07 - INTRODUÇÃO OO

14/09/16

Francisco Barretto – francisco.barretto@udf.edu.br

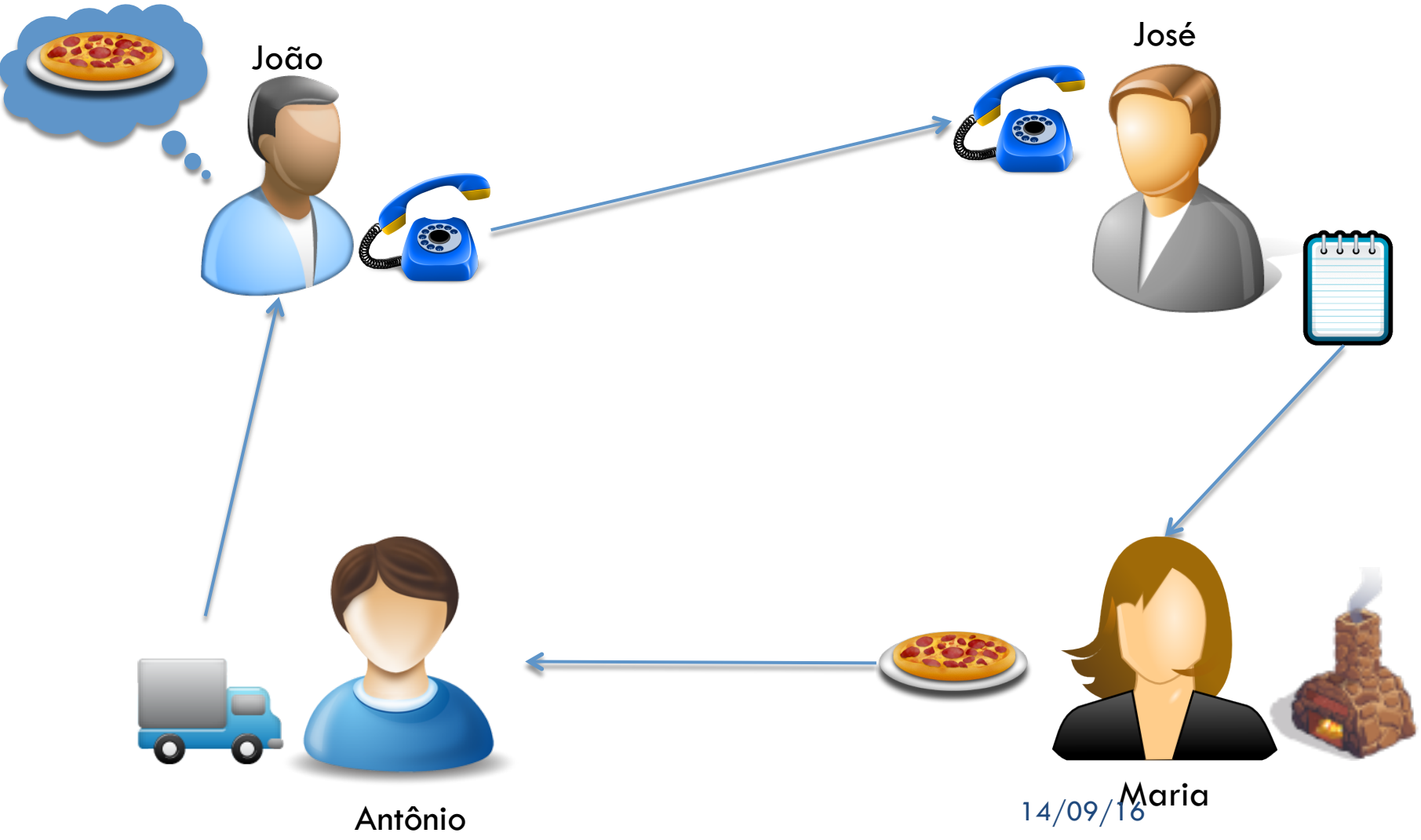
1. O paradigma da orientação a objetos

- O termo “paradigma da orientação a objetos” é baseado na chamada “analogia biológica” (Alan Kay):
  - Sistema de software funciona como um ser vivo;
  - Células interagindo;
  - Envio de mensagens;
  - Célula como unidade autônoma;

