Análise de Sistemas – AULA 05

BCC Noturno - EMA908915A

Prof. Rafael Oliveira
rpaes@ic.unesp.br

Universidade Estadual Paulista
“Júlio de Mesquita Filho”
UNESP

Rio Claro 2014 (Sem 2)

Análise Orientada a Objeto

- Roteiro da aula
  - Conceitos básicos
  - Exercícios
  - Aprofundamento de conceitos OO (prática)
  - (próxima aula) Modelagem OO: UML
    - Introdução a UML
    - Diagramas UML
    - Estudo de caso

Conceitos básicos

A orientação a objetos surgiu como uma abordagem de programação que procura explorar o nosso lado intuitivo. Os "átomos" da computação orientada a objetos (os próprios objetos), são análogos aos objetos existentes no mundo físico, o que produz um modelo de programação muito diferente da tradicional visão "funcional" e "procedimental" (Coleman, 1996).

Conceitos básicos

- A visão do modelo computacional **orientado a funções** poderia ser assim resumida:

  - Os "átomos" da computação são as funções;

  - As funções atuam sobre um único estado compartilhado;

  - Qualquer função pode atuar sobre qualquer parte do estado compartilhado.

Conceitos básicos

- Já no modelo computacional orientado a objetos:

  - A única maneira de se obter ou alterar o estado de um objeto é pelo envio de mensagens;

  - Cada objeto “é responsável” pela resposta a uma determinada mensagem;

  - Os objetos, quando compartilham uma única interface, são agrupados em classes.

Conceitos básicos

- Durante a execução do sistema, os objetos podem ser construídos ou destruídos, podem executar ações ou se tornar inacessíveis: estas características o tornam um modelo computacional dinâmico. Assim sendo, os “átomos” do processo de computação são os objetos:

  - Os objetos trocam mensagens entre si;

  - As mensagens resultam na ativação de métodos;

  - O emissor da mensagem não precisa saber como o objeto receptor organiza o seu estado interno;

  - Ao emissor da mensagem basta saber que o objeto receptor/servidor responde a certas mensagens de maneira bem definida.

Conceitos básicos

- Entretanto, o paradigma orientado a objetos deve, além do processo de programação, abranger também as demais fases do processo de desenvolvimento de sistemas computacionais.
- Um dos aspectos mais importantes e comuns entre os métodos de análise e projeto de sistemas orientados a objetos é viabilizar uma hierarquia com as especificações das classes de objetos que deverão ser implementadas para formar o software.

Conceitos básicos

- Um paradigma de programação, seja ele Estruturado ou Orientado a Objetos é a forma como a solução para um determinado problema é desenvolvida
- Por exemplo, em Orientação a Objetos os problemas são resolvidos pensando-se em interações entre diferentes objetos
- já no paradigma Estruturado procura-se resolver os problemas decompondo-os em funções e dados que somados formarão um programa

Conceitos básicos

- durante muito tempo a programação Estruturada foi o paradigma mais difundido porém
- a medida que os programas foram tornando-se mais complexos, surgiu a necessidade de resolver os problemas de uma maneira diferente
- em Orientação a Objetos os dados e as operações que serão realizadas sobre estes formam um conjunto único (objeto)
- em Orientação a Objetos a resolução de um problema é dada em termos de interações realizadas entre estes objetos

Conceitos básicos

- dizemos que um objeto encapsula a lógica de negócios
- concentrando a responsabilidade em pontos únicos do sistema
- viabilizando um maior reuso do código pela modularidade desta abordagem

Conceitos básicos

- Além da modularidade, que permite que um objeto seja utilizado em todo o sistema, diversos benefícios são decorrentes do uso desse paradigma
  - encapsulamento: detalhes de implementação ficam ocultos externamente ao objeto
  - reuso: uma vez criado, o objeto pode ser utilizado em outros projetos ou programas
  - manutenibilidade: manutenções e reparos de código podem ser feitas apenas em partes específicas do programa

Conceitos básicos

- Objetos são “coisas” que temos no mundo real e abstraímos no mundo virtual para que possamos manipulá-los na resolução de problemas
- Um objeto no mundo real sempre possui estado e comportamento
- um objeto possui características e ações que são pertinentes à sua natureza.

Conceitos básicos

unesp

Objeto	Estado	Comportamento
Pessoa	Nome, idade, RG	Falar, andar, cumprimentar
Cachorro	Nome, raça	Latir, correr
Conta bancária	Saldo, agência, número	Creditar, debitar
Carro	Cor, marca, modelo	Acelerar, frear, abastecer

Estado e **comportamento**, respectivamente, são transformados em **dados e procedimentos** quando programamos de forma estruturada e **atributos e métodos** quando utilizamos orientação a objetos.

