

Aula 3

Professora

Drucilla do Bem Oliveira

drucoliveira@hotmail.com

Ligações e Associações

- Meios de estabelecer relacionamentos entre objetos e classes
- Ligação
 - conexão física ou conceitual entre instâncias de objetos
 - Tupla - lista ordenada de instâncias de objetos
- Uma ligação é uma instância de uma associação
- Associação
 - descreve um grupo de ligações com estrutura e semântica comuns.
 - todas as ligações de uma associação interligam objetos de uma mesma classe.
 - verbos: conjunto de potenciais ligações
- As associações são intrinsecamente bidirecionais
- As associações são muitas vezes implementadas em lp's como ponteiros de um objeto para outro.
- Um ponteiro é um atributo de um objeto que contém a referência explícita de outro objeto.
- Uma ligação mostra um relacionamento entre dois ou mais objetos
- A modelagem da ligação como ponteiro encobre o fato de que a ligação não faz parte de qualquer dos objetos mas depende de todos.
- Cada associação no diagrama de classes corresponde
 - a um conjunto de ligações no diagrama de instâncias
 - da mesma forma como cada classe corresponde a um conjunto de objetos
- Notação para associações
 - uma linha entre classes
- Associação um-para-um

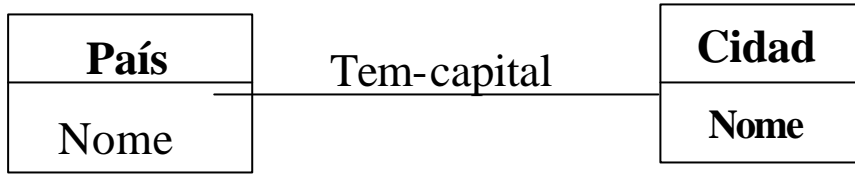


Diagrama de Classes

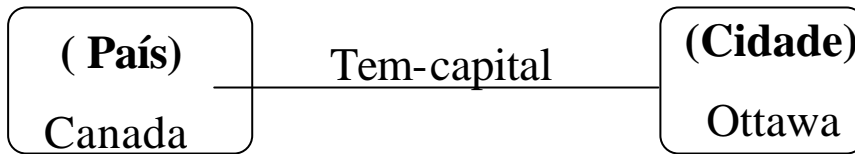
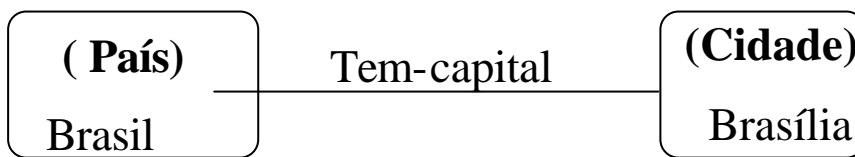


Diagrama de Instâncias



- Associação muitos-para muitos

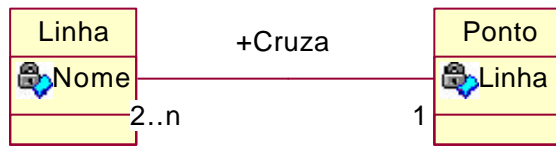


Diagrama de Classes

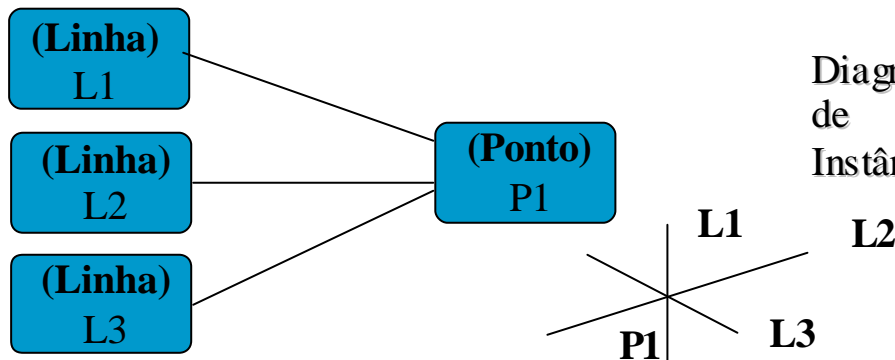
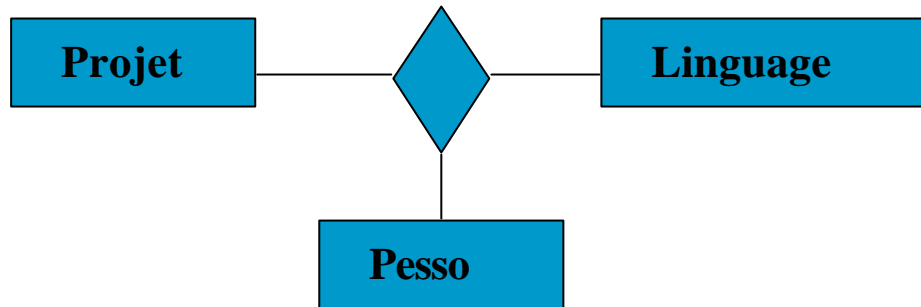