1. O paradigma da orientação a objetos

- Princípios da Orientação a Objetos:
  - Qualquer coisa é um objeto;
  - Objetos realizam tarefas através da requisição de serviços a outros objetos;
  - Cada objeto pertence a uma determinada *classe*. Uma classe agrupa objetos similares;
  - A classe é um repositório para comportamento associado ao objeto;
  - Classes são organizadas de forma hierárquica.

1. O paradigma da orientação a objetos



1. O paradigma da orientação a objetos

- Qual a relação entre orientação a objetos e modelagem de sistemas?
- Antes, utilizávamos os paradigma estruturado;
- Nesse paradigma, os elementos são dados e processos.
- Processos agem sobre dados para que um objetivo seja alcançado.

1. O paradigma da orientação a objetos

- Qual a relação entre orientação a objetos e modelagem de sistemas?
- Por outro lado, no paradigma OO, há um elemento, o objeto.
- Ele é uma unidade autônoma que contém seus próprios dados que são manipulados pelos processos definidos para o próprio objeto;
- Ele interage com outros objetos para alcançar um objetivo.

1. O paradigma da orientação a objetos

- O paradigma OO é o que utilizamos no cotidiano para a resolução de problemas.
- Uma pessoa atende a mensagens (requisições) para realizar um serviço;
- Essa mesma pessoa envia mensagens a outras para que estas realizem outros serviços...

Pra quem estava perdido no FB...

"O paradigma da orientação a objetos visualiza um sistema de software como uma coleção de agentes interconectados chamados objetos. Cada objeto é responsável por realizar tarefas específicas. É através da interação entre objetos que uma tarefa computacional é realizada." (Bezerra, 2004)

Resumindo...

"Um sistema de software orientado a objetos consiste de objetos em colaboração com o objetivo de realizar as funcionalidades desse sistema. Cada objeto é responsável por tarefas específicas. É através da cooperação entre objetos que a computação do sistema se desenvolve." (Bezerra, 2004)

2. Classes e Objetos

- Mundo real: coisas
  - Livro, copo, panela, carro, vendedor, ...
- Em Orientação a Objetos (OO): objetos

- Naturalmente agrupamos objetos;
- Categorizamos itens;
- Separamos características comuns;
- Em OO essas são as classes.

2. Classes e Objetos

- A Classe é uma descrição comum dos atributos e serviços comuns a um grupo de objetos;
- Classe é um molde a partir do qual construimos os objetos;
- Um objeto é uma instância de uma classe.

HORSE VARIATIONS


Orlov Trotter


Timor


Dale


Lipizzaner


Tarpan


Arab


Fjord Pony


Normandy Cob


Pinto


Falabella


Belgian Heavy Draught


Shetland

2. Classes e Objetos

- Uma classe é uma abstração das características de um grupo de coisas;
- Complexidade do mundo real;
- Identificação das características relevantes;
- Abstração destas características;
- Representação destas características relevantes na forma de classes;
- Simplificação do modelo.