Conceitos básicos

- Declarando uma classe com um método qualquer

```
public class GradeBook{
public void displayMessage()
{ System.out.println
(“Linguagens comerciais - 2012”); }
}
```

Conceitos básicos

- Instanciando um objeto de um outra classe

```
public class ExecuteGradeBook{
public static void main (String[] args)
{
GradeBook myGradeBook = new GradeBook();
myGradeBook.displayMessage();
}
}
```

Conceitos básicos

- Declaração de classe com um método com parâmetro

```
public class GradeBook{
public void displayMessage(String nomeCurso)
{ System.out.println
(“O nome do curso é”+nomeCurso); }
}
```

Conceitos básicos

- objetos são oriundos (instâncias) das classes
- uma classe é uma especificação para um determinado tipo de objeto
- para que o objeto seja de determinada classe ele obrigatoriamente, terá que respeitar a especificação

Conceitos básicos

- Por exemplo, vamos especificar que todo Objeto do tipo documento deve possuir, ao menos, foto, código, nome e data de nascimento.

Agora vamos **criar** um documento para o Alfredo e um para a Juliana:

Documento	Documento 1	Documento 2
Foto:	Img1.png	Img4.png
Código:	123456	789012
Nome:	Alfredo	Juliana
Data de nascimento:	20/05/1990	30/09/1987

Conceitos básicos

- nesta tabela a coluna Documento define a classe (especificação)
- as colunas Documento 1 e Documento 2 são os objetos
- desta forma, cada documento particular terá um valor diferente para os atributos definidos na especificação
- a compreensão desta diferença, entre classes e objetos, é parte fundamental do paradigma orientado a objetos.

Conceitos básicos

- Em termos de código ...

Esta especificação em Java poderia ser da seguinte forma:

```
class Documento {

        //Estado
        String foto; //Nome do arquivo de imagem
        String nome; //Nome da pessoa
        Integer codigo; //Codigo deste documento
        String dataNascimento; //Data de nascimento

}
```

Conceitos básicos

- EXERCÍCIO:
  - Crie a classe Documento
  - Após, crie uma outra classe e instancie dois objetos com os dados da tabela ilustrada nos slides anteriores
  - verifique o que ocorre quando você atribui a instância de um objeto a outro. Imprima os valores na tela

Conceitos básicos

Métodos realizam as ações que são realizadas sobre os atributos. Um conjunto de métodos define o comportamento de um objeto.

- Imagine um sistema de uma corrida de carros
- nosso objetivo inicial é criar uma especificação (classe) para os carros
- nosso carro de corrida terá seu estado definido pelo conjunto de atributos número de identificação, velocidade atual e velocidade máxima
- o comportamento será definido pelo conjunto de métodos acelerar, frear, ligar e desligar

Conceitos básicos

- vamos criar a nossa classe apenas com a definição do estado, inicialmente:

```
class CarroCorrida {

        //Estado
        Integer numeroIdentificacao;
        Double velocidadeAtual;
        Double velocidadeMaxima;

        //Comportamento...
}
```

Conceitos básicos

- adicionar os comportamentos ligar e desligar
- observe que os métodos criados tem retorno do tipo void

```
void ligar()
{
        System.out.println("VRUUUMmmmmmmmmm");
}


    void desligar()
{
        System.out.println("MMMmmmm.....");
}
```

Conceitos básicos

- adicionar iremos implementar o método acelerar
- regra de negócio: adiciona 10 unidades na velocidade

```
void acelerar()
{
        velocidadeAtual += 10;
        if(velocidadeAtual > velocidadeMaxima)
        {
                velocidadeAtual = velocidadeMaxima;
        }
}
```

Conceitos básicos

- agora vamos implementar o método frear

```
void frear(Integer intensidadeFreada)
{
        if(intensidadeFreada > 100)
        {
                intensidadeFreada = 100;
        }else if(intensidadeFreada < 0)
        {
                intensidadeFreada = 0;
        }

        velocidadeAtual -= intensidadeFreada*0.25;

        if(velocidadeAtual < 0)
        {
                velocidadeAtual = 0.0;
        }
}
```

Conceitos básicos

- Regra de negócio: O método frear recebe um parâmetro que significa a intensidade com que o pedal de freio foi acionado.
- Essa intensidade pode ser um valor entre 0 e 100, a velocidade após a freada é o resultado da intensidade da freada multiplicada pelo fator 0.25 tudo isto diminuído da velocidade atual, caso o resultado desta operação seja menor do que 0 (zero) então o valor final será zero.