Diagrama de Instâncias

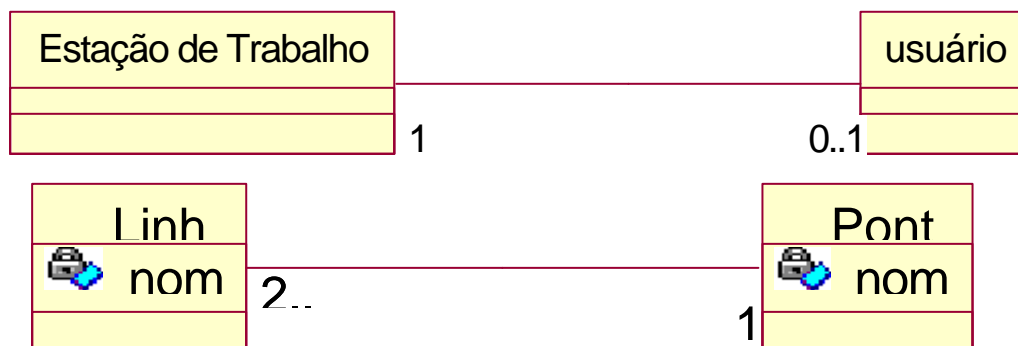
- As associações podem ser binárias, ternárias ou de ordem mais elevada.
- A grande maioria é binária qualificada (um tipo especial de associação ternária discutida mais adiante)
- As associações de ordem mais elevadas são difíceis de serem implementadas e entendidas

- Associação Ternária: unidade atômica e não pode ser subdividida em associações binárias sem perda de informação
- Símbolo: losângulo
- Nomes das associações: opcionais, critério de modelagem - deixadas sem nome quando podem ser identificadas pelas classes



Multiplicidade

- Especifica quantas instâncias de uma classe relacionam-se a uma única instância de uma classe associada
- A multiplicidade restringe o número de objetos relacionados
- Muitas vezes descrita como “uma” ou “muitas”
- Número na extremidade de uma associação como:
 - 1 : exatamente 1
 - 1+ : um ou mais
 - 3-5 : três a cinco inclusive
 - 2,4,8 :ou 2 ou 4 ou 8



