplo, o contexto que um grupo de classes tem em comum (estoque, venda etc.) ou os atores que utilizam seus serviços (vendedor, almoxarife etc.)

Os pacotes geralmente são um resumo do modelo de desenho, mostram as classes que participam da realização dos casos de uso de forma coerente e representam a funcionalidade do sistema do ponto de vista lógico da arquitetura.

O propósito básico é obter as estruturas mais importantes - as classes com seus relacionamentos de associação, agregação e generalização. Esse tipo de diagrama também é útil na revisão do que é armazenado no sistema e nas estruturas de armazenamento. Também é possível especificar os pacotes lógicos do sistema e suas dependências, ilustrando as camadas da futura aplicação.

Uma classe deve ser apresentada pelo menos uma vez em um diagrama. Isso significa que, dependendo do porte do sistema, para melhor entender o modelo, uma classe pode aparecer diversas vezes e em vários diagramas de uma mesma visão.

Este é o passo inicial para, segundo o RUP, "transformar o sistema de uma mera declaração de comportamento requerido em uma descrição de como o sistema irá trabalhar".

CLASSES DE ANÁLISE

Segundo a documentação da UML pelo OMG, "uma classe descreve um conjunto de objetos que compartilham as mesmas especificações de atributos, operações, restrições e semântica. A finalidade de uma classe é classificar objetos e especificar os recursos que caracterizam a estrutura e o comportamento desses objetos".

As **classes de análise** representam um modelo conceitual primário para elementos que têm responsabilidades e comportamento no sistema. Elas são usadas para capturar esses elementos que representem as classes de protótipo e são o primeiro passo para maiores abstrações que o sistema deve manipular. Classes de análise, geralmente, são mantidas em um alto nível, numa visão puramente conceitual. Como veremos adiante, elas também causam as maiores abstrações no desenho do modelo.

O objetivo da análise é fazer um mapeamento preliminar do comportamento requerido para os elementos de modelagem no sistema a serem implementados posteriormente nas fases de construção e implantação. Realizamos um refinamento nos detalhes e na precisão do desenho a fim de conseguir classes de análise que possam evoluir antes de serem solidificadas nas atividades de especificação e implementação.

Um modelo de análise separado é importante, principalmente quando o sistema deve ser desenhado para múltiplos ambientes, com arquiteturas separadas. O modelo de análise é uma abstração, ou uma generalização, do modelo de desenho. Ele omite muitos detalhes do desenho e da implementação a fim de fornecer uma visão geral da funcionalidade do sistema. Em algumas empresas, onde sistemas vivem por décadas, ou onde eles variam muito, um modelo de análise tem provado ser bastante útil.

O modelo de classes é importante para implementadores nas atividades de especificação, desenhistas de outras partes do sistema para entender como sua funcionalidade pode ser usada, desenhistas de caso de uso para instanciá-los na realização de casos de uso, testadores para testar as classes e planejar as atividades de teste etc.

No diagrama, uma classe é representada seguindo alguns padrões, quais sejam:

- O nome da classe deve estar em negrito e centralizado no primeiro compartimento.
- A primeira letra do nome da classe deve ser maiúscula.
- Os nomes de atributos e operações devem ser escritos em letras minúsculas e, quando tratar-se de nomes compostos, utiliza-se a notação húngara, onde as iniciais dos termos, fora o primeiro, são maiúsculas.
- Os nomes de classes abstratas devem ser escritos em itálico.
- Podem ser exibidos completamente atributos e operações, mas se for necessário, podem ser suprimidos em outros contextos, mostrando apenas o nome da classe.