Conceitos básicos

- Mas, observando novamente o nosso modelo de carro de corrida percebemos que faltam algumas características importantes
- no nosso caso o piloto com seus atributos são fundamentais para uma corrida de automóveis.
- estes atributos são nome, idade e habilidade.
- obviamente, cada carro de corrida terá de ter um atributo piloto como sendo seu condutor

Conceitos básicos

```
class Piloto {

        String nome;
        Integer habilidade;
        Integer idade;

}

class CarroCorrida {

        //Estado
        Integer numeroIdentificacao;
        Double velocidadeAtual = 0.0;
        Double velocidadeMaxima = 100.0;
        Piloto piloto;

        //...
```

Conceitos básicos

- Agora que temos um Piloto, podemos alterar nosso método acelerar

```
void acelerar()
{
        velocidadeAtual += 10 + piloto.habilidade*0.1;
        if(velocidadeAtual > velocidadeMaxima)
        {
                velocidadeAtual = velocidadeMaxima;
        }
}
```

Conceitos básicos

Temos o nosso carro de corrida com piloto. Agora podemos criar o nosso programa que simula uma corrida de carros entre duas equipes. a equipe “Velocidade” e a equipe “Trapaceiros” serão as equipes hipotéticas:

- baixe a classe corrida hipotetica (moodle)
- carregue a classe em sua IDE
- implemente as classes Piloto e CarroCorrida

Conceitos básicos

- EXERCÍCIOS:
  - **1** Implemente uma classe Pessoa com os seguintes atributos: Nome, idade e CPF.
  - **2** Utilizando a classe implementada no exercício anterior crie um programa que instancie 2 pessoas com todos os atributos e imprima os valores.
  - **3** Implemente uma classe que represente uma sala de aula, esta sala pode ter o máximo de 10 alunos, se extrapolar este limite deve ser impressa uma mensagem avisando que o número máximo de alunos foi atingido.

Conceitos básicos

- **4** Implemente uma classe de Endereço com os seguintes atributos: Estado, Cidade, Bairro, Rua, CEP e telefone.
- **5** Adicione a classe Pessoa desenvolvida no exercício **1** um atributo de endereço utilizando a classe desenvolvida no exercício **4**.
- **6** Implemente um método construtor para a classe Pessoa de forma que uma instância desta classe seja criada apenas se possuir nome, idade e cpf. (pergunte ao professor o que é construtor caso não saiba)

Conceitos básicos

- **7** Um usuário deseja uma calculadora que efetue as quatro operações básicas. As expressões permitidas são binárias envolvendo apenas dois números, por exemplo, **2 + 3.5 ou 3 * 3.2**. Identifique os objetos, seus métodos e atributos.

Conceitos básicos

- **8** Seguindo a abordagem de orientação a objetos, identificar no enunciado abaixo os objetos e usuários do sistema. Liste os nomes dos objetos, seus atributos e os usuários do sistema. Faça o mesmo para a análise e projeto estruturados identificando as grandes funções e suas decomposições.

UMA LOCADORA de veículos necessita de um sistema para facilitar o atendimento a seus clientes. Os carros são classificados por tipo: popular, luxo e utilitário. As informações que interessam à locadora sobre cada um dos veículos são: placa do carro, tipo e valor diário do aluguel.

Os funcionários da locadora são responsáveis pelo cadastro dos clientes e dos veículos. Eles também fazem as locações e encerram as mesmas. Há clientes especiais e comuns. Os especiais têm direito a uma taxa de desconto e um valor de quilometragem extra nas suas locações. Um cliente é identificado pelo nome, número do cartão de crédito e data de expiração.

Conceitos básicos

- **9**

Para atender as necessidades de informação de uma biblioteca universitária foi proposto um sistema que deve atender as seguintes características:

- O cadastro dos usuários da biblioteca com endereço completo. Os usuários podem ser classificados em três grupos: Professores, Alunos e Funcionários.
- O cadastro das obras da biblioteca, que podem ser classificadas em: Livros científicos, periódicos científicos, periódicos informativos, periódicos diversos, entretenimento, etc.
- A língua em que se encontra o exemplar da obra.

- A mídia onde se encontra o exemplar da obra.
- Os autores da obra com o controle da nacionalidade do autor.
- As editoras dos exemplares com o ano de edição de cada exemplar.

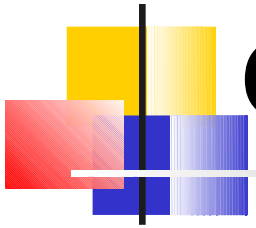
Identifique os possíveis objetos com seus respectivos atributos e métodos.

Conceitos básicos

- **10** Você deve criar um sistema o gerenciamento de um zoológico. Identifique classes e objetos para a implementação do sistema.

Conceitos básicos

- Após essa visão introdutória sobre OO, a seguir aprofundaremos e revisaremos conceitos técnicos associados a Orientação a Objetos.

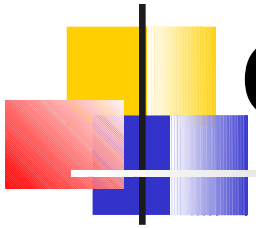


Conceito de OO (cont.)