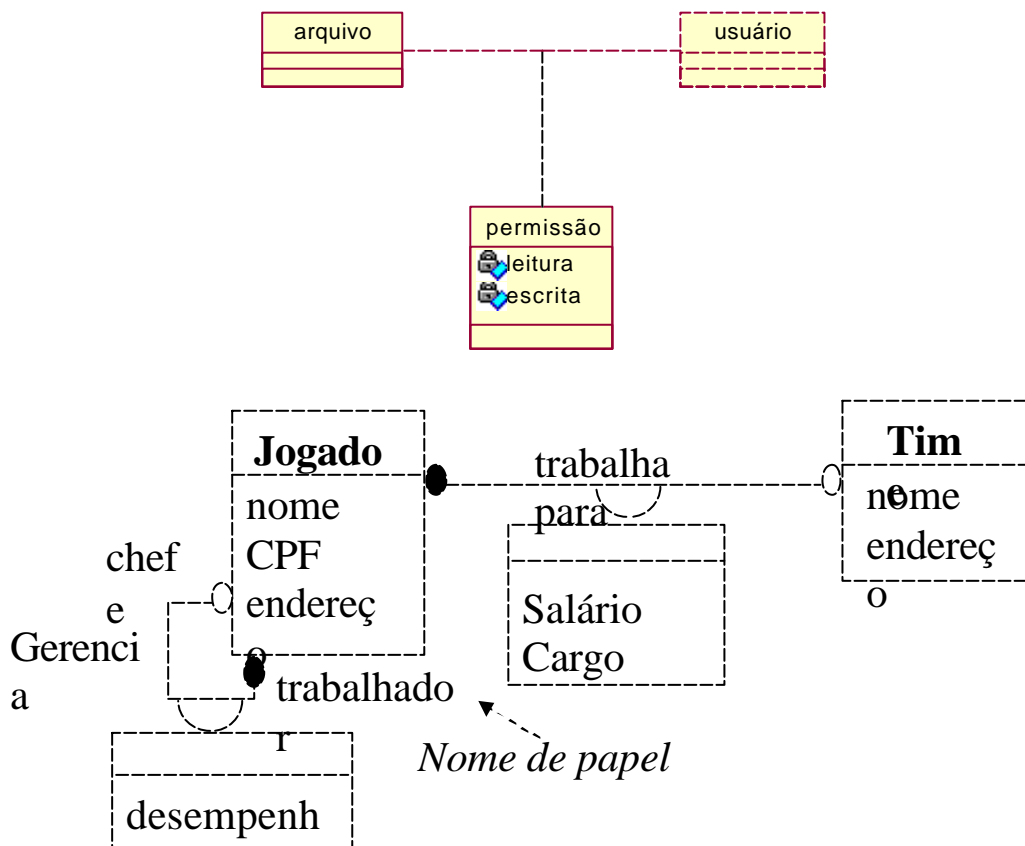
- Requisitos vagos = Multiplicidades incertas
- Não devemos nos preocupar demais no início do desenvolvimento
- O mais importante é a distinção entre a existência entre um e muitos

A Importância das Associações

- Não é conceito novo
- Amplamente utilizada na modelagem de bancos de dados
- Poucas LP's suportam associações explícitas
- Associações: modelam uma informação subordinada a mais de uma classe

Atributos de Ligação

- Atributo : propriedade dos objetos de uma classe
- Atributo de Ligação: propriedade das ligações de uma associação



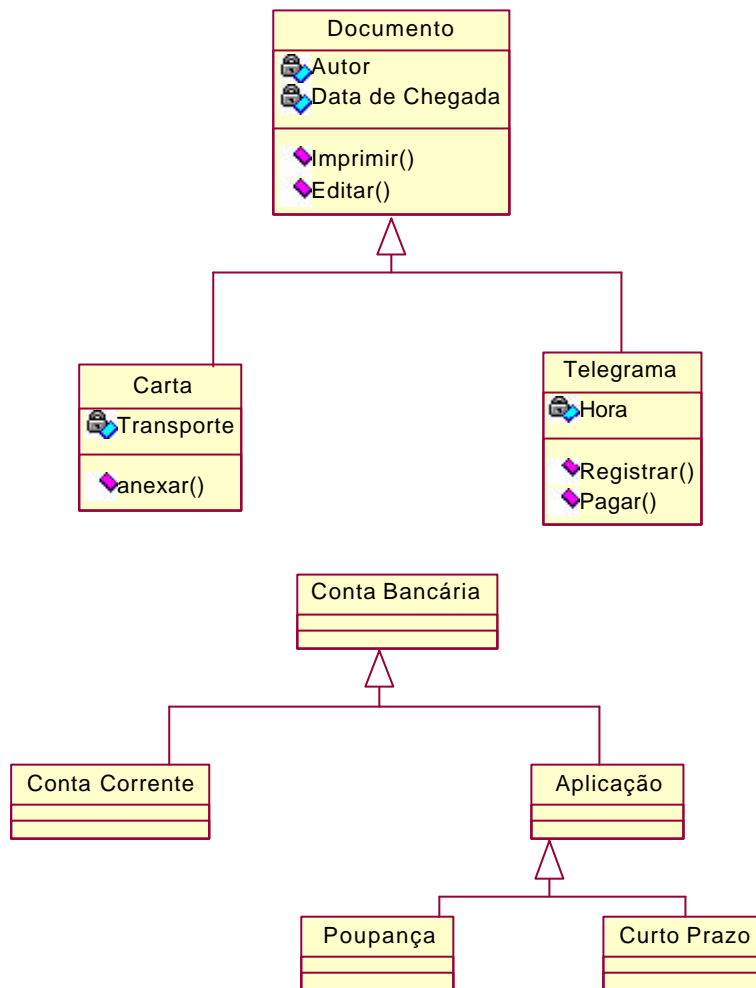
Associação como Classe

- Uma ligação torna-se a instância de uma classe
- O quadro de atributos de ligação é um caso especial de uma associação como classe: pode ter um nome e operações além de atributos