EXEMPLO DE DIAGRAMA DE CLASSES

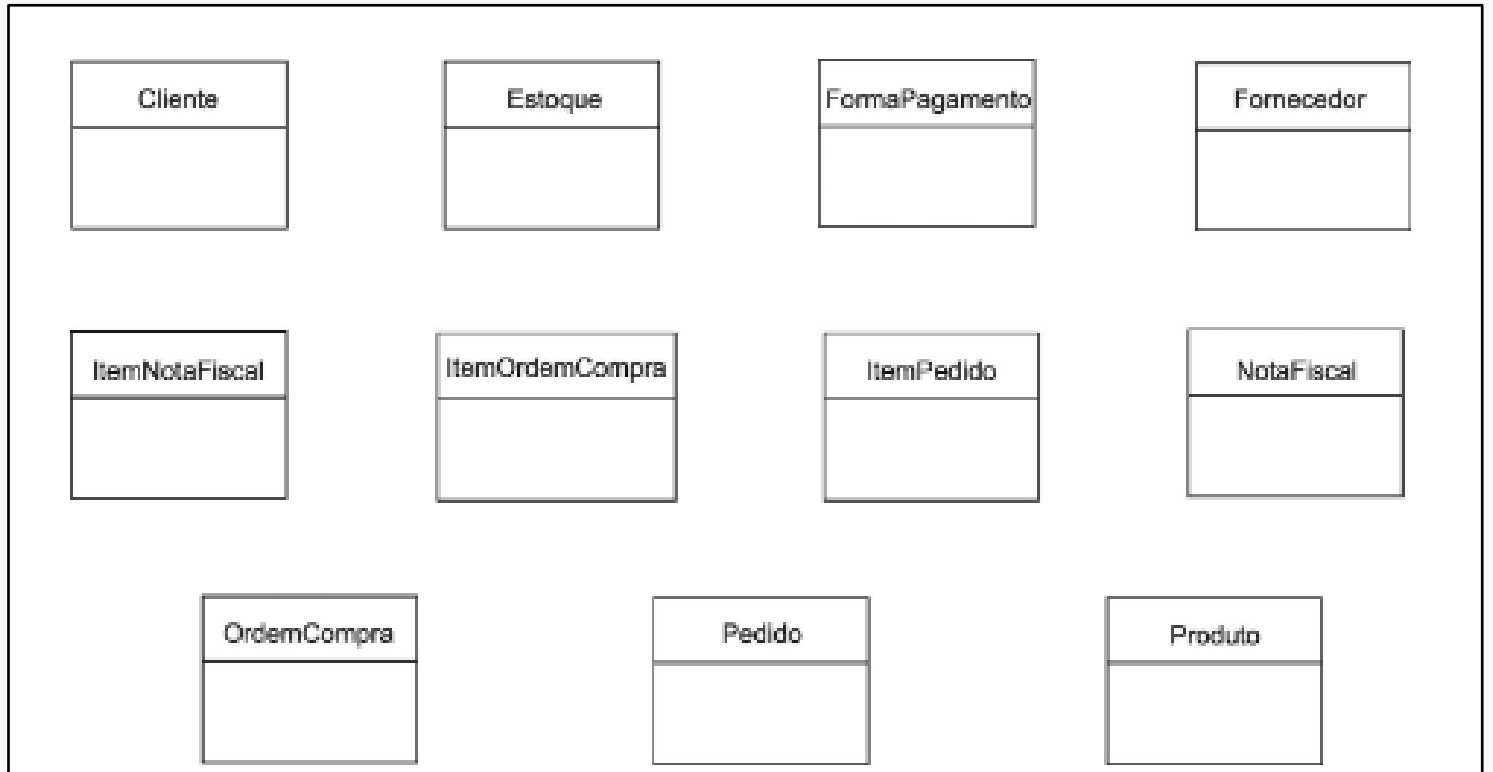


Figura 9.3. Primeiras classes do modelo do sistema de vendas.

EXEMPLO DE DIAGRAMA DE CLASSES

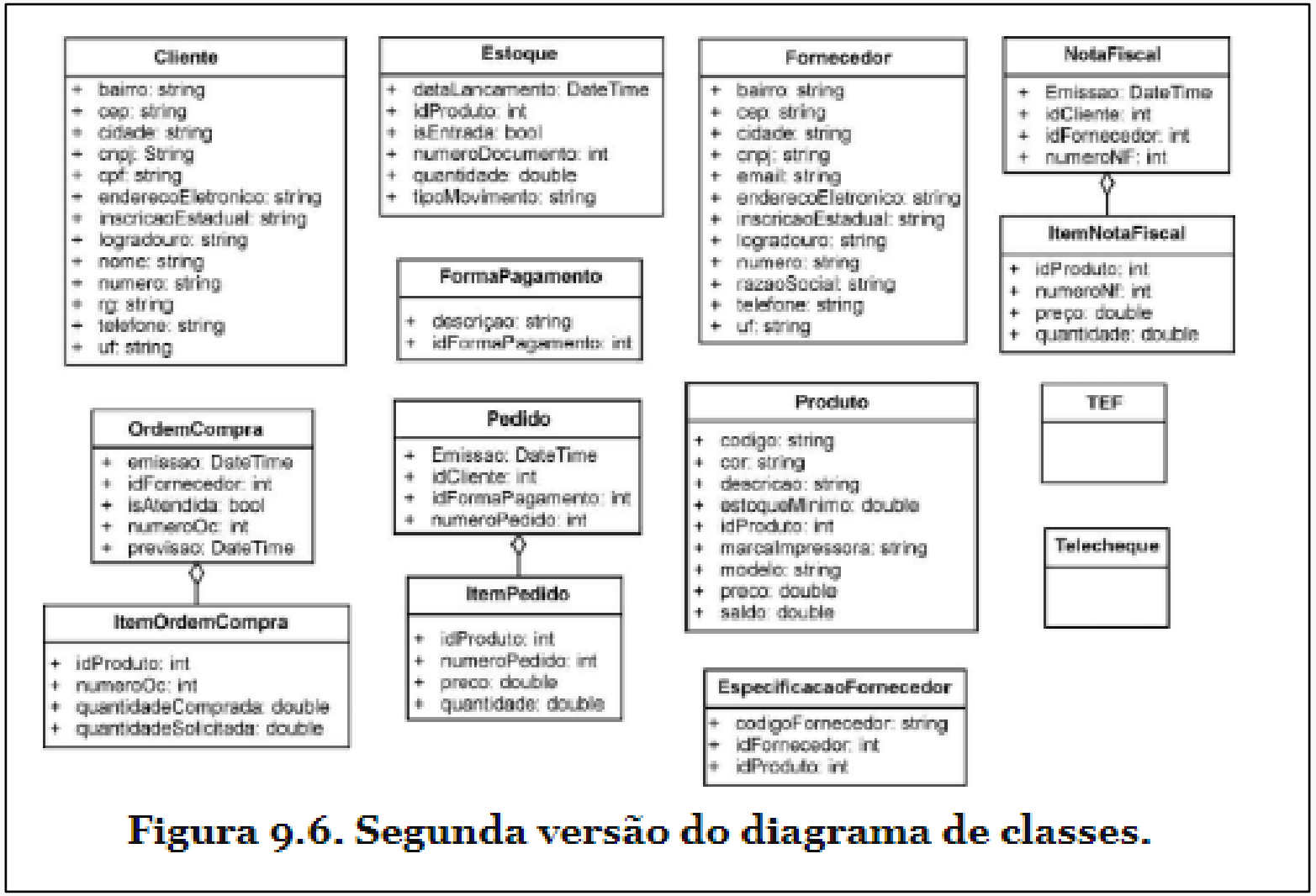
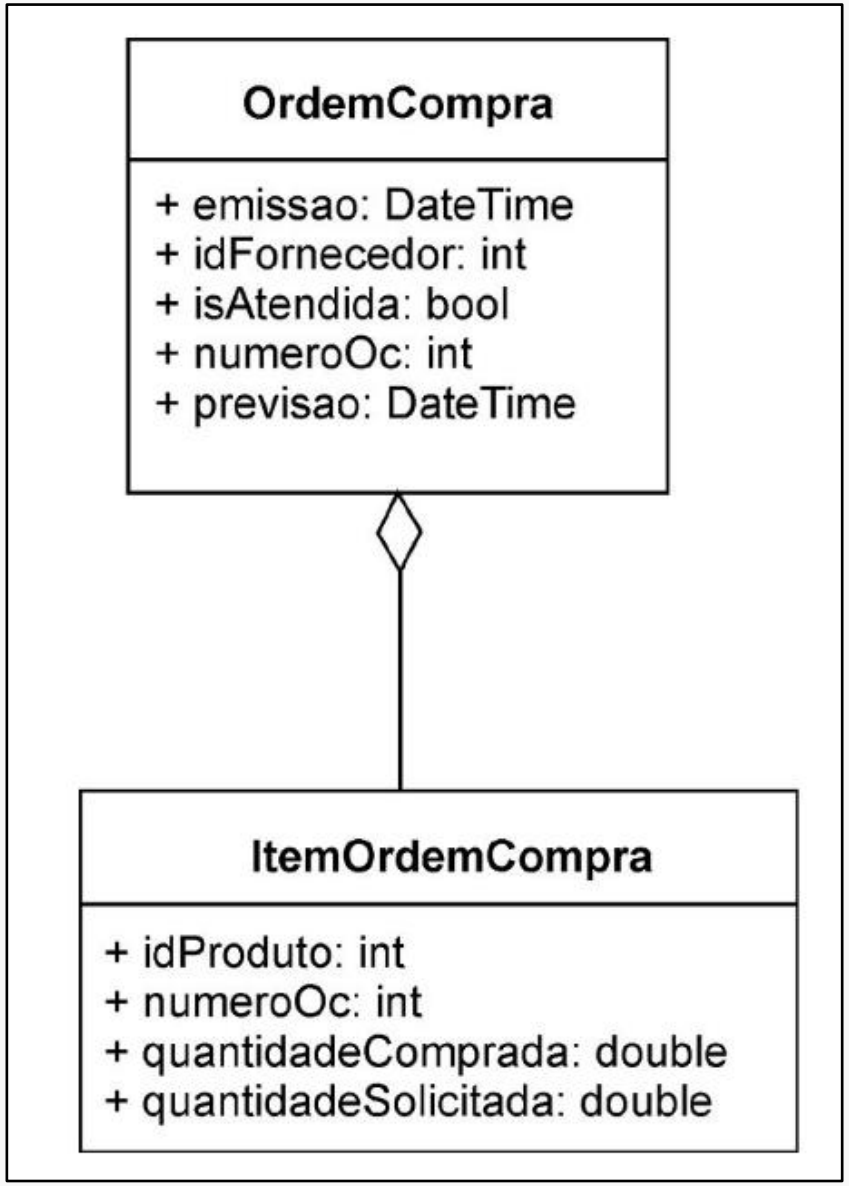