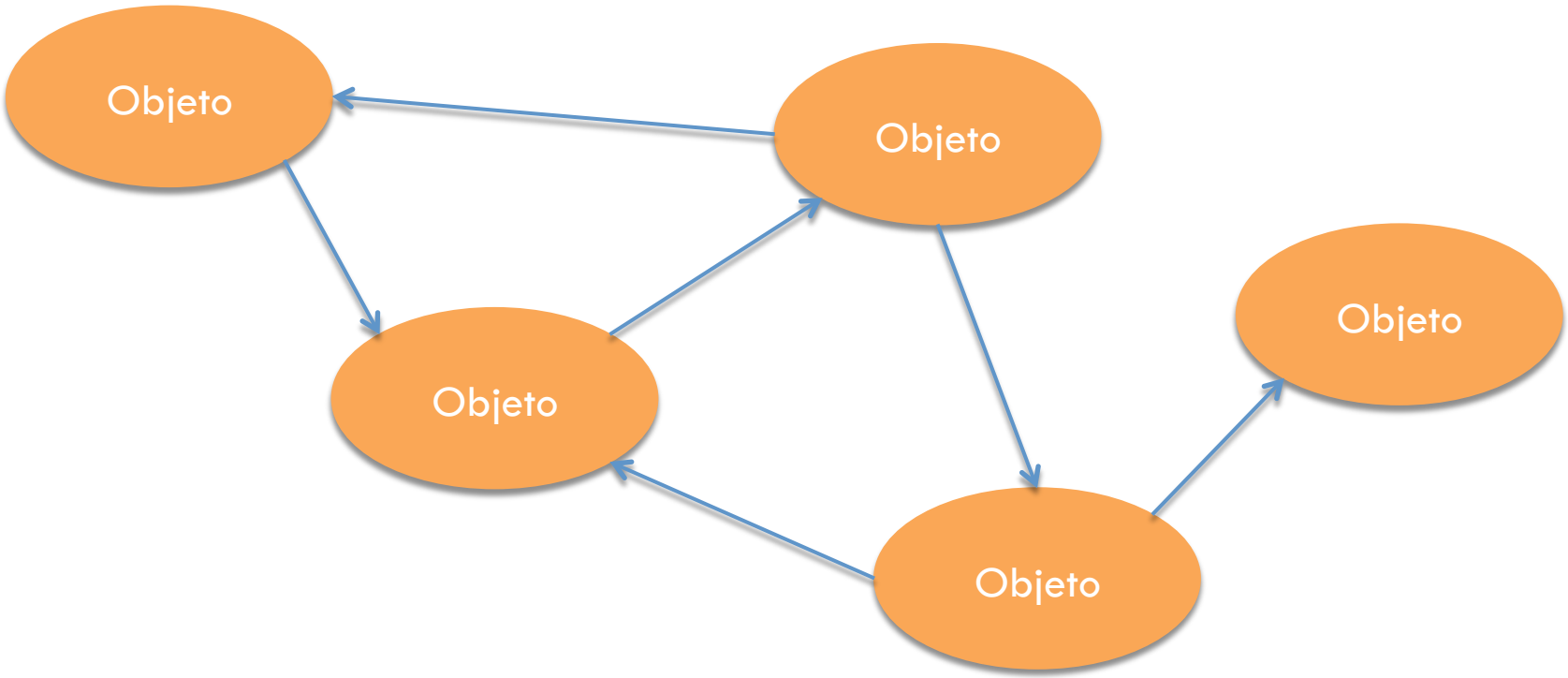
3. Mensagens

- Há ordem na execução de operações pelos objetos;
- Objeto recebe um estímulo (mensagem);
- Objeto executa uma operação;