Herança x Agregação

Herança

- Mecanismo para derivar novas classe a partir de classes existentes
 - é um processo de refinamento
- Uma classe derivada herda
 - representação de dados e operações de sua classe base
 - seletivamente pode adicionar novas operações
 - estender a representação de dados
 - redefinir a implementação das operações
- Classe base proporciona a funcionalidade que é comum a todas as classes derivadas
- Uma classe derivada proporciona a funcionalidade adicional que especializa o seu comportamento
 - classe derivada, subclasse ou classe filha
 - herda atributos e métodos de sua classe base, superclasse ou classe pai
- Classes Ancestrais
 - das quais a superclasse herda
- Classes Descendentes
 - que herdaram de uma subclasse



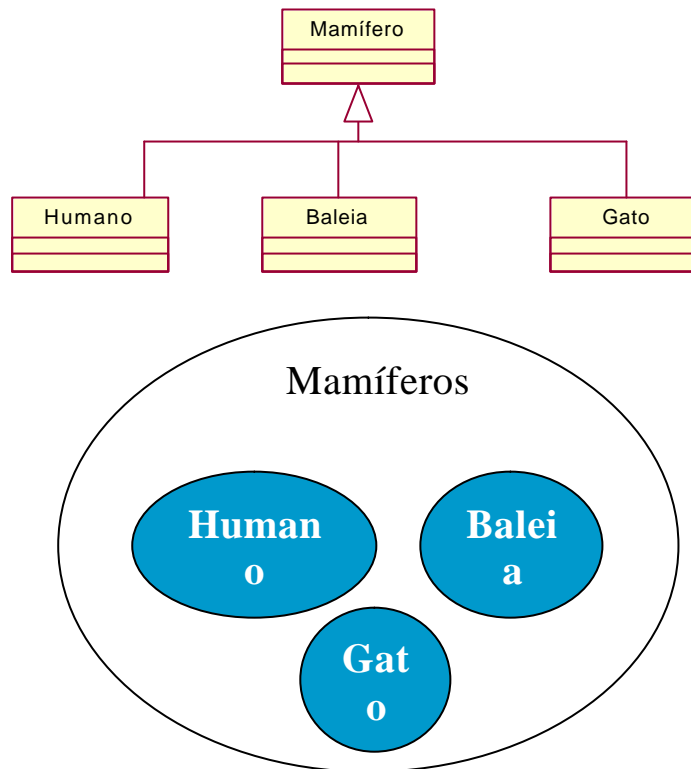
Usos de Herança

- Dois tipos principais
 - de implementação
 - usada para implementar um tipo abstrato de dados que é similar a outros tipos já existentes
 - Pilha como subclasse de lista para reusar a implementação de lista
 - não é um bom uso
 - de comportamento
 - construção de hierarquia de tipos composta de subtipos e supertipos

Herança de Comportamento

- S é um Subtipo de um tipo T se S proporciona o mesmo comportamento de T

- conjunto/subconjunto
 - um subtipo é um subconjunto do supertipo
 - subtipo pode ter propriedades e operações adicionais ao supertipo
 - preocupa-se com *compartilhamento de comportamento*
 - generalização/especialização



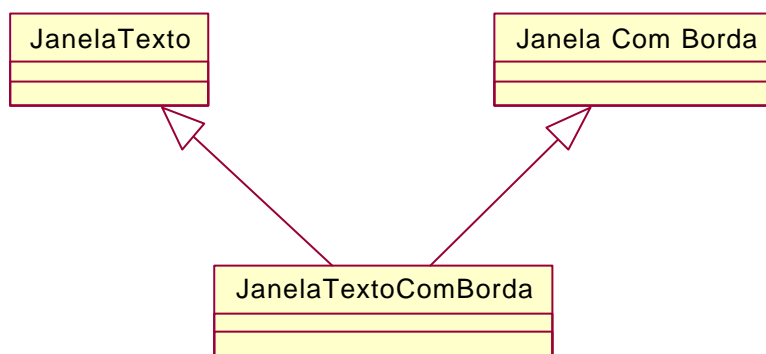
- Quando uma classe B herda comportamento de A
 - assume-se que uma instância de B é uma instância de A
 - uma pilha não é uma subclasse de lista
 - um gato é uma subclasse de mamífero
 - por outro lado um gato difere de uma baleia e portanto pode ter suas próprias operações

