



ANÁLISE DE SISTEMAS

UML

por

Antônio Maurício Pitangueira

2011



Modelagem de Classes de Análise

*"O engenheiro de software amador está sempre à procura
profissional saber que tal panacéia não existe"*

Introdução

Na prática o diagrama de classes é bem mais utilizado que o diagrama de objetos.

Tanto que o modelo de objetos é também conhecido como modelo de classes.

Esse modelo durante o desenvolvimento do SSOO.

À medida que o SSOO é desenvolvido, o modelo de objetos é incrementado com novos detalhes.

Há três níveis sucessivos de detalhamento:

.

Objetivo da Modelagem de Classes

O objetivo da modelagem de classes de análise é prover respostas para as seguintes perguntas:

Por definição um sistema OO é composto de objetos...em um nível alto de abstração, que objetos constituem o sistema em questão?

Quais são as classes candidatas?

Como as classes do sistema estão relacionadas entre si?

Quais são as responsabilidades de cada classe?

Modelo de Classes de Análise

Representa termos do domínio do negócio.

idéias, coisas, e conceitos no mundo real.

Objetivo: descrever o _____ representado pelo sistema a ser desenvolvido, sem considerar características da _____ a ser utilizada.

É um dicionário "visual" de conceitos e informações relevantes ao sistema a sendo desenvolvido.

Duas etapas:

modelo conceitual (modelo de domínio).

modelo da aplicação.

Elementos de notação do diagrama de classes normalmente usados na construção do modelo de análise:

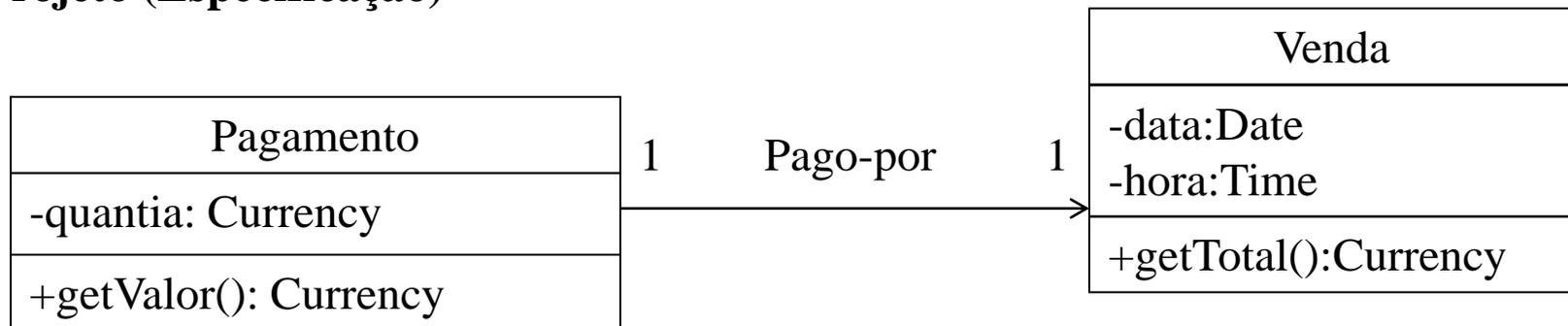
classes e atributos; associações, composições e agregações (com seus adornos); classes de associação; generalizações (herança).

Modelo de Análise: Foco no Problema

O modelo de análise não representa detalhes da solução do problema. Embora este sirva de ponto de partida para uma posterior definição das classes de software (especificação).



Projeto (Especificação)