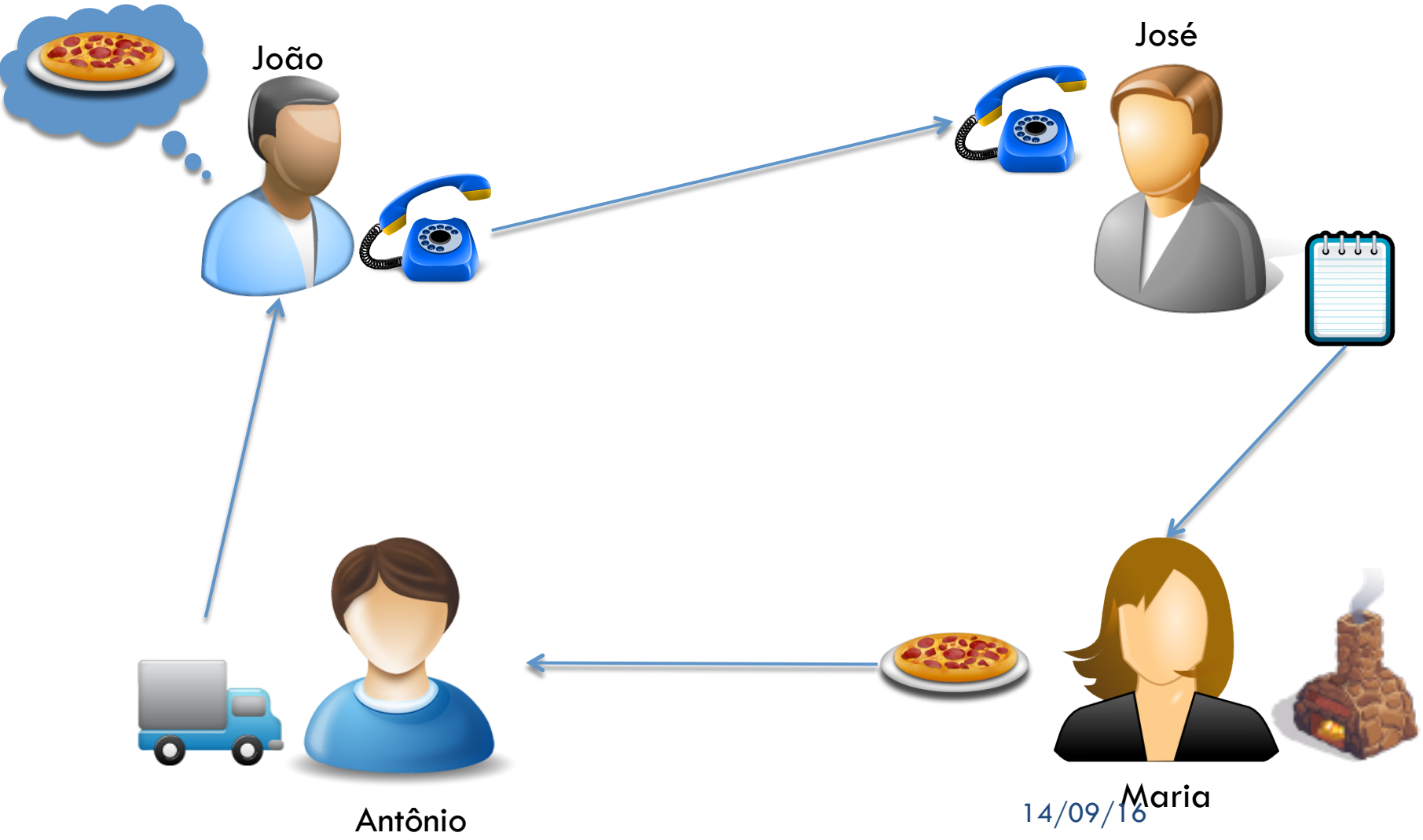
- Objeto enquanto entidade ativa reage aos estímulos enviados.

- Estímulo = Mensagem

3. Mensagens



Parece familiar?