Agregação

- Para compartilhar comportamento onde não existe generalização/especialização
 - recomenda-se **agregação** ao invés de herança de implementação (ou derivação)

Herança Múltipla

- A hierarquia de classes gerada com herança simples é uma *árvore*
- Herança Múltipla
 - combina-se várias classes para produzir uma nova classe
 - a hierarquia torna-se um grafo acíclico dirigido
 - uma classe pode ter mais de um predecessor imediato
- Um objeto `JanelaTextoComBorda` pode herdar seu comportamento das classes `JanelaTexto` e `JanelaComBorda`



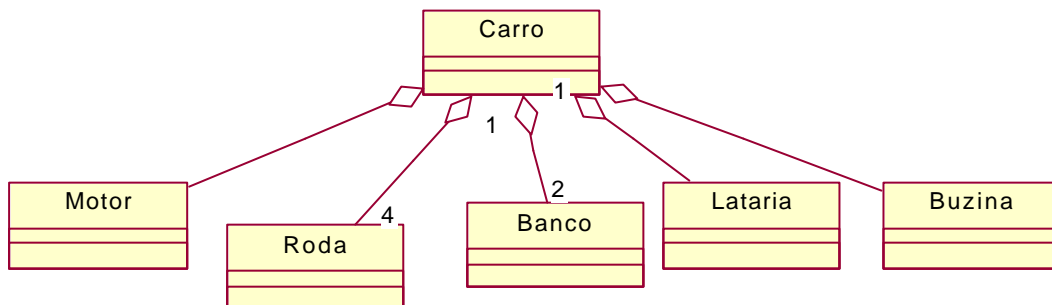
- Existe a possibilidade de conflitos
 - operações e atributos com o mesmo nome e semântica diferentes
- Soluções
 - linearização
 - renomeação
 - qualificação
- Linearização
 - ordem linear e total das classes
 - a busca do atributo ou método é feita na ordem estabelecida
- Renomeação
 - requer a renomeação de atributos e operações conflitantes
- Qualificação
 - quando houver ambigüidade, requer o uso de operador de escopo
- As duas últimas promovem maior flexibilidade
- É poderosa para especificação de classes e reuso

- Desvantagem:
 - perda da simplicidade conceitual
 - adiciona mais complexidade ao problema