Figura 9.6. Segunda versão do diagrama de classes.

DETALHE

Associação denominada **Agregação**:



FORMAS NORMAIS - AGREGAÇÃO

Assim como acontece na modelagem de dados, o modelo orientado a objetos também precisa ser examinado a fim de eliminar redundância de valores de atributos. Esse processo é conhecido como **normalização**, um processo formal que visa simplificar os atributos das classes para garantir a estabilidade e a integridade do modelo.

O processo de normalização obedece a algumas regras, denominadas **formas normais**, as quais devem ser aplicadas o quanto antes, de preferência durante a tarefa de especificação dos atributos, como procedemos neste caso. Outros, porém, preferem levantar os atributos de todas as classes para somente depois revisar o modelo a fim de normalizá-lo.

Para obter a primeira forma normal, basta investigar se existem ocorrências repetitivas de atributos na classe. Se houver, transfira esses atributos para uma nova classe e estabeleça um relacionamento de agregação entre essa nova classe e a primeira (basta selecionar o elemento **Aggregate**, o quarto ícone do grupo **Class Relationships**, na caixa de ferramentas e fazer a associação). Na verdade, estamos pensando indiretamente na implementação do modelo em um banco de dados relacional. Para um banco orientado a objeto, em geral, não há necessidade de trabalhar a primeira forma normal.

FORMAS NORMAIS - AGREGAÇÃO

No caso da ordem de compra, quando confeccionamos os diagramas de interação, logo definimos um objeto para seus itens, mas e se tivéssemos "passado batido" e definido uma única classe OrdemCompra? Ao definir o modelo de análise, teríamos essa classe com os seguintes atributos: número, data de emissão, atendida, data de previsão, identificador do produto, quantidade solicitada e quantidade comprada...

Neste caso, cada instância ou ocorrência de ordem de compra deveria ser identificada inequivocamente. No mundo real, cada ordem de compra é identificada por um número único, mas vários produtos para uma mesma ordem de compra obrigam uma redundância indevida que precisa ser eliminada.

AGREGAÇÃO

Definição dada por PAGE-JONES, 2001.

1. O objeto agregado pode potencialmente existir sem os seus objetos constituintes. Por exemplo, mesmo que demita todos os funcionários de um departamento, você ainda tem um departamento. Entretanto, essa propriedade nem sempre é útil. Eu consigo imaginar uma cidade destruída e vazia, mas que tal um rebanho vazio? Indagando aos mestres do Zen, eu gostaria de saber: vocês conseguem vigiar um rebanho sem ovelhas?[17]
2. A qualquer hora, cada objeto pode ser um constituinte com mais de um agregado. Novamente, um agregado do mundo real pode ou não aproveitar-se dessa propriedade. Uma pessoa pode pertencer a mais do que um clube no qual freqüentemente ela faça suas refeições, mas pode uma ovelha pertencer a mais de um rebanho? Os mestres do Zen mencionados estão novamente silenciosos sobre esse ponto discutível.
3. A agregação tende a ser homeômera (cujas partes são todas semelhantes). Isso significa que os constituintes de um agregado típico pertencerão à mesma classe. Por exemplo, os constituintes de um parágrafo são todos sentenças, enquanto os constituintes de uma floresta são todos árvores.