- Princípios da Orientação a Objetos:
  - Abstração;
  - Encapsulamento;
  - Classe e Objeto;
  - Herança;
  - Escala (Todo-Parte);
  - Associação;
  - Comunicação com Mensagens;
  - Polimorfismo;

Conceito de OO (cont.)

- **Abstração** é a habilidade de ignorar os aspectos de um assunto não relevantes para o propósito em questão, tornando possível uma possível **concentração maior nos assuntos principais**. A abstração consiste na seleção que um desenvolvedor faz de alguns aspectos suprimindo outros.

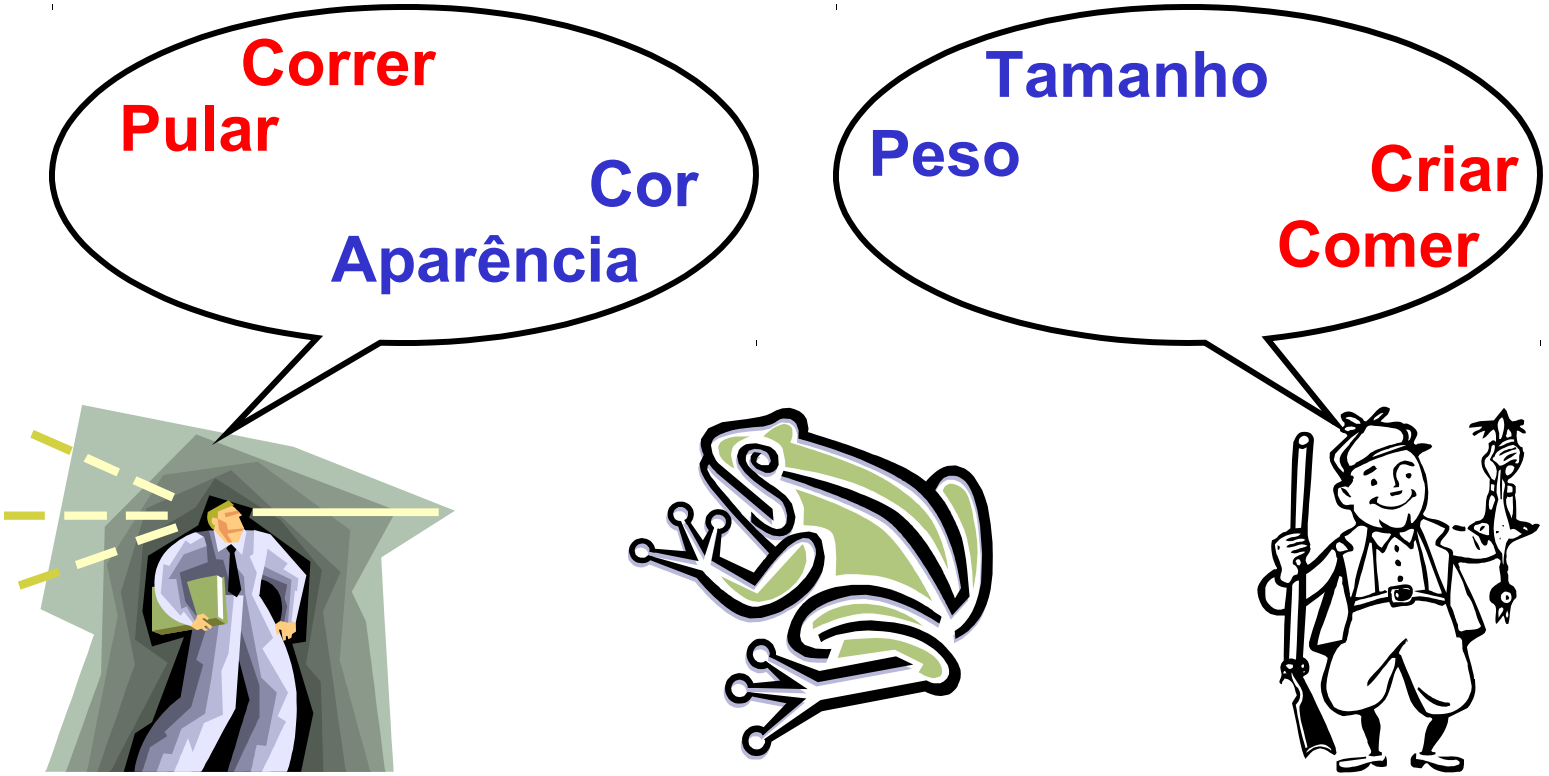


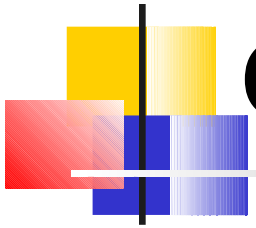
Conceito de OO (cont.)