Agregação x Generalização

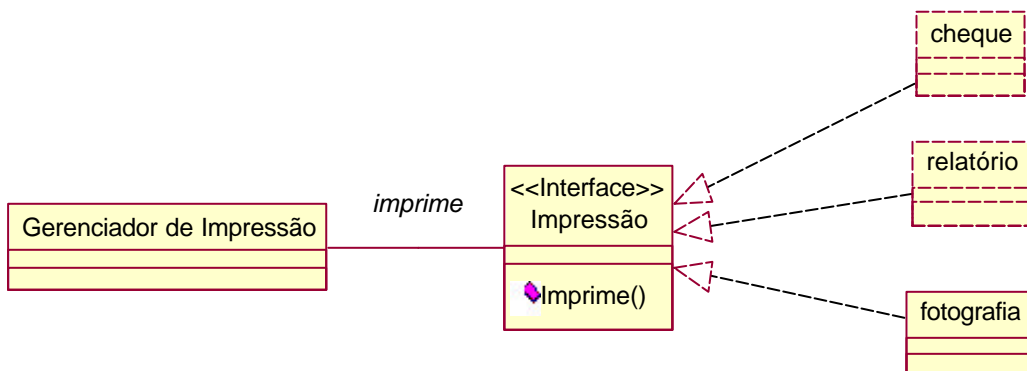
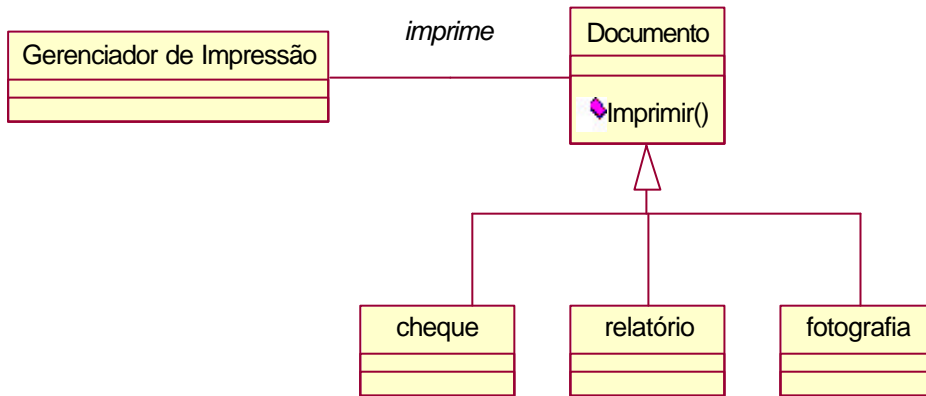
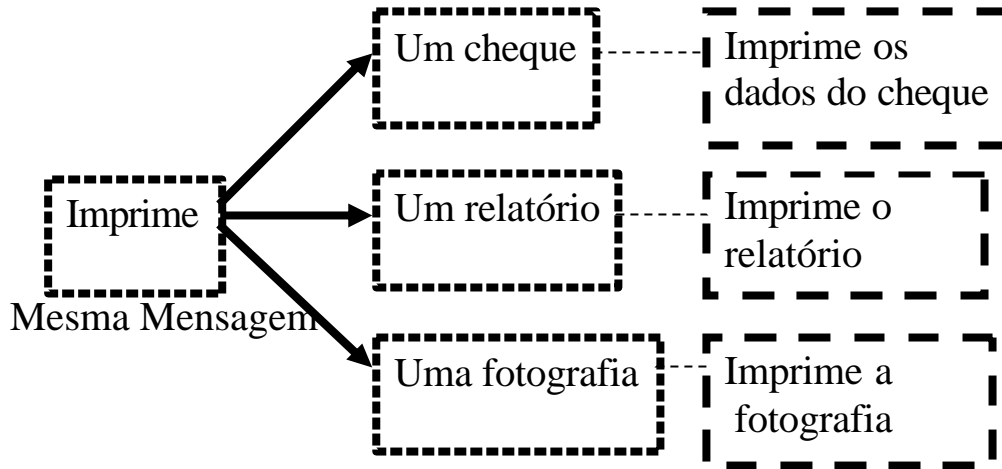
- Generalização é conveniente para a construção e manutenção de classes relacionadas
- Um carro é composto de lataria, motor bancos e rodas
 - jamais poderia ser definida como especialização destas outras classes
 - um objeto agregado pode referenciar mais de uma instância de uma mesma classe

Agregação



Polimorfismo

- No contexto OO significa que: o objeto A que requisita uma operação de outro objeto B, não necessita conhecer a classe do objeto B
- B pode pertencer a uma classe arbitrária
- Polimorfismo significa que instâncias diferentes de classes diferentes podem estar associadas a uma outra instância
 - o estímulo (a mensagem que vai através desta associação) pode ser interpretado diferente - função da classe receptora
- Muitas Formas
 - Um mesmo nome para funções com um mesmo comportamento aplicada a classes diferentes
 - Funções são polimórficas quando seus parâmetros podem ter mais de um tipo
 - Tipos são polimórficos se suas operações podem ser aplicadas a operandos de mais de um tipo



Formas de Polimorfismo

- O Polimorfismo pode ser obtido de várias formas nas Linguagens de Programação
 - não existe um modo único e sistemático de determinar o resultado de uma função em termos dos tipos de seus argumentos de entrada
 - potencialmente com um conjunto infinito de tipos de modo disciplinado

Polimorfismo Ad Hoc

- Coerção
 - mapeamento interno de tipos
 - equivalente a uma conversão automática de tipo
- Sobrecarga (overloading)
 - nome da função usado mais de uma vez com diferentes tipos de parâmetros
 - a função é implementada para os diferentes tipos de parâmetros
 - a informação do tipo seleciona a implementação

Polimorfismo Universal

- Paramétrico
 - uma função é codificada e trabalhará uniformemente num intervalo de tipos
 - chamadas de *funções genéricas*
- Inclusão
 - está relacionado a noção de subtipo
 - subtipo é uma instância de polimorfismo de inclusão
 - todo objeto do subtipo pode ser usado no contexto do supertipo