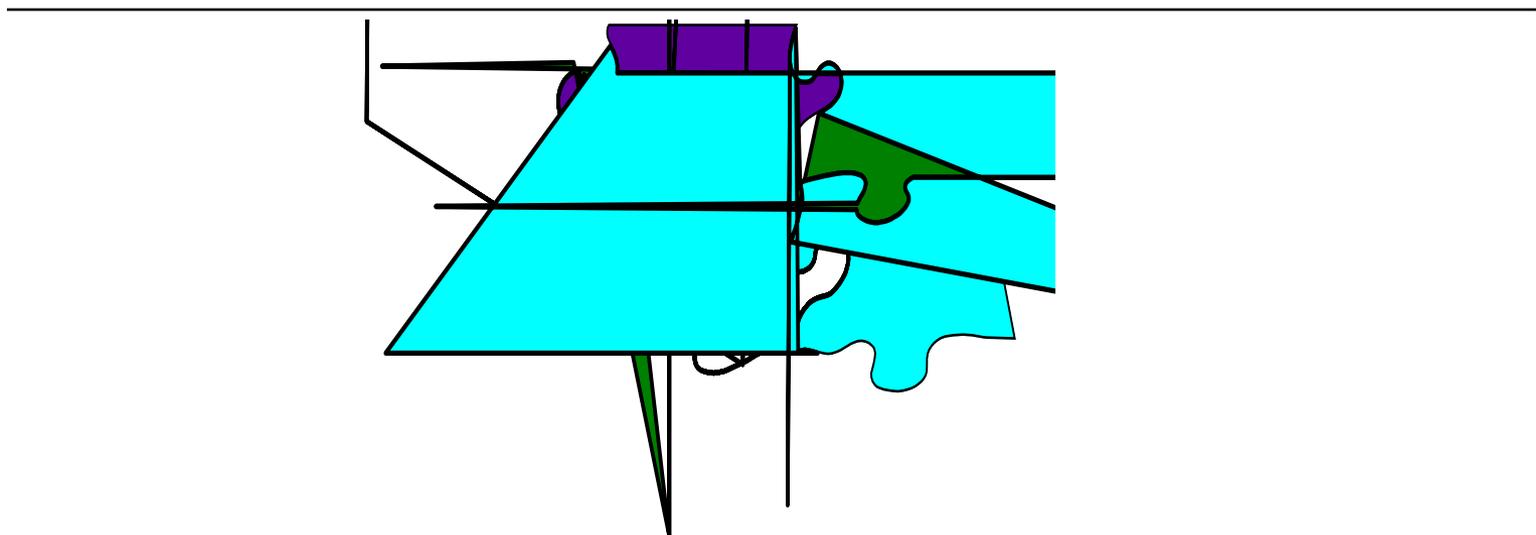


Diagrama de classes

Diagrama de Classe

O mais importante diagrama de UML.

Representação das abstrações importantes do sistema e de suas relações.

Define a arquitetura (estrutura) estática do sistema.

Serve como base para construção de outros diagramas.

Classes

Uma classe descreve esses objetos através de *atributos* e *operações*.

Atributos correspondem às informações que um objeto armazena.

Operações correspondem às ações que um objeto sabe realizar.

compartimentos exibidos.

Detalhamento utilizado depende do estágio de desenvolvimento e do nível de abstração desejado.



Exemplo (classe ContaBancária)

Diagrama de Classe

Persistência: preservar de maneira permanente os objetos de uma classe, ou seja, a classe precisa ser persistente

Semelhança com entidades definidas com tabelas em banco de dados

Diagrama de Classe

Elementos do modelo de classes:

Classes

Relacionamentos

Associação – relação de utilização

Associação Unária ou reflexiva

Associação Binária

Associação Ternária

Agregação

Composição

Generalização – relação de herança

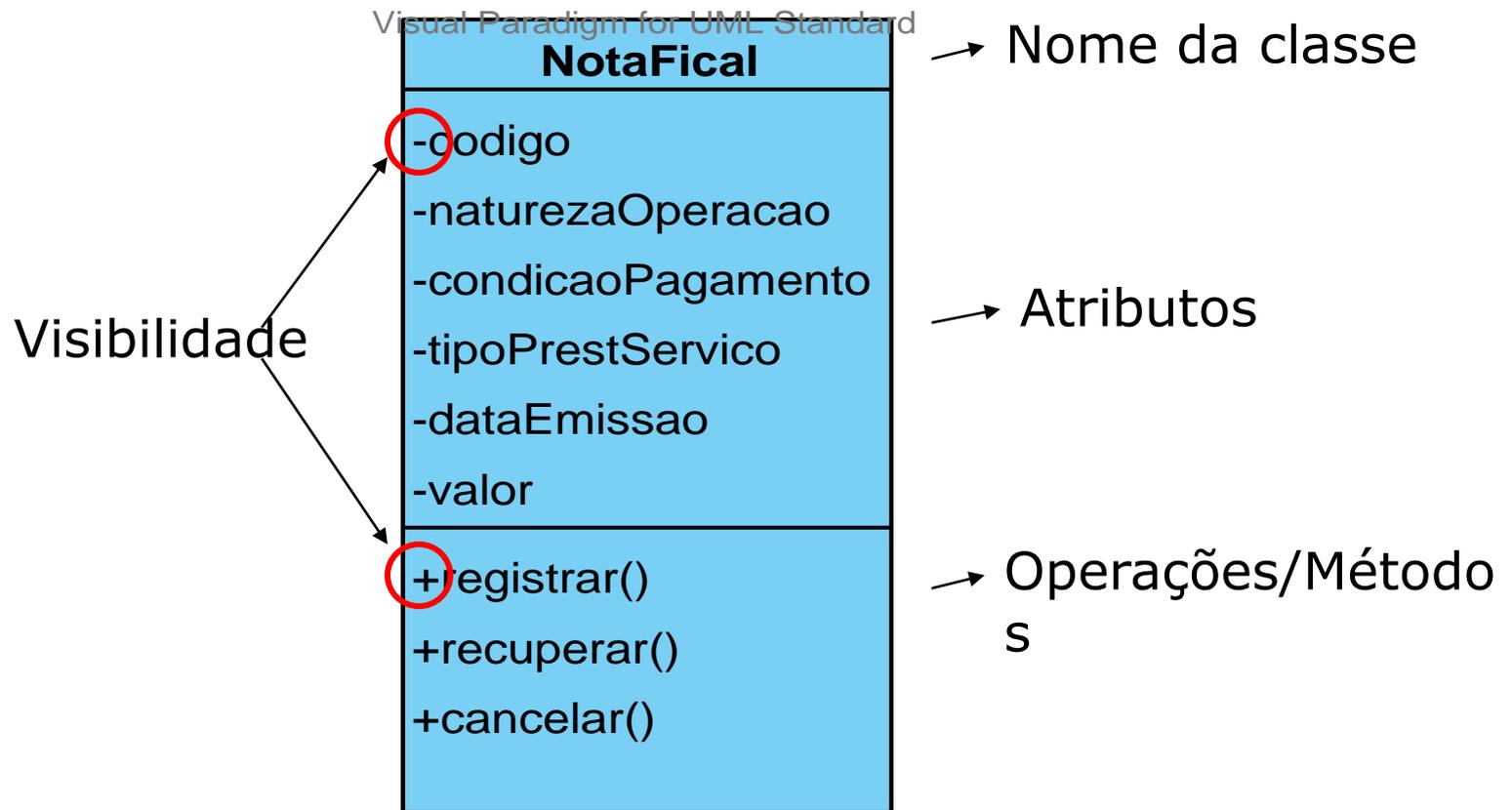
Dependência

Realização

Classe Associativa

Restrição

Exemplo de Classe



Visibilidades de Métodos e Atributos

As seguintes visibilidades são possíveis em uma operação:

Pública (+): a operação é visível para elementos de modelo

Protegida (#): a operação é visível somente para a própria classe e suas subclasses ou para **amigos** da classe (dependente de linguagem)

Privada (-): a operação é visível somente para a própria classe e para os **amigos** da classe

Identificação das Classes

A especificação do problema refere normalmente objetos concretos, mas deve-se abstrair a realidade e achar descrições comuns a vários objetos semelhantes

Escolher cuidadosamente os nomes

Identificando classes

Identificação de entidades;

Identificação de atributos;

Identificação de operações e

Identificação de relacionamento.

Identificação de entidades (classe)

Classes que permitam representar no mundo computacional elementos do mundo real

Procurar na especificação de casos de uso por conceitos que representem objetos do domínio de aplicação a ser desenvolvido

Identificação de atributos

Propriedades que caracterizam um objeto

Atributos identificados devem estar alinhados com as necessidades do usuário para o problema

Nomes significativos para os atributos

Se necessário identificar a visibilidade dos atributos

- " - " privados
- " + " públicos
- " # " protegido

Identificação de operações

Também chamadas de métodos ou serviços

Operações são ações que o objeto é capaz de efetuar

Identificar ações que o objeto de uma classe é responsável pode desempenhar dentro do escopo do sistema que será desenvolvido

Normalmente operações são públicas, permitindo sua utilização por outros objetos

Relações: Associações

Associação: é uma relação que permite especificar que objetos de uma dada classe se relacionam com objetos de outra classe.

As associações podem ter um nome. O nome pode ter uma seta associada, indicando a direção em que o nome deve ser lido.

Uma associação representa relacionamentos (ligações) que são formados entre objetos durante a execução do sistema.

Note que, embora as associações sejam representadas entre classes do diagrama, tais associações representam ligações possíveis entre os objetos das classes envolvidas.



Notação para Associações

Na UML associações são representadas por uma linha que liga as classes cujos objetos se relacionam.

Exemplos:

Multiplicidades

Representam a informação dos limites inferior e superior da quantidade de objetos aos quais outro objeto pode se associar.

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

| | |
|--|--|
| | |
| | |
| | |
| | |
| | |
| | |

Exemplos (multiplicidade)

Exemplo

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

Uma corrida está associada a, no mínimo, dois velocistas

Uma corrida está associada a, no máximo, seis velocistas.

Um velocista estar associado a várias corridas.

Conectividade

A **conectividade** corresponde ao tipo de associação entre duas classes: " " , " " e " " .