EXEMPLO DE DIAGRAMA DE CLASSES

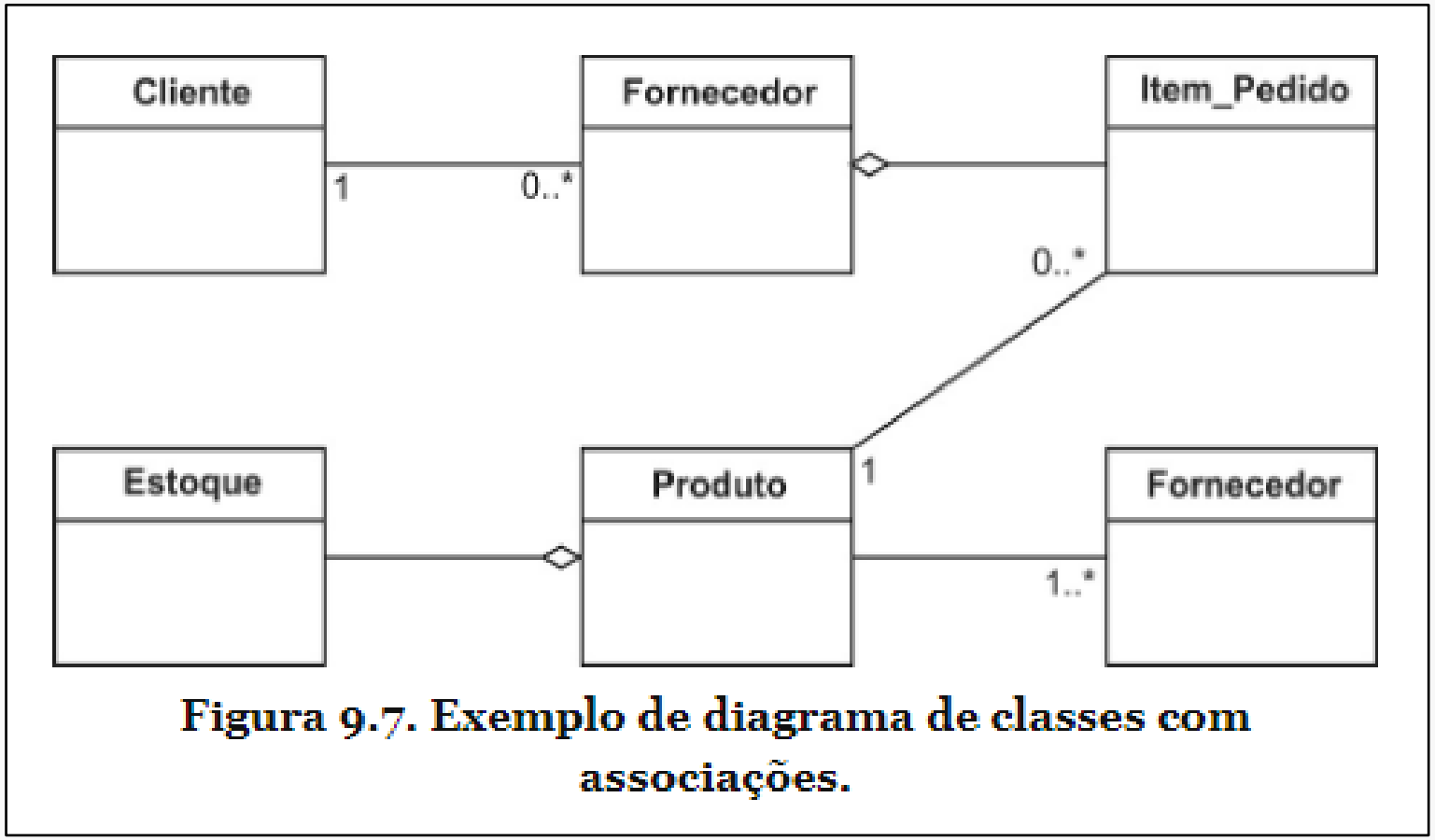


Figura 9.7. Exemplo de diagrama de classes com associações.

ASSOCIAÇÕES NO DIAGRAMA DE CLASSES

Uma associação implica que dois elementos do modelo têm um relacionamento implementado como uma instância de uma classe. Trata-se de uma conexão (ou *link* - ligação) que pode incluir nomes de papéis em cada extremidade, cardinalidade, direção e restrições.

Em outras palavras, uma associação declara que pode haver ligações entre instâncias de tipos associados. Ligação é um conjunto com um valor para cada extremidade da associação, sendo cada valor uma instância do tipo da extremidade.

Para associações com múltiplas instâncias, denominadas "**n-árias**", a multiplicidade mais baixa de uma extremidade é tipicamente **0**. Se a multiplicidade desse tipo de associação for **1** (ou mais), implica que uma ou mais ligações devem existir para todas as combinações possíveis de valores para a outra extremidade.

Como vimos, no caso das classes **OrdemCompra** e **ItemOrdemCompra**, uma associação pode representar uma agregação, isto é, um relacionamento **todo/parte**.

Uma associação é representada por uma linha simples ligando dois elementos. O símbolo de associação pode ser adornado como segue:

- O nome da associação pode ser mostrado perto do símbolo de associação, mas não muito perto de uma extremidade para evitar confusão com o nome da extremidade.
- Uma barra que aparece em frente ao nome de uma associação, ou em seu lugar se nenhum nome é mostrado, marca a associação como

derivada.

- Um texto pode ser colocado perto do símbolo de associação, mas longe o suficiente de qualquer extremidade para não ser confundido com uma propriedade em uma extremidade.

Podemos também usar uma seta para indicar que a associação deve ser lida como associação no sentido para o qual a seta aponta.

Generalizações entre associações podem ser mostradas usando uma seta de generalização entre os símbolos de associação. Uma extremidade da associação é a conexão entre a linha descrevendo uma associação e o ícone (geralmente uma caixa) descrevendo o classificador conectado. Um nome pode ser colocado perto do final da linha para mostrar o nome da extremidade de associação. O nome é opcional, isto é, pode ser suprimido.

Embora não sejam empregadas no exemplo, vale lembrar que várias outras notações podem ser colocadas perto da extremidade da linha de associação:

- uma multiplicidade;
- um texto entre chaves aplicado a uma extremidade da associação;
- **{subset <nome_propriedade>}** para mostrar que a extremidade é um subconjunto da propriedade chamada **<nome_propriedade>**;
- **{redefined <nome_extremidade>}** para mostrar que a extremidade redefine **<nome_extremidade>**;
- **{union}** para indicar que a extremidade é derivada e refere-se a uma união de seus subconjuntos;
- **{ordered}** para mostrar que a extremidade representa um conjunto ordenado;
- **{bag}** para definir que a extremidade representa uma coleção que permite ao mesmo elemento aparecer mais de uma vez;
- **{sequence}** ou **{seq}** para indicar que a extremidade representa uma sequência ordenada.