Acoplamento Dinâmico

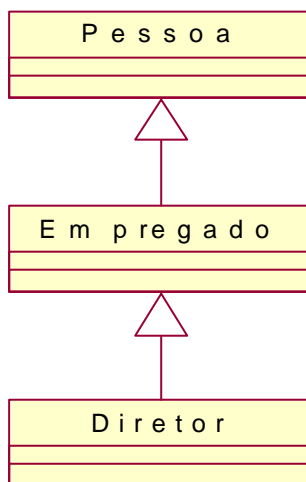
- Acoplamento (Binding)
 - uma associação entre um atributo e uma entidade
 - dinâmico: ocorre em tempo de execução
- Linguagens OO fazem verificação estática de tipos e acoplamento dinâmico (+/-)
 - este é usado para amarração do nome de uma operação à sua implementação
 - depende do tipo de objeto que recebe a mensagem

Exemplo

- João - pessoa
- Maria - empregado
- Joana - diretor
- empregado é subtipo de pessoa
- diretor é subtipo de empregado
- poderia ter violação de verificação de tipos

```
Pessoa *Joao = new Pessoa
Empregado *Maria = new
    Empregado
Diretor * Joana = new Diretor
Joao := Maria (i)
Joao := Joana (ii)
```

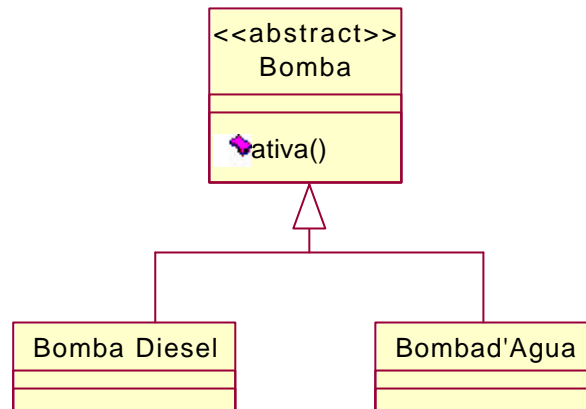
- João é atribuído a um objeto diferente do seu tipo estático
- As atribuições são válidas pois são subtipos de pessoa



- Uma mensagem enviada em tempo de execução é acoplada dinamicamente dependendo do tipo de objeto que recebe
- Todas as operações sobrecarregadas na classe derivada têm que proporcionar os mesmos serviços da superclasse

Classes Abstratas x Concretas

- Classe Abstrata
 - não possui instâncias diretas
 - classes descendentes tem instâncias diretas
 - pode definir o protocolo para uma operação, sem definir o código
- Classe Concreta
 - classe que pode ser instanciada
 - classes concretas tem que evidenciar a implementação de suas operações



Classe Abstrata

- Presença de operações abstratas
- Habilidade de criar instâncias
 - se uma classe possui operações abstratas então a classe não pode ser instanciada pois não pode responder a todas as mensagens
 - uma classe pode não ser preparada para instanciar diretamente e não possui operações abstratas

Delegação

- Mecanismo parecido com herança
- Herança aplica-se somente a classes
- Delegação opera diretamente entre objetos
 - objetos *delegadores* delegam seu comportamento a objetos relacionados (*delegados*)
 - relacionamento *delega-para* pode ser estabelecido dinamicamente
 - Herança é fixada quando a classe é criada
 - Delegação: as operações e atributos na classe delegadora não sobrecarregadas são herdadas pela classe delegada
- Compartilhamento de Comportamento
 - herança: baseado em classes
 - delegação: baseado em instâncias
- Delegação
 - facilita a mudança da implementação de operações em tempo de execução

Metaclasses

- Metainformação é um termo genérico
 - dado que descreve outro dado
 - classes são vistas como “fábricas” que criam e iniciam instâncias
- Metaclass é uma classe que descreve outra
 - ela é uma classe cujas instâncias são classes
 - uma metaclass guarda a metainformação de uma classe
 - classe neste caso seria um objeto da metaclass

- os métodos da metaclass
 - métodos da classe - podem ser usados para recuperar e atualizar os valores de atributos da classe
- noção de atributos de classe e métodos de classe
 - atributo de classe
 - vale para todos os objetos da classe