- Pessoas, lugares, coisas e conceitos do mundo real são normalmente complexos, **quando queremos diminuir a complexidade selecionamos parte do que estamos analisando**, em vez de tentarmos compreender o todo.
- Ao aplicar a abstração de objetos o desenvolvedor define atributos e comportamentos, que manipulam exclusivamente estes atributos. A seguir apresenta-se uma breve descrição de cada um deles:
  - Atributo;
  - Comportamento.

Conceito de OO (cont.)

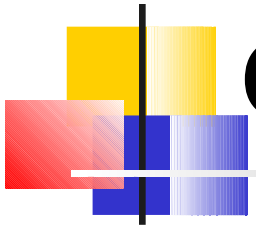
- Abstração (**diferentes visões**):





Conceito de OO (cont.)

- **Encapsulamento** é o princípio usado no desenvolvimento de uma **estrutura global** de programa, estabelecendo que cada componente do programa deve conter uma única decisão de projeto. A **interface** para cada modulo é definida de forma a **revelar o menos possível** sobre seu funcionamento interno.

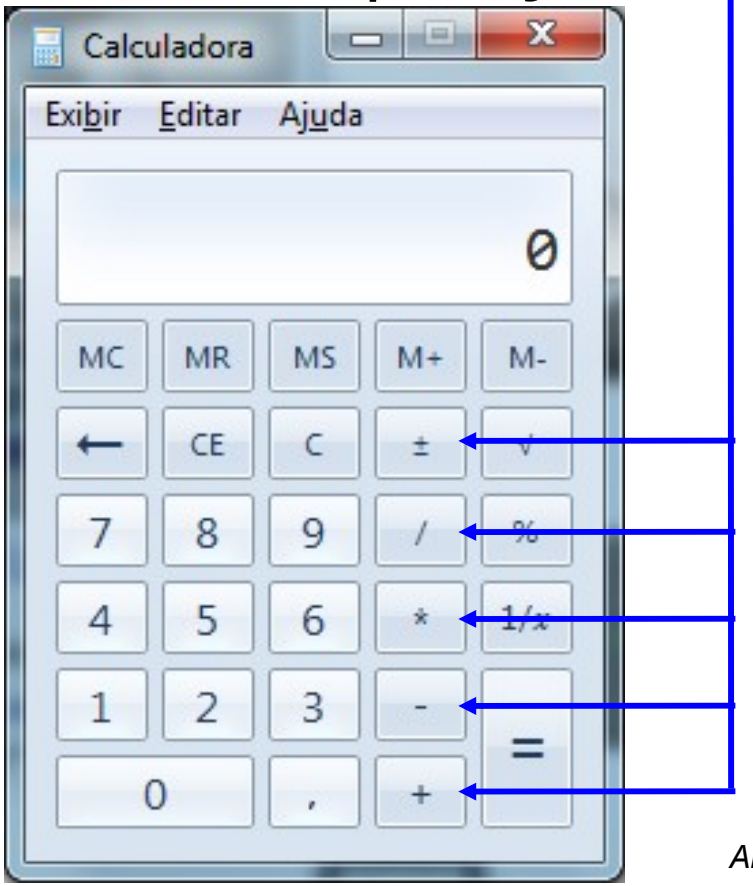


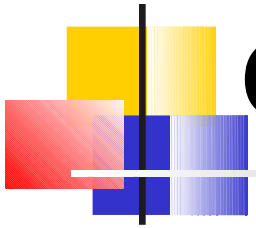
Conceito de OO (cont.)

- O Objetivo do encapsulamento é **restringir o escopo ou a visibilidade da informação** (ocultamento de informações) para obter melhor:
  - Legibilidade;
  - Manutenibilidade; e,
  - Principalmente, reusabilidade no desenvolvimento de um novo sistema.

Conceito de OO (cont.)

- Encapsulamento de operações:





Conceito de OO (cont.)

- **Classe e Objeto:** das idéias de abstração e encapsulamento tem-se o **Tipo Abstrato de Dados (TAD)**, que possui uma interface para ocultar detalhes de implementação, possibilitando que se tenham diferentes especificações, em diferentes tempos, sem afetar as aplicações que utilizam o TAD.

Conceito de OO (cont.)

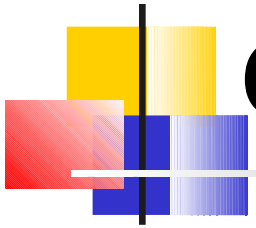
- Classe e Objeto:

Classe: `Pessoa`

Objeto: `Pedro`

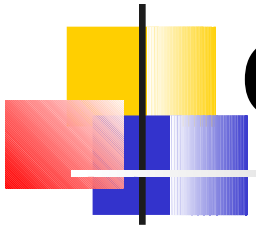
Objeto: `Fernanda`

Objeto: `José`



Conceito de OO (cont.)