Interlúdio: Pilares da OO

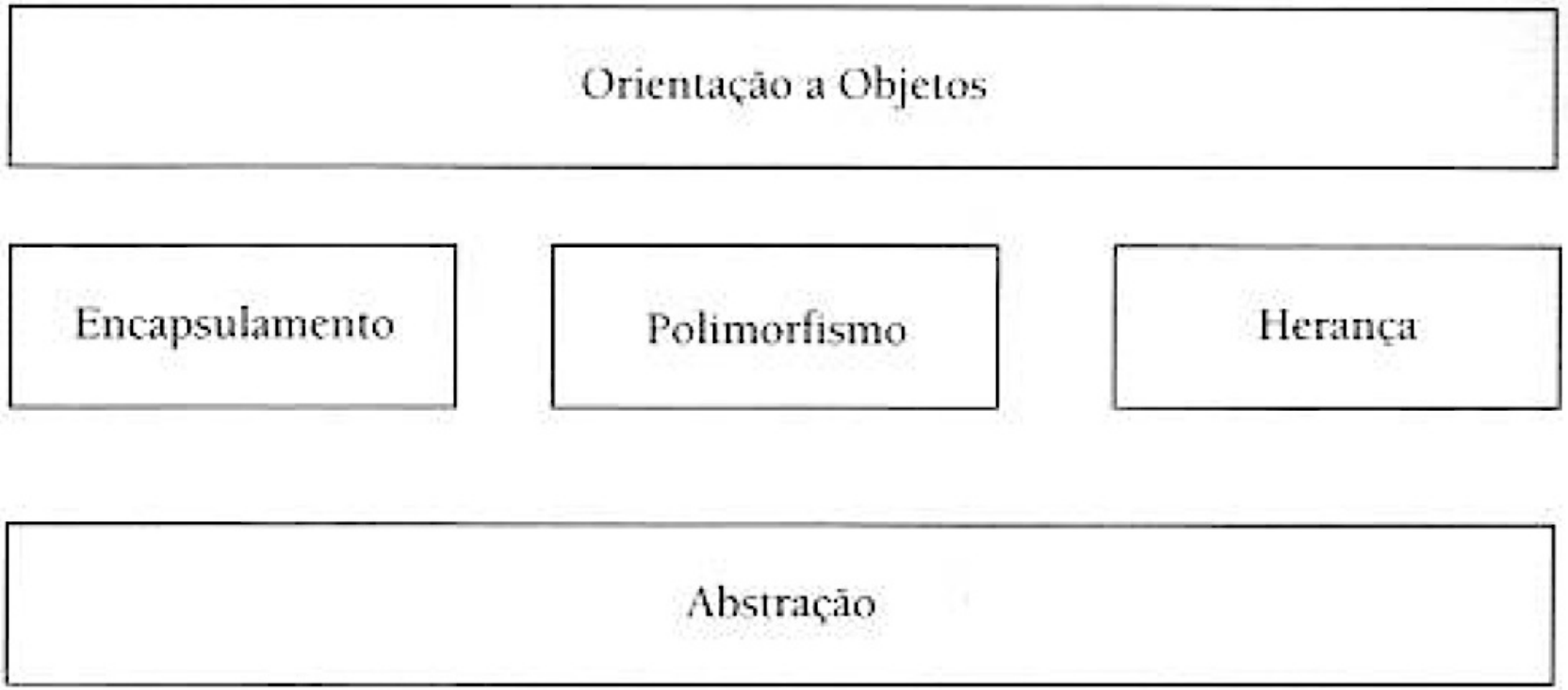


Figura 1-2 Princípios da orientação a objetos como aplicações do Princípio da Abstração.

4. Princípio da Abstração

- Uma abstração é qualquer modelo que inclui os aspectos mais importantes, essenciais de alguma coisa, ao mesmo tempo em que ignora os detalhes menos importantes.
- Gerenciamento da complexidade;
- Concentração nas características relevantes;
- Dependente da perspectiva;