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

Acessórios para Associações

Para melhor esclarecer o significado de uma associação no diagrama de classes, a UML define três recursos de notação:

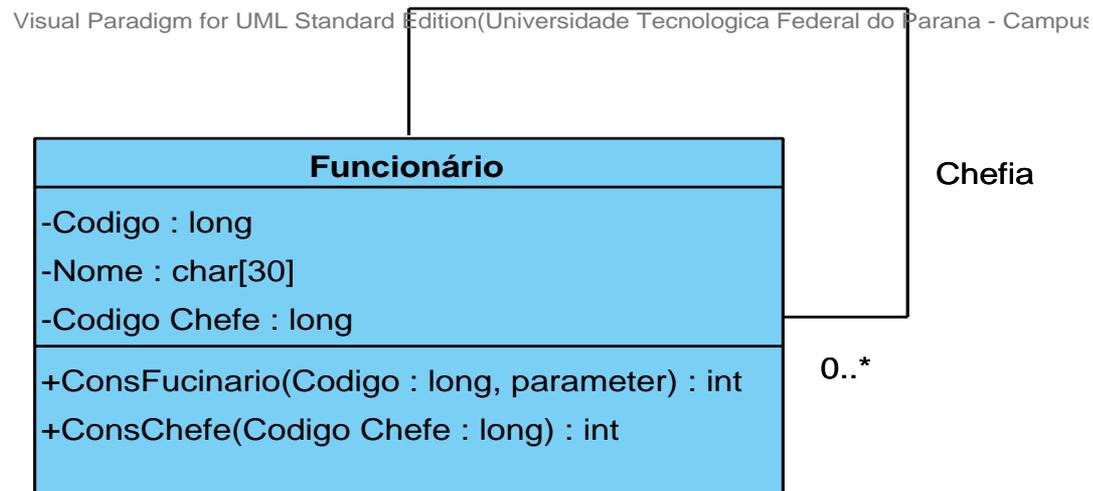
Nome da associação: fornece algum significado semântico a mesma.

Direção de leitura: indica como a associação deve ser lida

Papel: para representar um papel específico em uma associação.

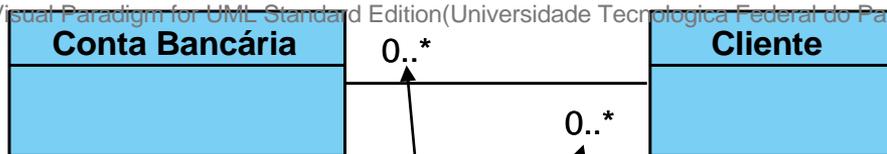
Associação Unária/Reflexiva/Auto- associação

Ocorre quando existe um relacionamento de uma classe para consigo mesma.

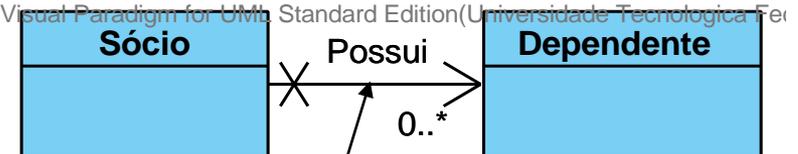


Associação Binária

Representa uma relação estrutural entre duas classes indicando que estas se comunicam através de troca de mensagem.



Multiplicidade

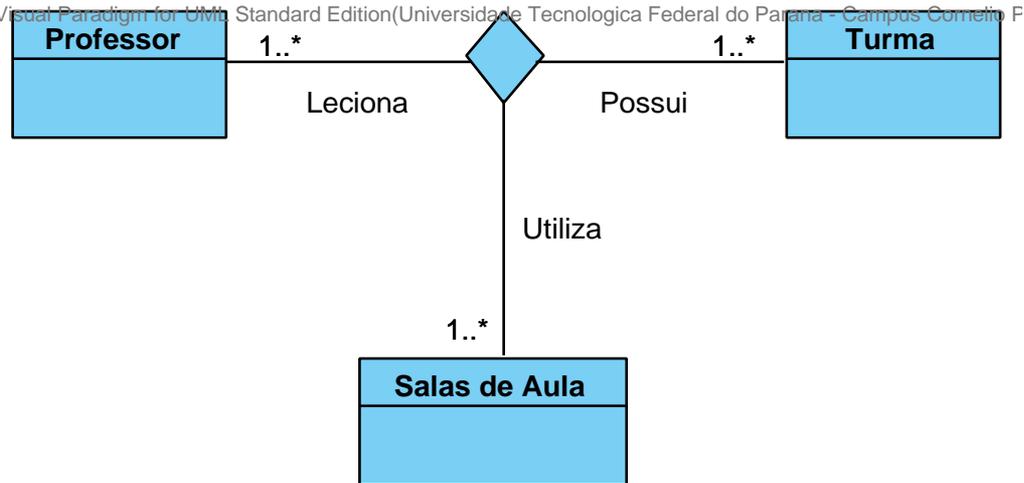


Navegabilidade da Associação

Associação Ternária ou N-ária

Conectam mais de duas classes.