- **Herança** é o mecanismo para **expressar a similaridade entre Classes iguais** a outras que já foram definidas;
- Este princípio permite representar membros comuns, serviços (métodos) e características (atributos), uma só vez;
- Assim como especializar estes membros em casos específicos.

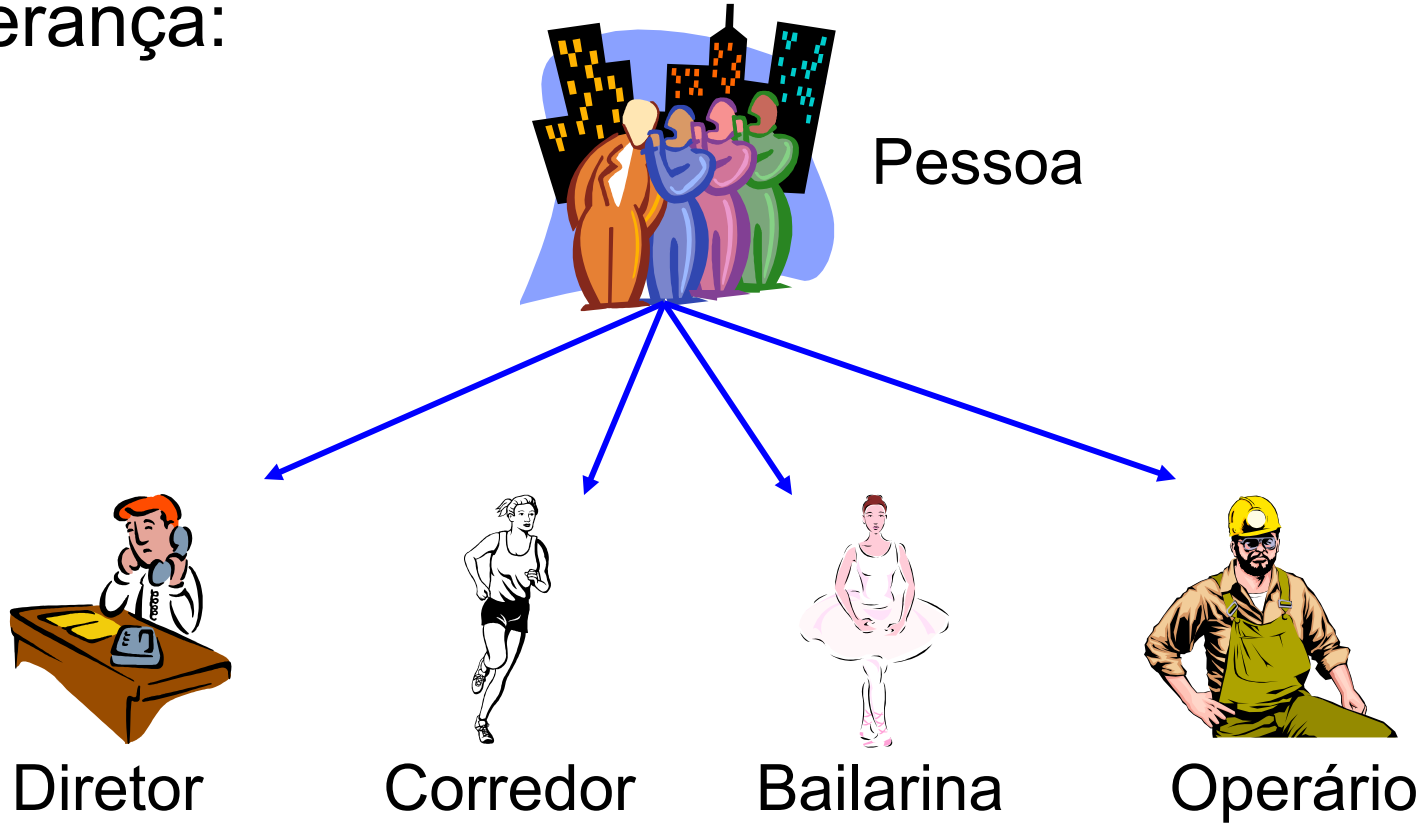


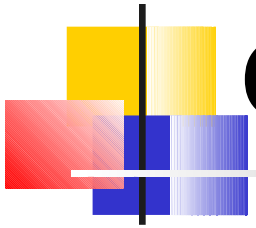
Conceito de OO (cont.)

- A herança permite, portanto, a **reutilização de especificações comuns**, logo no início das atividades de análise;
- A herança define uma relação entre classes do tipo: **é-um(a)**, onde **uma classe compartilha a estrutura e o comportamento** definidos em uma ou mais classes.

Conceito de OO (cont.)

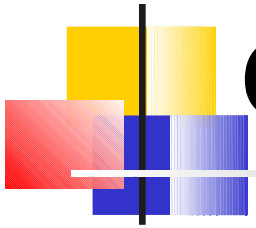
- Herança:





Conceito de OO (cont.)