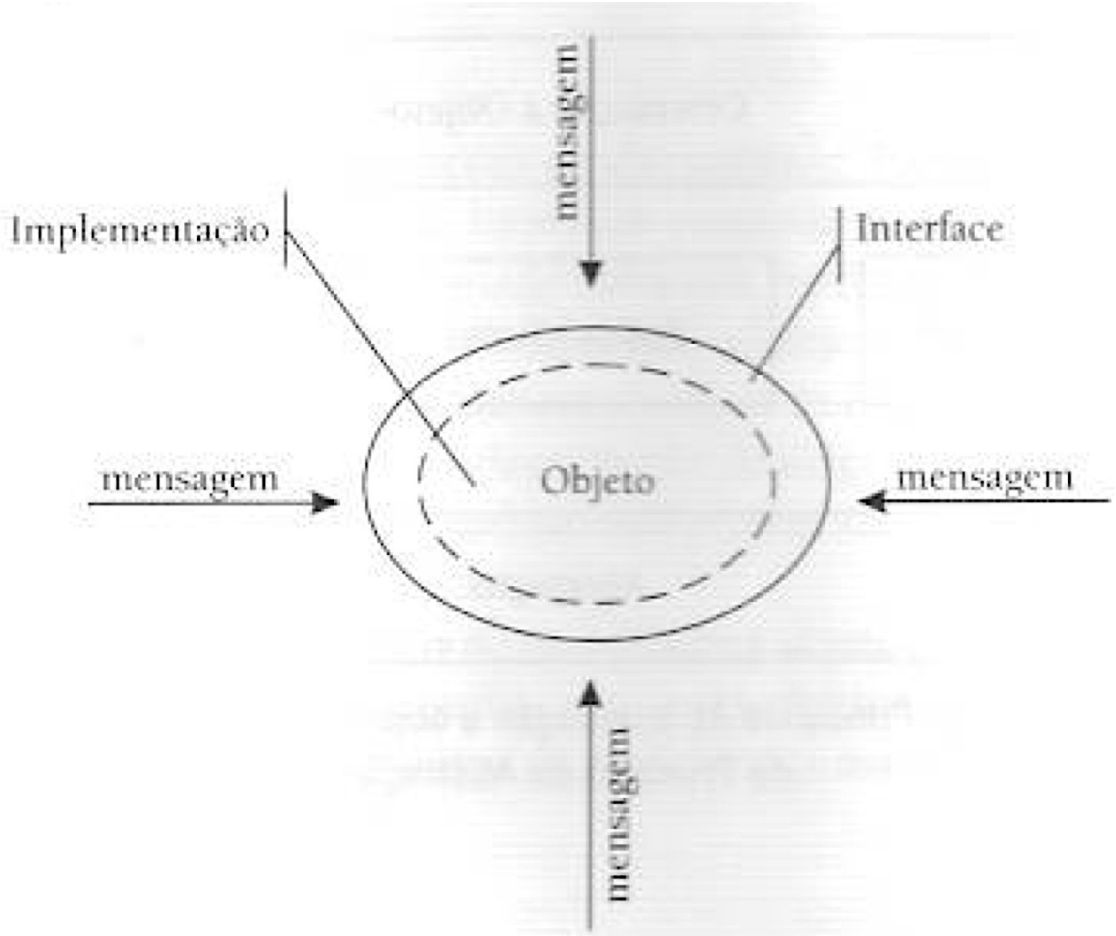
4.1 Encapsulamento

- Os objetos possuem comportamento;
- Comportamento são as operações realizadas por um objeto;
- E os comportamentos internos?
  - Digestão, por exemplo. Deve ser acessado externamente?
- Comportamento também deve expressar o que deve ser acessível externamente o que é interno;
- Encapsulamento restringe o acesso ao comportamento interno dos objetos.

4.1 Encapsulamento

- A classe descreve o comportamento de um objeto;
- Um objeto possui uma interface;
- A interface conhece o que o objeto faz, sem descrever como faz;
- A interface de um objeto define os serviços que ele realiza, ou seja, que mensagens ele recebe;
- A interface pode ter várias formas de implementação;
- A implementação da interface não importa para quem solicita a execução da tarefa.

4.1 Encapsulamento



4.2 Polimorfismo

- O termo polimorfismo é originário do grego e significa "muitas formas" (poli = muitas, morphos = formas).
- Polimorfismo é o princípio pelo qual duas ou mais classes derivadas de uma mesma superclasse podem invocar métodos que têm a mesma identificação (assinatura) mas comportamentos distintos, especializados para cada classe derivada, usando para tanto uma referência a um objeto do tipo da superclasse.

4.2 Polimorfismo

- Em outras palavras, polimorfismo é a capacidade de um objeto comportar-se como se fosse derivado de mais de uma classe;
- Esse comportamento polimórfico se adequa à quem chama:
  - João pode responder quando demanda-se um comportamento humano;
  - João também pode responder quando demanda-se um comportamento mamífero;