Úteis para associações complexas, no entanto deve-se evitar utilizá-las, pois sua leitura é, por vezes, difícil de se interpretar.



Associação de Agregação

Tipo especial de associação.

Demonstra que as informações de um objeto (objeto-todo) precisam ser complementadas pelas informações contidas em um ou mais objetos de outra classe (objeto-parte).

Um objeto-parte não pode ser destruído por um objeto diferente do objeto-todo



Agregações e Composições

Algumas particularidades das agregações/composições:

são assimétricas, no sentido de que, se um objeto A é parte de um objeto B, o objeto B não pode ser parte do objeto A.

propagam comportamento, no sentido de que um comportamento que se aplica a um todo automaticamente se aplica às suas partes.

as partes são normalmente criadas e destruídas pelo todo. Na classe do objeto todo, são definidas operações para adicionar e remover as partes.

Se uma das perguntas a seguir for respondida com um sim, provavelmente há uma agregação onde X é todo e Y é parte.



Exemplos

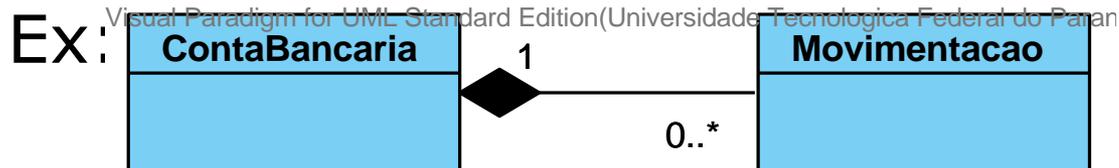
Associação de Composição

Uma variação da agregação

Representa um vínculo mais forte entre objetos-todo e os objetos-parte.

O objeto-parte é exclusivo do objeto-todo.

Um objeto-parte não pode associar-se a mais de um objeto-todo





Exemplos

Agregações e Composições

As diferenças entre a agregação e composição não são bem definidas. A seguir, as diferenças mais marcantes entre elas.

Destruição de objetos

Na agregação, a destruição de um objeto todo não implica necessariamente na destruição do objeto parte.

Pertinência

Na composição, os objetos parte pertencem a um único todo.

Por essa razão, a composição é também denominada agregação não-compartilhada.

Em uma agregação, pode ser que um mesmo objeto participe como componente de vários outros objetos.

Por essa razão, a agregação é também denominada agregação compartilhada.

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  é mais genérico que o conceito .

Exemplo: o conceito  é mais específico que o conceito .

Esse é o chamado ***relacionamento de herança***.
relacionamento de generalização/especialização
relacionamento de gen/espec

Generalizações e Especializações

Terminologia

| | | | |
|-----------|---|---|----------------------------|
| | X | . | |
| | X | . | |
| | X | . | |
| classe de | | | X classe de |
| e | | | (herança em vários níveis) |

Notação definida pela UML

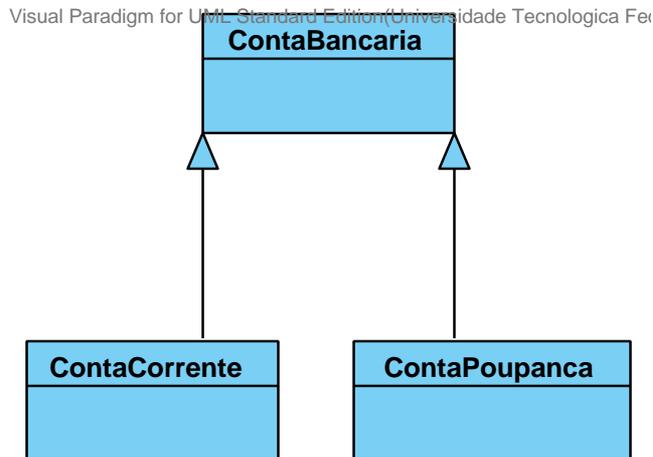
Generalização

Seu objetivo é identificar classe-mãe, chamadas gerais e classes filhas, chamadas especializadas.

Identificar similaridades de estrutura/comportamento entre várias classes.