ASSOCIAÇÕES

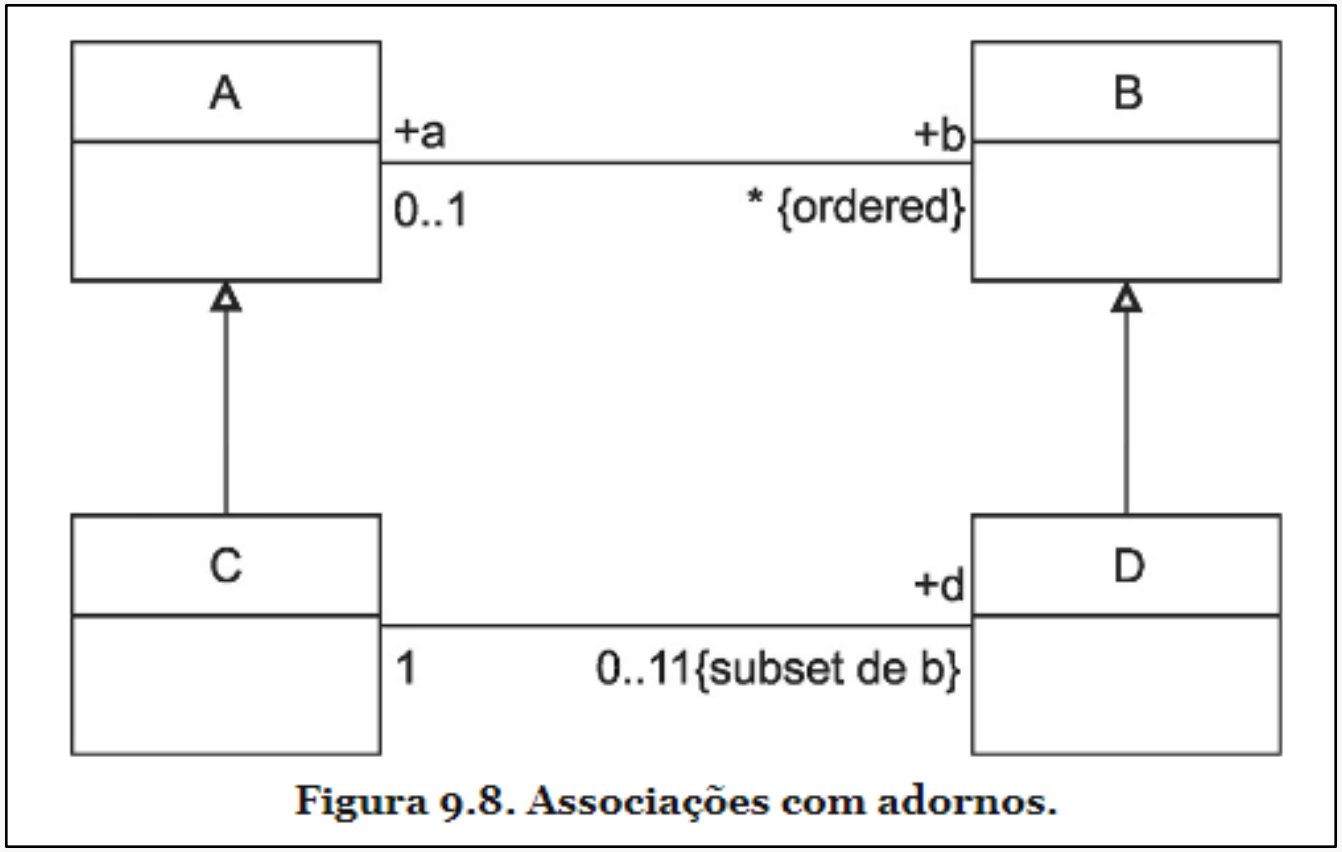


Figura 9.8. Associações com adornos.

COMPOSIÇÃO

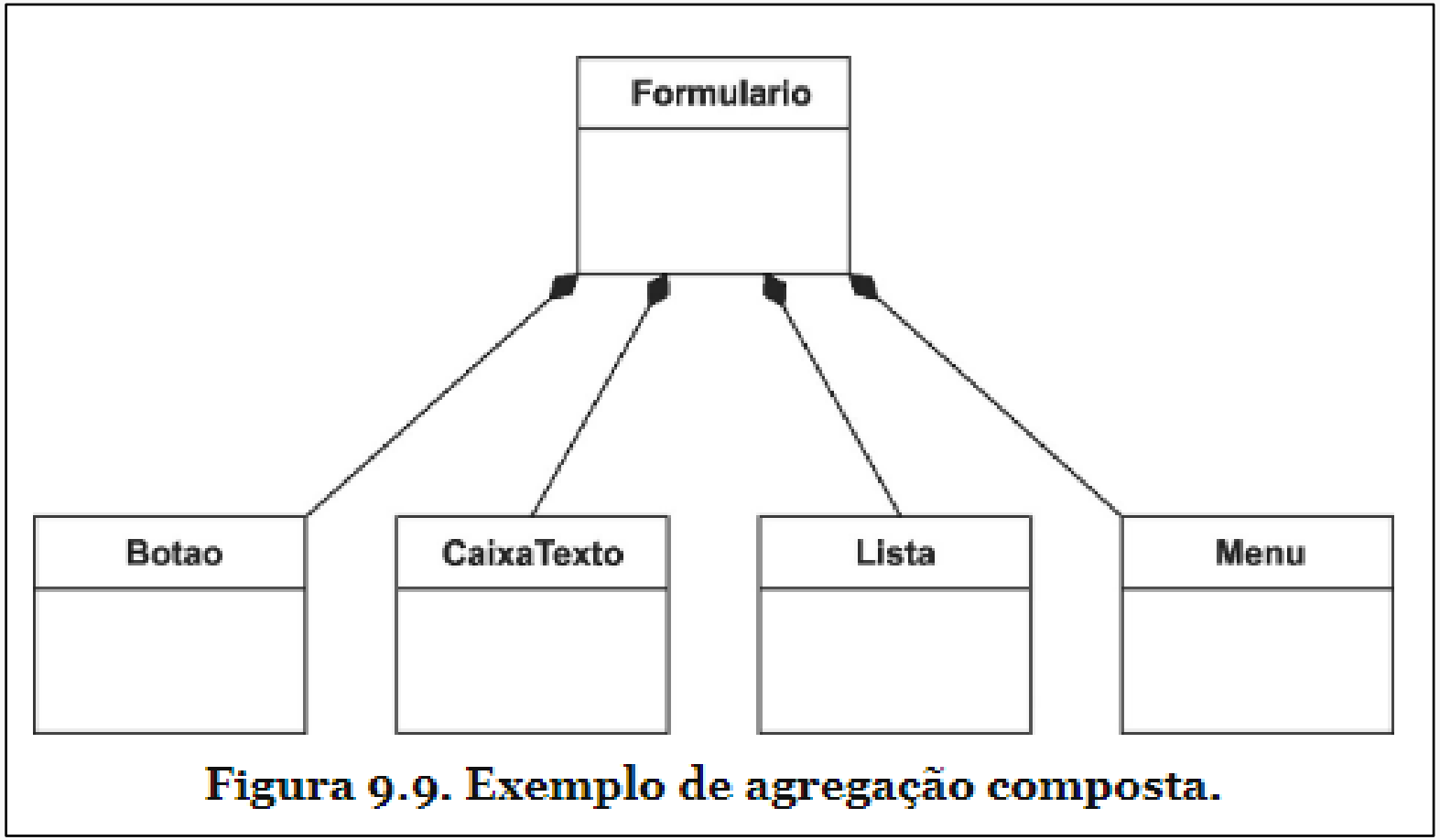


Figura 9.9. Exemplo de agregação composta.