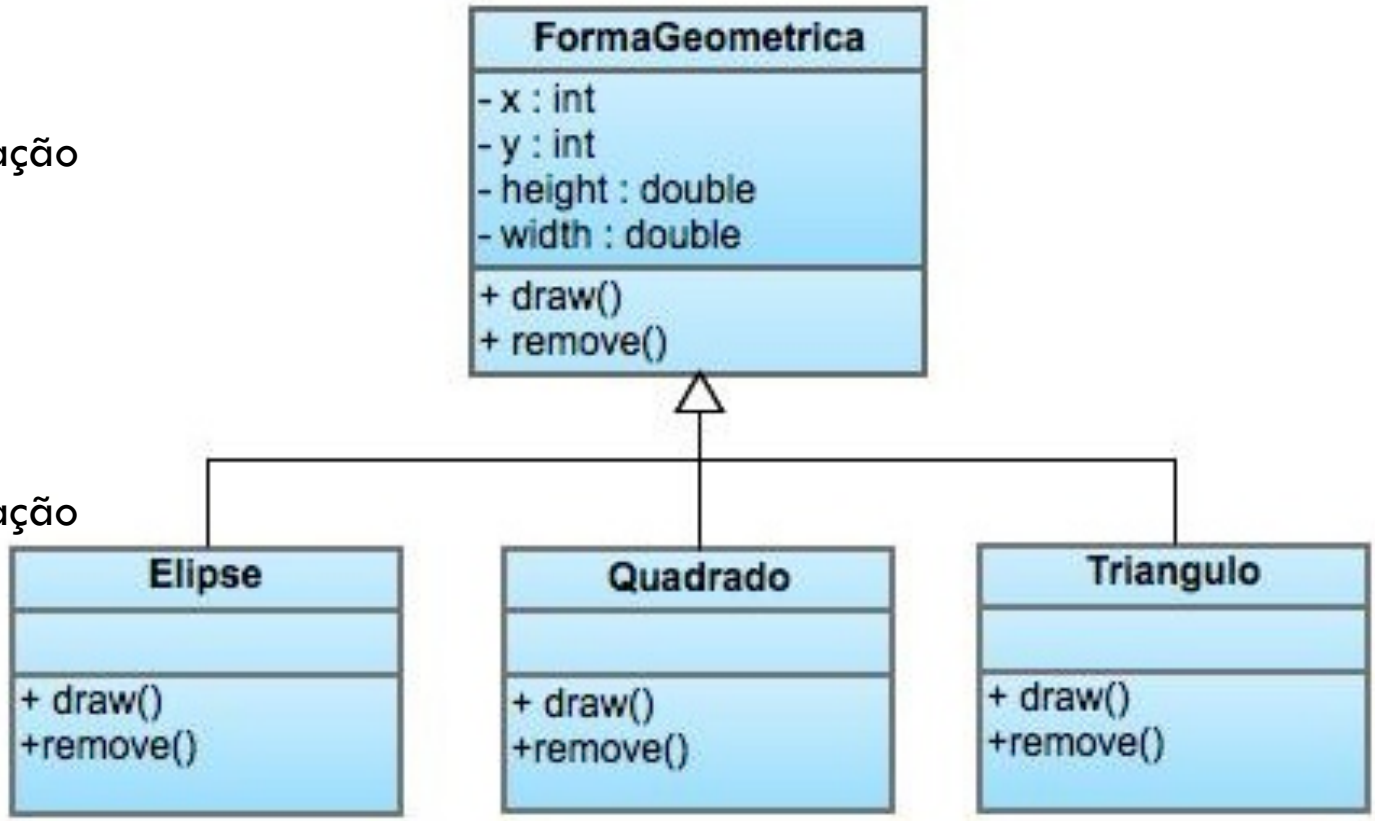
4.3 Herança

- Herança é um mecanismo que permite que características comuns a diversas classes sejam fatoradas em uma classe base, ou superclasse.
- A partir de uma classe base, outras classes podem ser especificadas.
- Cada classe derivada ou subclasse apresenta as características (estrutura e métodos) da classe base e acrescenta a elas o que for definido de particularidade para ela.

4.3 Herança

Maior Abstração

Menor Abstração



4.3 Herança

- Há várias formas de relacionamentos em herança:
  - Extensão.
  - Especificação.
  - Combinação de extensão e especificação.