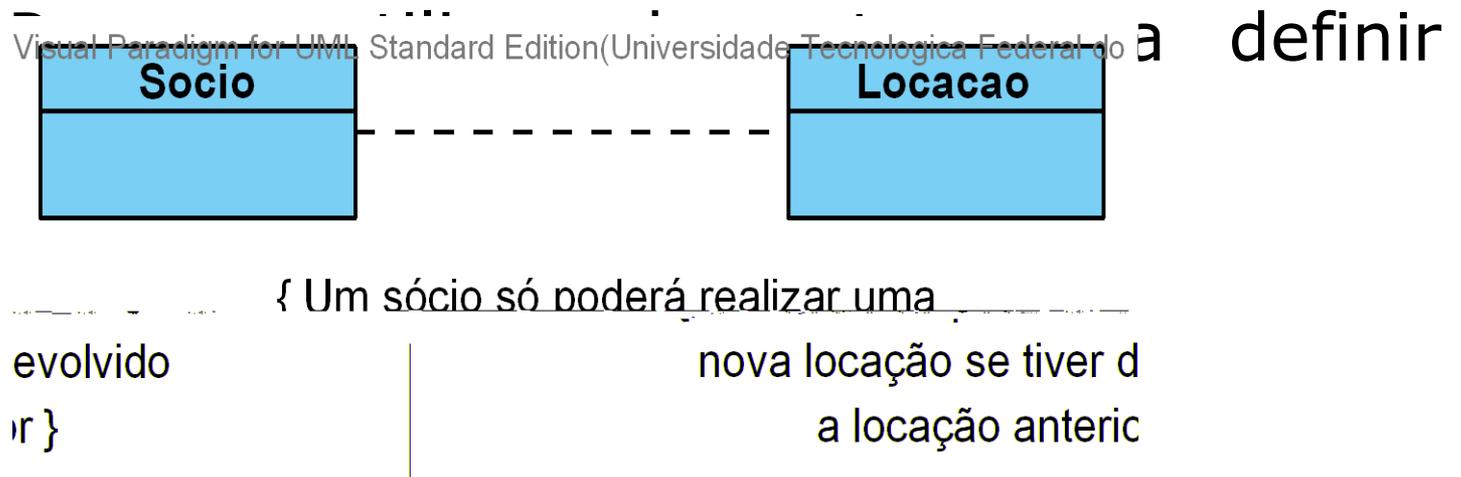
Reaproveita atributos).

(métodos e

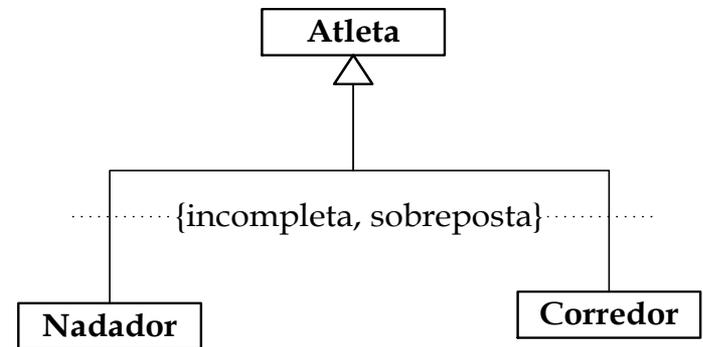
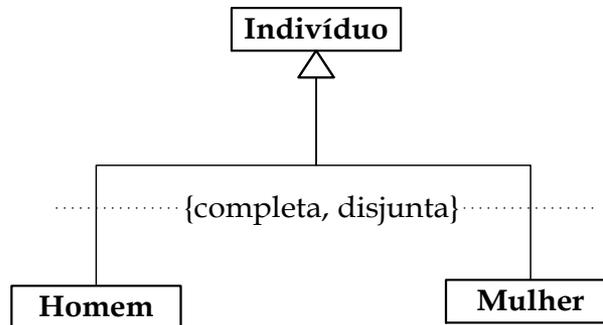
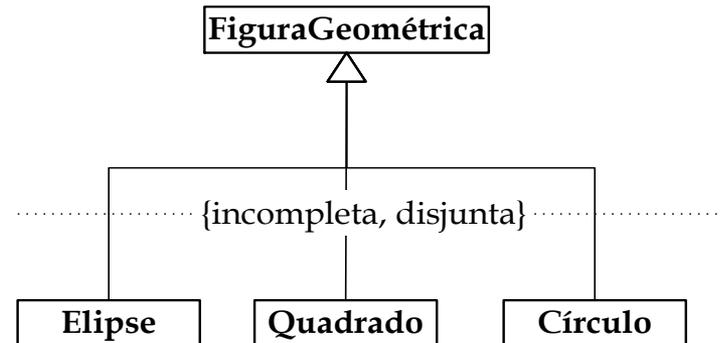
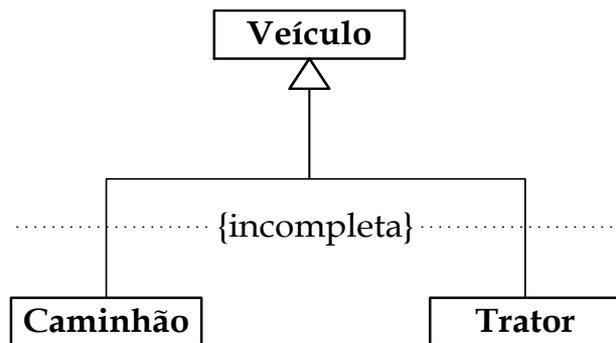