- **Escala** é o princípio que permite ao desenvolvedor considerar algo muito grande através do enfoque **Todo-Parte**, que se trata de um dos serviços básicos e naturais de organizações dos seres humanos que orienta o desenvolvedor através de um modelo extenso (todo e partes).



Conceito de OO (cont.)

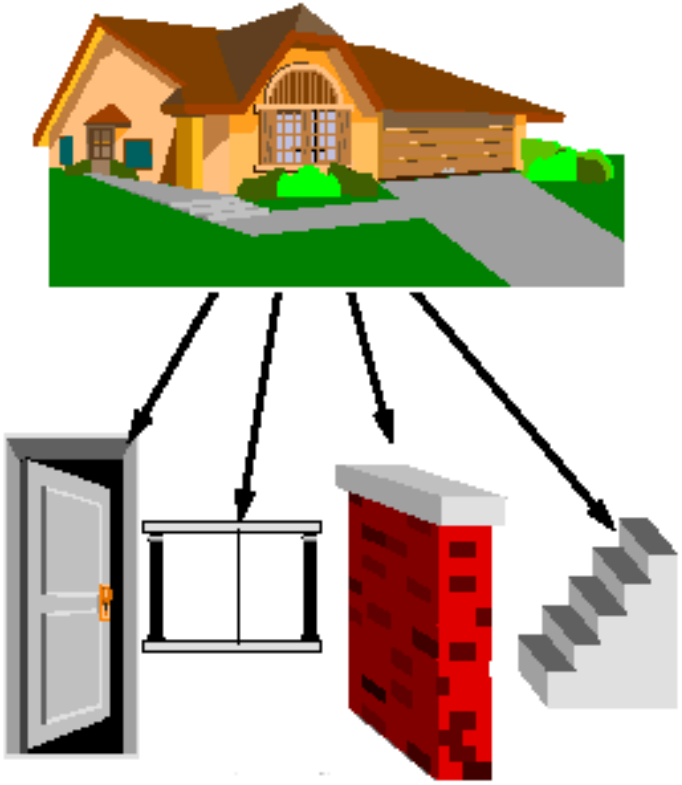
- Todo-Parte, também é conhecido como **Agregação / Agregação por Composição**, é um mecanismo pelo qual permite a construção de uma classe agregada a partir de outras classes componentes. Usa-se dizer que um objeto da classe agregada (**Todo**) tem objetos das classes componentes (**Partes**).

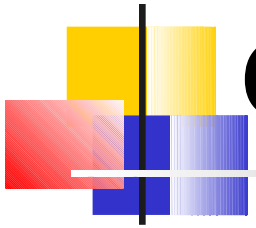
Conceito de OO (cont.)

- Todo-Parte (Agregação / Composição):

Todo

Partes





Conceito de OO (cont.)

- **Associação** vem do relacionamento entre as entidades do mundo real. É usada para agrupar certos objetos que ocorrem em algum ponto no tempo ou sob circunstâncias similares;
- É uma união ou conexão de ideias. Na AOO, a associação é **modelada através de uma conexão de ocorrências;**
- Uma conexão de ocorrência é um relacionamento que um objeto precisa ter com outro(s) objeto(s), para cumprir suas responsabilidades.

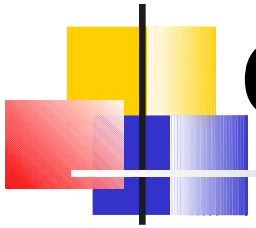
Conceito de OO (cont.)

- Associação:

Faz

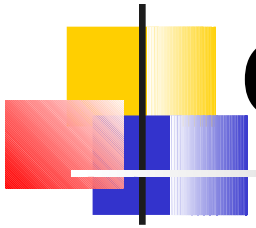
Cliente

Pedido



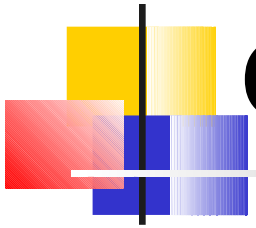
Conceito de OO (cont.)

- Relacionamentos:
  - Cardinalidade:
    - 1-1
    - 1-* | *-1
    - *-*
  - Navegabilidade:
    - Unidirecional (→)
    - Bidirecional (←→)



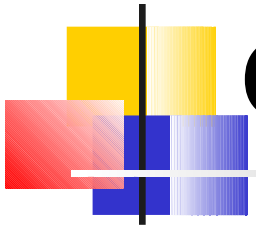
Conceito de OO (cont.)

- Conceitos complementares:
  - **Coesão:** está relacionado ao tratamento das informações/comportamentos de uma classe. Ela deve "cuidar" das suas tarefas e não pedir para que outras classes façam;
  - **Acoplamento:** está relacionado ao relacionamento entre as classes num sistema. Quanto mais mensagens trocadas entre elas maior é seu acoplamento.