Pra quem ficou caçando Pokemon...

- Um objeto é uma construção de software que encapsula estado e comportamento. Os objetos permitem que você modele seu software em termos reais e abstração.

Vantagens e Objetivos

- A POO busca produzir sistemas de software que tenham as seguintes características:

1. **Natural**: Programar utilizando a terminologia de seu problema em particular. Não há a necessidade de se aprofundar nos detalhes técnicos enquanto se projeta o programa. Mas é importante "pensar com os pés no chão", saber que a funcionalidade que se determina é possível de se implementar .

2. **Confiável**: Os objetos isolam o conhecimento e a responsabilidade de onde pertencem.

3. **Reutilizável** : Devemos reinventar a roda? Criar classes bem feitas é uma tarefa difícil que exige concentração e atenção à abstração. Mas classes bem feitas podem ser reutilizadas. Uma vez que o problema esteja resolvido, voce deve reutilizar a solução.

4. **Manutenível**: Corrigir o erro em um único lugar.

5. **Extensível** : O software não é estático. Ele deve crescer e mudar com o passar do tempo, para permanecer útil. A POO apresenta ao programador vários recursos para estender o código, como herança, polimorfismo, sobreposição, delegação e uma variedade de padrões de projetos.

6. **Oportunos**: O ciclo de vida do projeto de software moderno é freqüentemente medido em semanas. A POO diminui o tempo de desenvolvimento, fornecendo software confiável, reutilizável e facilmente extensível. Quando se divide um programa em vários objetos, o desenvolvimento de cada parte pode ocorrer em paralelo. Vários desenvolvedores podem trabalhar em classes paralelas.

Desvantagens e Armadilhas

- Inicialmente em Orientação a Objetos existem quatro armadilhas a serem evitadas.
- Pensar na POO simplesmente como uma linguagem
  - Voce não programa OO so porque utiliza uma linguagem que implementa a OO. POO é um estado da mente que exige que voce veja seus problemas como um grupo de objetos e use encapsulamento, herança e polimorfismo corretamente.
- Medo da reutilização
  - Aprender a reutilizar sem culpa freqüentemente é uma das lições mais difíceis de aprender e há basicamente duas dificuldades que passamos aqui:

  a. Os programadores gostam de criar;
  b. E aqueles que não confiam no software que não escrevem.
- Pensar em OO como uma solução para tudo
  - Existem ocasiões que voce não deve utilizar a OO. Há a necessidade do uso do bom senso para a escolha da ferramenta correta. O sucesso aparece somente com planejamento, projeto e codificação cuidadosos.
- Programação egoísta
  - Lembre-se dos outros desenvolvedores quando programar. Faça interfaces limpas e inteligíveis. O mais importante: escreva documentação. Documente suposições, parâmetros de métodos, documente o máximo que você puder.

Exercício 1:

- Defina o que é:
  - Classe;
  - Comportamento;
  - Abstração;
  - Objeto

Exercício 2:

- Explique e relacione os termos objeto, classe, herança e mensagem.
- Dê exemplos de cada um desses conceitos.

Exercício 3

- Identifique as classes, atributos e métodos necessários para modelar e implementar:
  - Uma conta corrente que possui um número, um saldo, um status que informa se ela é especial ou não, um limite e um conjunto de movimentações.
  - Uma movimentação que possui uma descrição, um valor e uma informação se ela é uma movimentação de crédito ou débito.
  - Um banco que armazene um conjunto de contas e forneça métodos que permitam que sejam feitos criações de conta, exclusão de contas, saques (uma conta corrente só pode fazer saques desde que o valor não exceda o limite de saque-limite + saldo negativo), depósitos, emissão de saldo e extrato e transferência entre contas.

Referência

BEZERRA, Eduardo 1972-. Princípios de análise e projeto de sistemas com UML: um guia prático para modelagem de sistemas orientados a objetos através da linguagem de modelagem unificada. 2. ed. totalmente rev. e atual. Rio de Janeiro: Elsevier, 2004