Restrição

Informações extras que definem condições a serem validadas durante a implementação dos relacionamentos entre classes. Representadas por textos limitados por chaves.



Restrições sobre gen/espec



Semântica da Herança

Subclasses herdam as características de sua superclasse

É como se as características da superclasse estivessem definidas também nas suas subclasses

Além disso, essa herança é transitiva e anti-simétrica

Note a diferença semântica entre a herança e a associação.

A primeira trata de um relacionamento _____, enquanto que a segunda representa relacionamentos _____.

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".



Herança de Associações

Não somente atributos e operações, mas também associações são herdadas pelas subclasses.

No exemplo abaixo, cada subclasse está associada a Pedido, por herança.

Propriedades da Herança

Transitividade: uma classe em uma hierarquia herda propriedades e relacionamentos de todos os seus ancestrais.

Ou seja, a herança pode ser aplicada em vários níveis, dando origem a

uma classe que herda propriedades de uma outra classe pode ela própria servir como superclasse.

Assimetria: dadas duas classes A e B, se A for uma generalização de B, então B não pode ser uma generalização de A.

Ou seja, pode haver ciclos em uma hierarquia de generalização.



Propriedades da Herança

Dependência

Não costuma ser encontrado nos diagramas de classes.

Identifica um certo grau de dependência de uma classe em relação a outra.

A mudança em uma classe implica em outro.



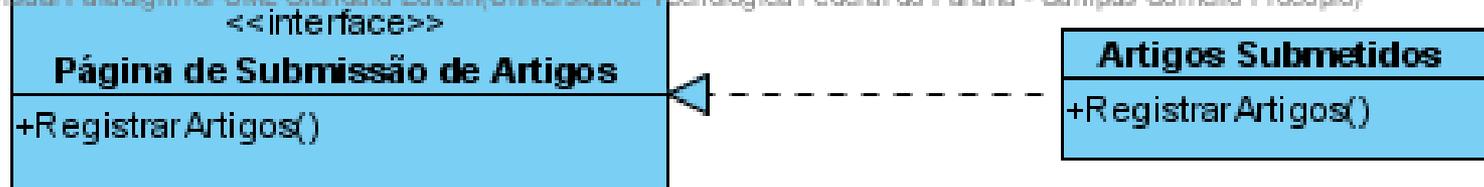
Realização

Caso especial que mistura características dos relacionamentos de generalização e dependência

Utilizada para identificar classes responsáveis por executar funções para as classes que representam interfaces.

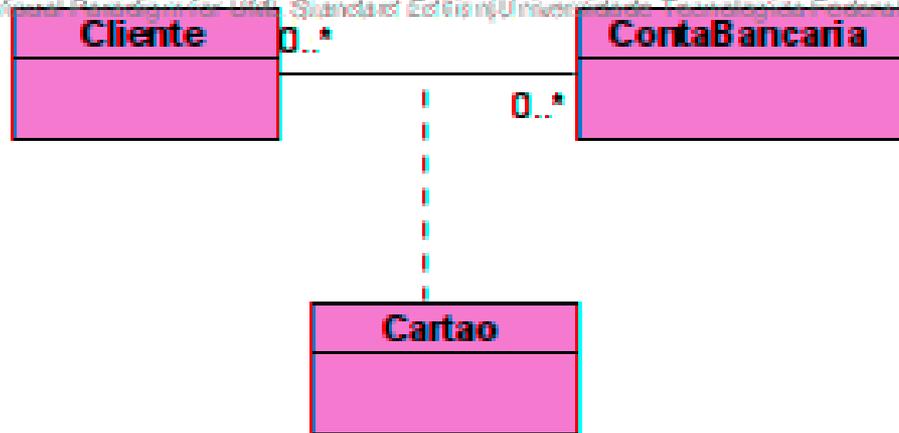
Herda o comportamento e não sua

Visual Paradigm for UML - Standard Edition (Universidade Tecnológica Federal do Paraná - Campus Cornélio Procopio)



Classe Associativa

Representa situações onde determinados atributos não podem ser colocados em nenhuma das classes participantes de uma associação, por conter informações exclusivas à ligação dos objetos das duas classes.