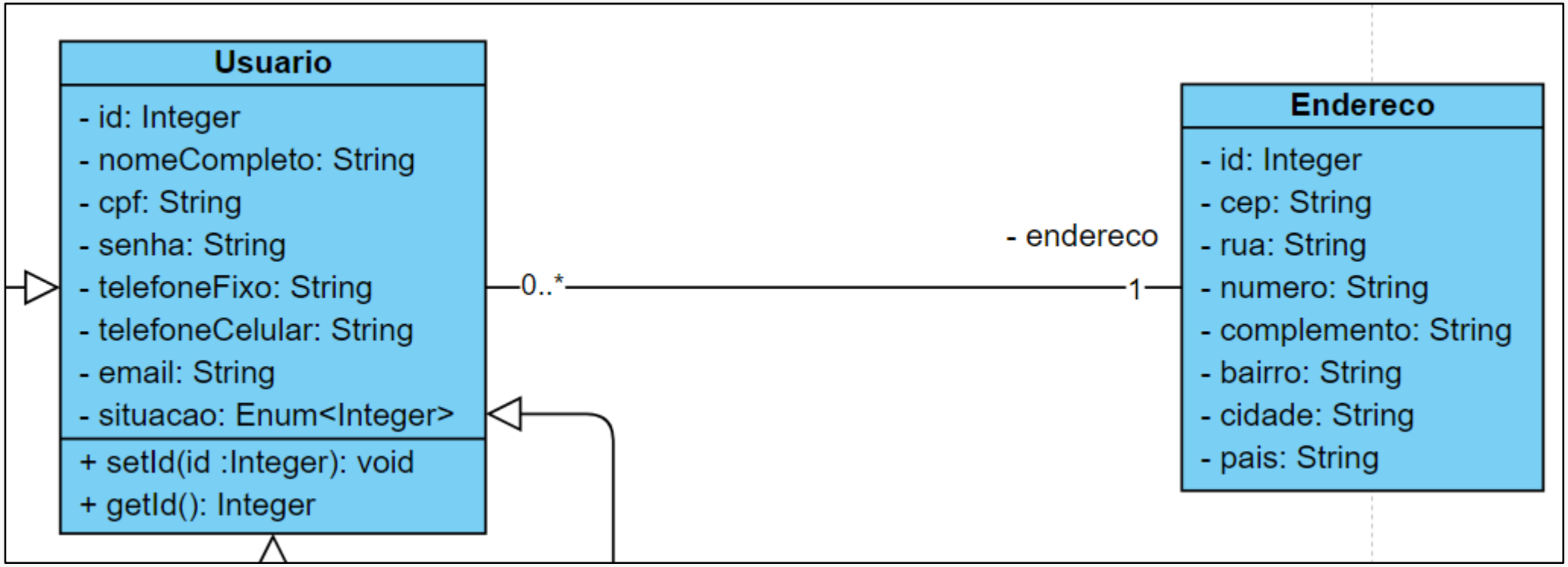
COMPOSIÇÃO

Definição dada por PAGE-JONES, 2001.

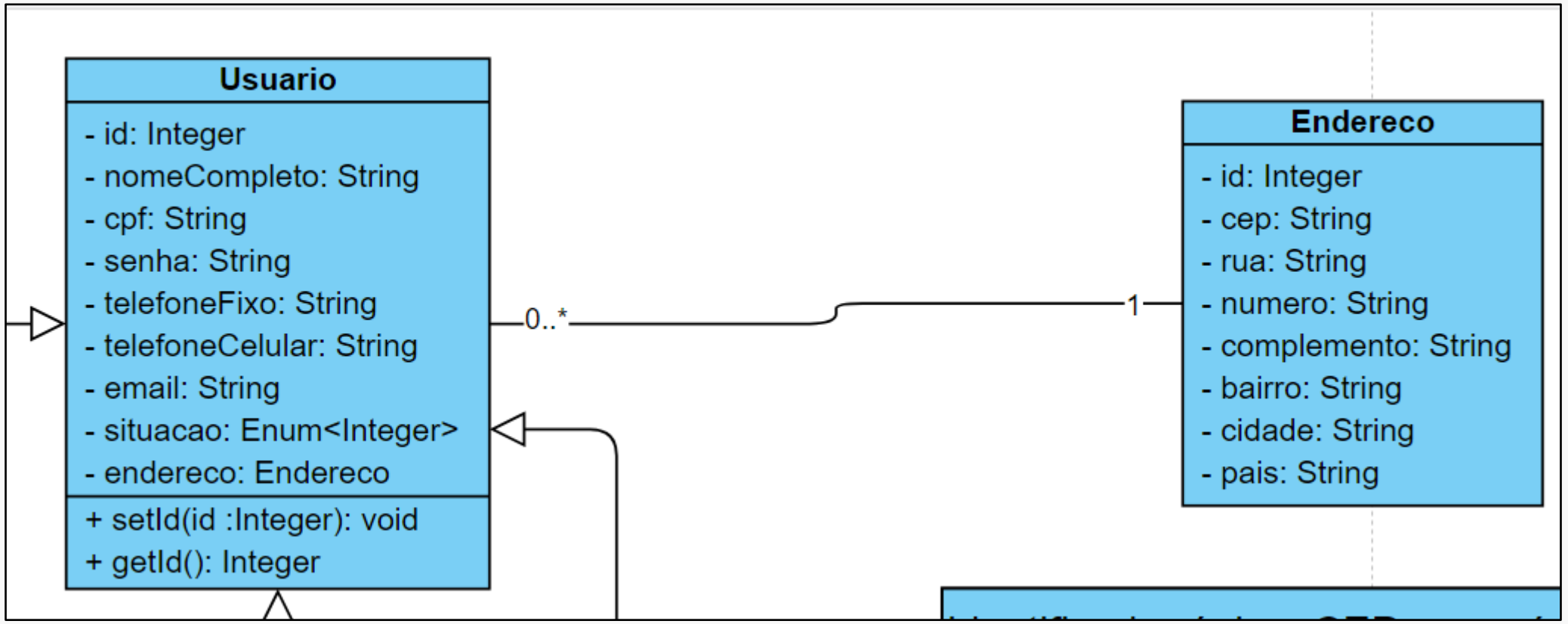
As três características mais importantes da composição são os que seguem:

1. O objeto composto não existe sem os seus componentes. Por exemplo, remova as cerdas, o cabo e a pecinha de borracha de uma escova de dentes e você não mais terá uma escova de dentes. Na verdade, a simples remoção das cerdas de uma escova de dentes faz com que ela dificilmente seja qualificada como tal. Dessa forma, o tempo de vida de um objeto composto não pode ultrapassar o tempo de vida de seus componentes.
2. A qualquer hora, cada dado objeto componente pode ser parte de somente um objeto composto.
3. A composição é tipicamente heterômera (cujas partes não são semelhantes). Os componentes provavelmente são de tipos misturados: algumas rodas, alguns eixos, alguns pedaços de madeira... e aí está a sua carroça.

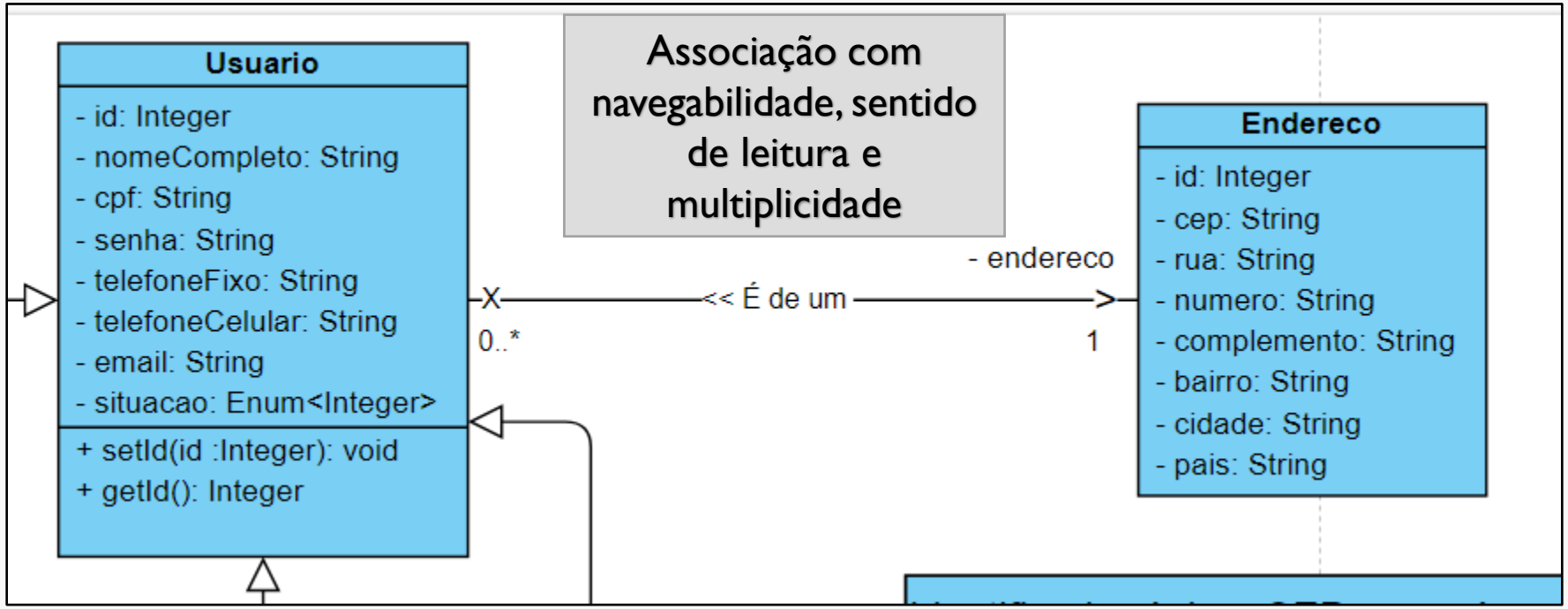