Conceito de OO (cont.)

- **Comunicação com mensagens**, representa a comunicação entre os objetos do sistema através da conexões de mensagens, que modela a dependência de processamento de um objeto, indicando quais serviços (métodos) ele precisa para cumprir suas responsabilidades.

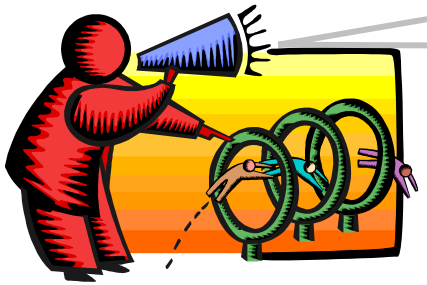


Conceito de OO (cont.)

- As conexões de mensagens existem somente em função dos serviços (métodos) e fazem mapeamento:
  - De um objeto para outro;
  - De um objeto para uma classe (criar objetos);
  - De uma classe para outra (criação de objetos dentro de outros objetos).

Conceito de OO (cont.)

- Comunicação com mensagens:



Vamos executar a próxima obra.

Chefe

Por onde começamos Sr.

Mecânico

Ordem recebida Sr.

Operário

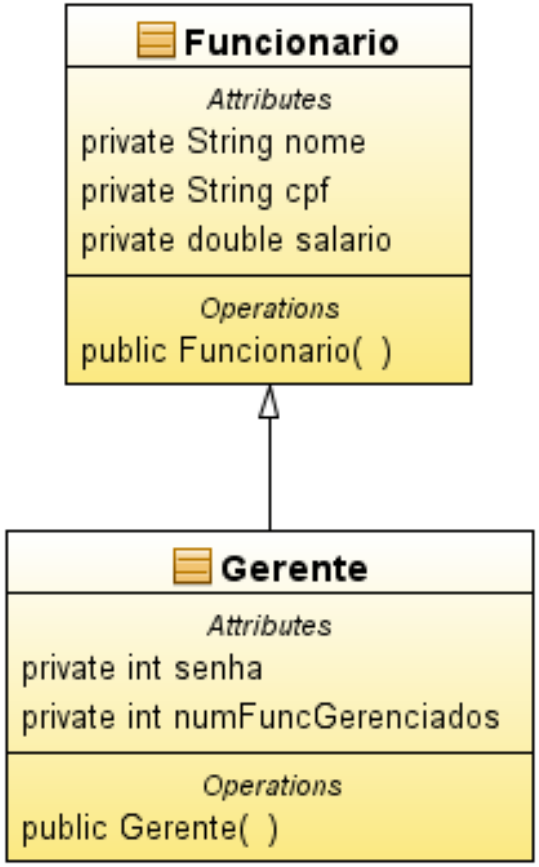
Conceito de OO (cont.)

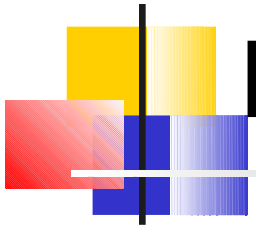
- **Polimorfismo**
  - Poli ~ muitos
  - Morfismo ~ formas
- Várias formas.

- **Polimorfismo:** permite o “programar geral” em vez do “programar específico”, isto é, permite escrever programas que compartilham a mesma superclasse em uma hierarquia de classes como se fossem objetos da superclasse.

Conceito de OO (cont.)

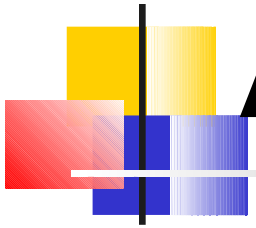
- Vamos definir um método que atribui uma bonificação aos funcionários. Questões:
  - Como isso será implementado, pois essa bonificação ocorre em escala, funcionários, gerentes, diretores, etc?
  - Como o sistema irá reconhecer o método?
  - Ele será implementado no Funcionário?
  - Cada um terá uma bonificação diferenciada!





Principais Métodos OO

- Booch;
- OMT (Object Modeling Technique);
- OOSE(Object-Oriented Software Enginnering);
- Shalaer/Mellor;
- Coad/Yourdon Orientado a Objetos;
- Martin/Odell;
- Martin/Odell;
- Wirfs-Brock;
- Embley/Kurtz
- **UML – Unified Modeling Language;**
- Catalysis.



A Linguagem UML

- A UML é um **conjunto de técnicas** que apoiam a elaboração de projetos de sistemas orientados a objetos. Essas técnicas podem ser aplicadas na visualização, na especificação, na construção e na documentação de sistemas. Para isso, recebe contribuição de vários métodos OO.
- Veremos na próxima aula!!!!!