Como encontrar relações entre as classes identificadas

Generalização: verifica-se se há alguma relação "é um tipo de" entre as classes identificadas.

Associação: verifica-se há a necessidade de um objeto de uma classe utilizar serviços disponibilizados por um objeto de outra classe ou, simplesmente, "conhecer" o outro objeto.

Auto-Associação: verifica-se se existe alguma classe que necessita ser relacionada a ela mesma.

Como encontrar relações entre as classes identificadas (continuação)

Agregação: verifica-se se há alguma relação "parte de" entre as classes.

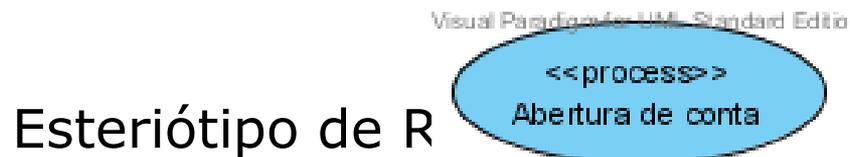
Composição: verifica-se se há alguma relação "parte de" forte entre as classes.

Navegação: verifica-se se existem "navegações" desnecessárias entre classes.

Classe associativa: verifica-se se existem informações que precisam estar vinculadas à associação de dois objetos (mas não a um deles em particular)

Estereótipos

Estereótipos são maneiras de destacar ou diferenciar um componente ou relacionamento de outros componentes ou relacionamentos iguais, atribuindo-lhe características especiais ou modificando-as de alguma forma (GUEDES, 2005)



Estereótipos <<entity>>

Tornar explícito que uma classe é uma entidade, ou seja, a classe contém informações recebidas ou geradas por meio do sistema

Classes com estereótipo <<entity>> também fornecem a informação de normalmente terão muitos objetos e que terão um período de vida longo

Significa também que estes objetos da classe precisam ser preservados (persistentes)

Estereótipos <<boundary>>

Conhecido com estereótipo de fronteira

Importante quando é preciso definir a existência de uma interface para o sistema

Identifica que uma classe serve de comunicação entre atores externos e o sistema

Muitas vezes uma classe <<boundary>> é associada a própria interface do sistema

Pode haver a interação com uma classe <<control>>

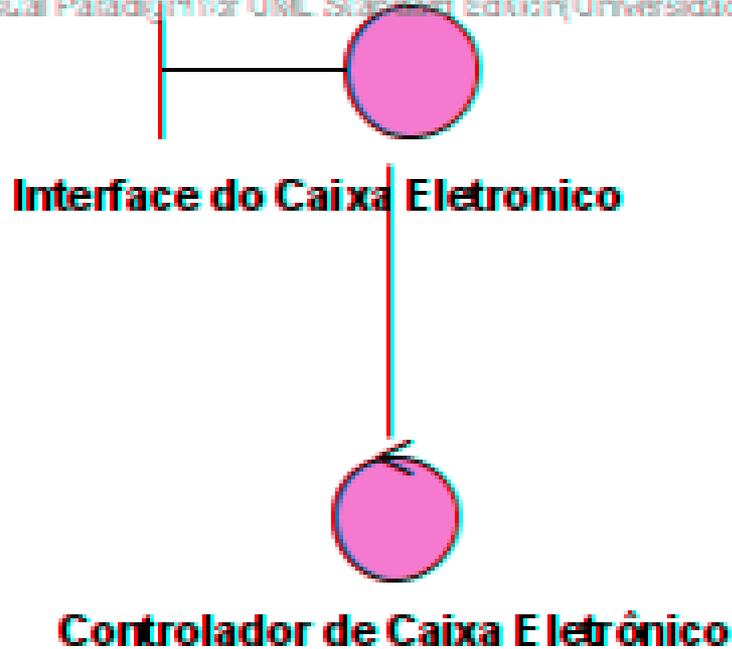
Esteriótipos <<control>>

Identifica classes que servem de intermédio entre as classes <<boundary>> e outras classes do sistema.

Interpretam os eventos ocorridos sobre os objetos <<boundary>>, como movimentos do mouse ou pressionamento de um botão e retransmitem-nos para os objetos das classes de entidades que compõem o sistema⁵²

Exemplo

Visual Paradigm for UML Standard Edition (Universidade Te...



Visual Paradigm for UML Standard Edition (Universidade Te...