EXEMPLO – ASSOCIAÇÃO ENTRE CLASSES



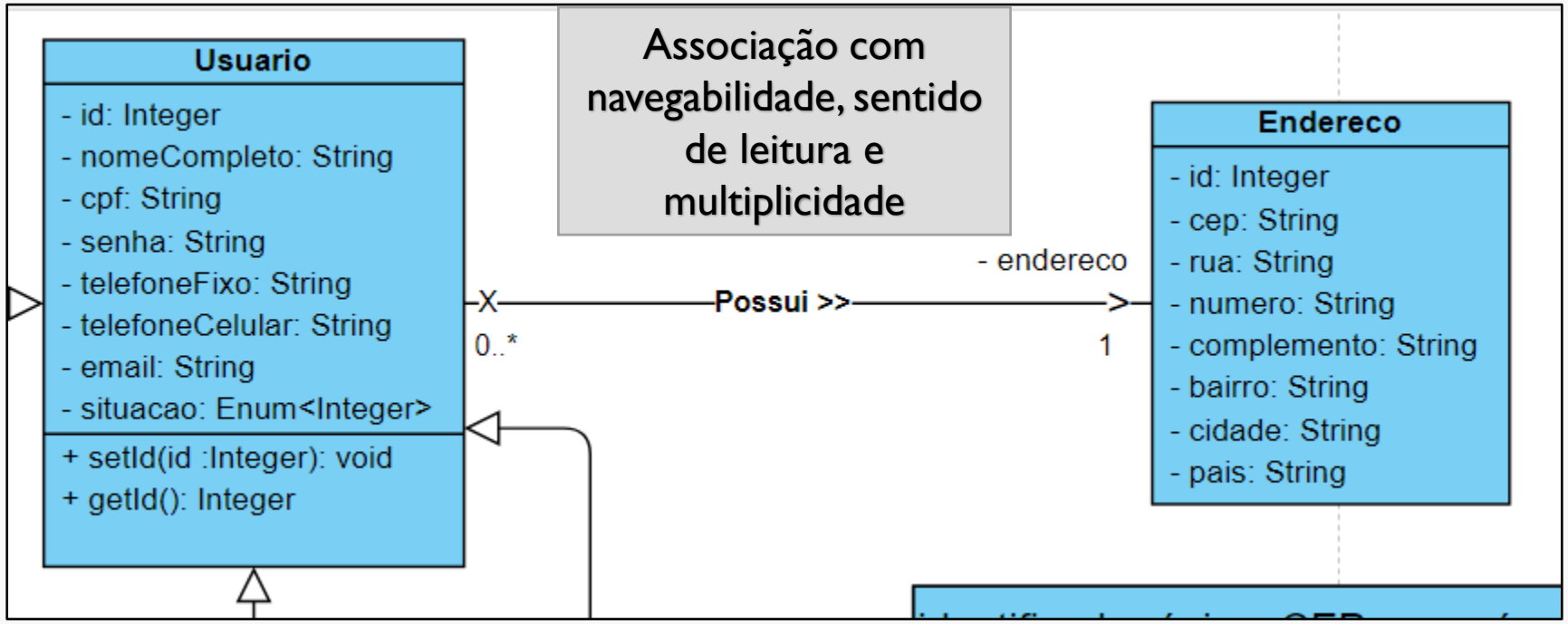
EXEMPLO – ASSOCIAÇÃO ENTRE CLASSES



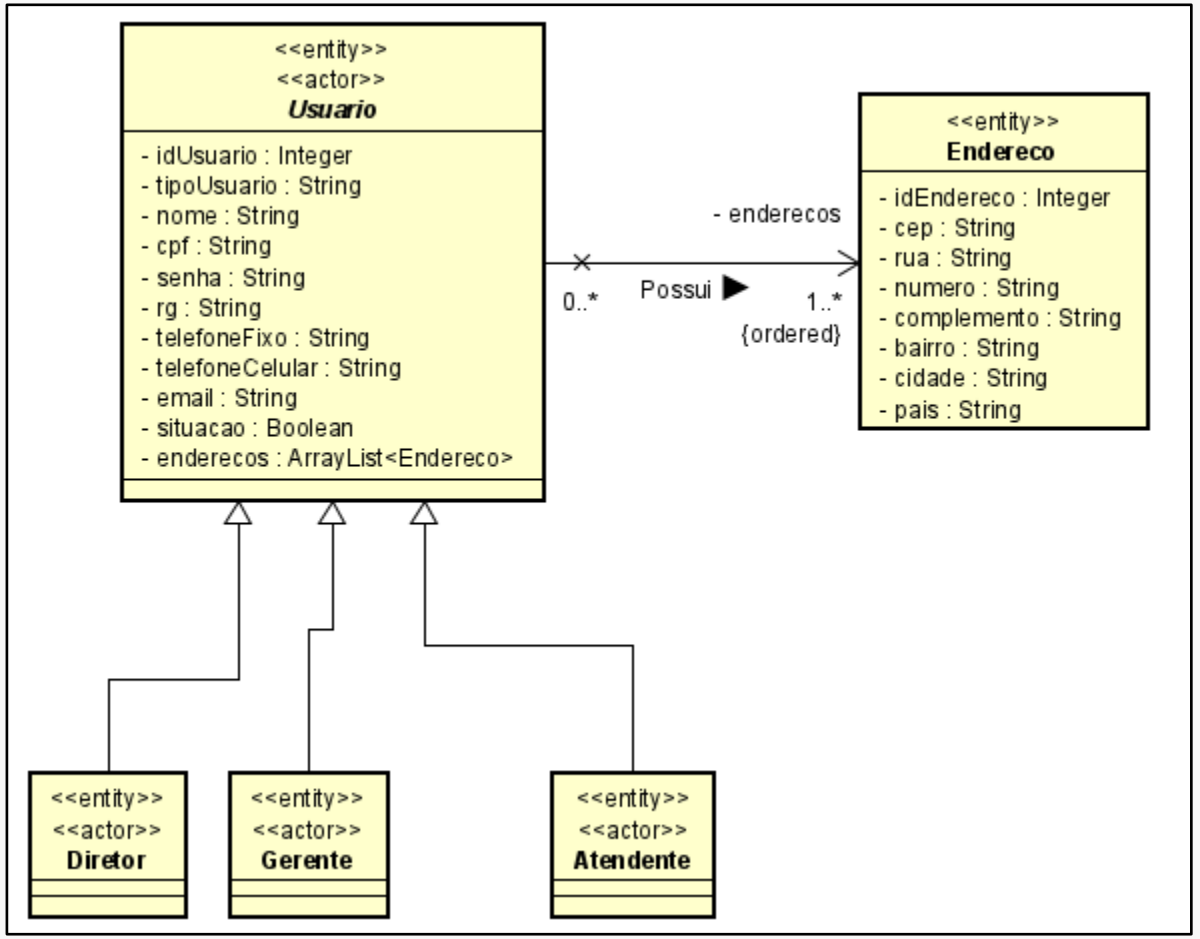
EXEMPLO – ASSOCIAÇÃO ENTRE CLASSES



EXEMPLO – ASSOCIAÇÃO ENTRE CLASSES



EXEMPLO – ASSOCIAÇÃO ENTRE CLASSES - ASTAH



PRÓXIMA AULA...

- Mais sobre diagramas de Classes;
- Encerramento das atividades letivas;

REFERÊNCIAS

- LIMA, Adilson da Silva. UML 2.5: Do requisito à solução. 1ª Edição. 2014. Editora Érica + Saraiva.
- PAGE-JONES, Meilir. Fundamentos do Desenho Orientado a Objetos com UML. Editora Person Makron Books. 2001. Disponível em: https://plataforma.bvirtual.com.br/Leitor/Publicacao/33/pdf/0?code=j5wo7p1VdfU5LWVsmWniLqUQ6i2CfmGqLqkOfjosfGzBFuSqP2b0zby90AE7ONctOboJVc7oqkh0Yk3/pWmj9A==