
Métodos e Atributos
Herança
Classes e Objetos
Classes Abstratas
Polimorfismo

Programação Orientada a Objetos

As técnicas orientadas a objeto permitem que o software seja construído de objetos que tenham um comportamento específico. Os próprios objetos podem ser construídos a partir de outros, os quais, por sua vez, podem ainda ser construídos de outros.

A análise de sistemas no mundo orientado a objeto é feita analisando-se os objetos e os eventos que interagem com esses objetos. O projeto de software é feito reusando-se classes de objetos existentes e quando necessário, construindo-se novas classes.

Técnicas orientadas a objeto podem ser usadas para simplificar o projeto de sistemas complexos. O sistema pode ser visualizado como uma coleção de objetos, estando cada um dos objetos em um determinado estado. Os objetos são construídos a partir de outros objetos.

A análise e o projeto orientados a objeto modelam o mundo em termos de objetos que tem propriedades e comportamentos e eventos que disparam operações que mudam o estado dos objetos. Os objetos interagem com outros objetos.

A modelagem e o projeto orientados a objeto são os paradigmas que devem integrar todas as ferramentas e técnicas poderosas para a criação de software. Estratégia de desenvolvimento baseada no conceito de que o sistema deve ser construído a partir de componentes reutilizáveis, chamados de objetos.

Conceitos

Entre as idéias fundamentais básicas para a tecnologia orientada a objeto incluem-se:

- Classes;
- Objetos;
- Métodos;
- Herança;
- Encapsulamento;

Cada conceito é uma idéia ou um entendimento pessoal que temos do nosso mundo. Os conceitos que adquirimos nos permitem dar sentido sobre as coisas do nosso mundo. Essas coisas às quais nossos conceitos se aplicam são denominados objetos.

Objeto

- Um objeto pode ser real ou abstrato.
- Os objetos possuem informações (contém dados) e desempenham ações (possuem funcionalidade).
- Qualquer coisa à qual um conceito ou tipo de objeto se aplica – uma instância de um conceito ou tipo de objeto.
- Um objeto é uma instância de uma classe.

Exemplo:

- **Uma fatura;**
- **Uma organização;**
- **Um vôo de avião;**
- **Uma pessoa;**
- **Um lugar.**

Na análise e no projeto OO, estamos interessados no comportamento do objeto. As operações são codificadas como métodos. A representação de software OO do objeto é, dessa forma, uma coleção de tipos de dados e métodos. No software OO: Um objeto é qualquer coisa, real ou abstrata, a respeito da qual armazenamos dados e os métodos que os manipulam.

Exemplo: um tipo de objeto poderia ser Fatura e um objeto poderia ser a fatura nº. 51.783.

OBS: O termo objeto, porém, é diferente do termo entidade. A entidade preocupa-se apenas com os dados enquanto o objeto preocupa-se tanto com os dados como com os métodos com os quais os dados são manipulados.

Métodos

- Os métodos especificam a maneira pela qual os dados de um objeto são manipulados.
- Uma especificação dos passos pelos quais uma operação deve ser executada.
- Ele é um script de implementação de uma operação.
- Diferentes métodos podem ser usados para executar a mesma operação.
- Os métodos de um tipo de objeto referenciam somente as estruturas de dados desse tipo de objeto.
- A ação que um objeto ou uma classe podem desempenhar.
- Os métodos são similares às funções e procedures do universo da programação estruturada.

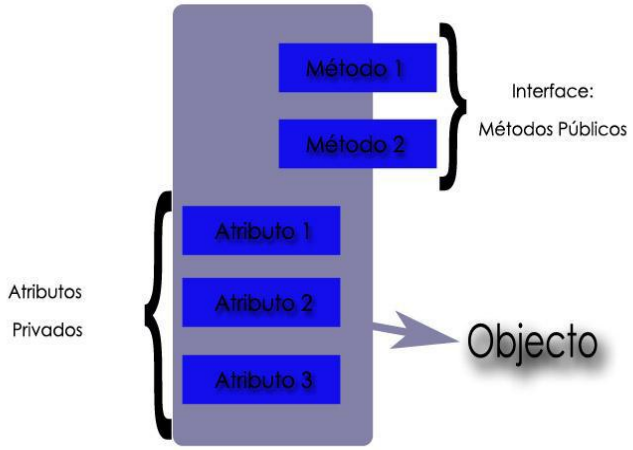
Um objeto é, dessa forma, um conjunto das suas propriedades representadas pelos tipos de dados e seu comportamento representado pelos métodos.

Encapsulamento

O ato de empacotar ao mesmo tempo dados e objetos é denominado encapsulamento. O objeto esconde seus dados de outros objetos e permite que os dados sejam acessados por intermédio de seus próprios métodos. Isso é chamado de ocultação de informações (*information hiding*).

- O encapsulamento protege os dados do objeto do uso arbitrário e não-intencional.
- O encapsulamento é o resultado (ou ato) de ocultar do utilizador (outro programador) os detalhes da implementação de um objeto.
- O encapsulamento é importante porque separa a maneira como um objeto se comporta da maneira como ele é implementado.
- A definição de como implementar os conhecimentos ou ações de uma classe, sem informar como isto é feito.

Exemplo:



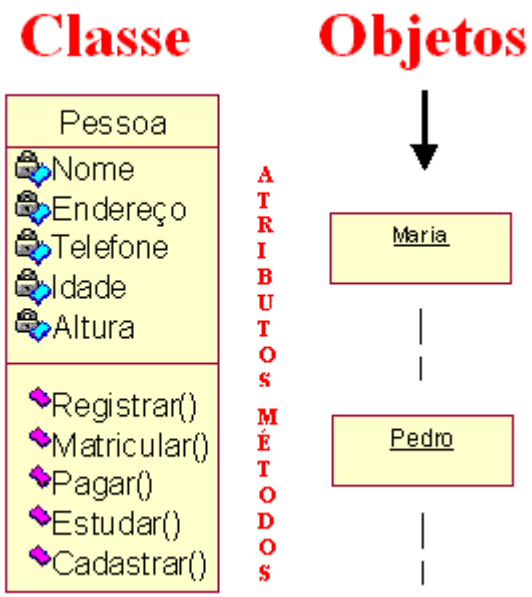
Cada objeto encapsula uma estrutura de dados e métodos. Uma estrutura de dados encontra-se no centro de um objeto. O objeto é manipulado por métodos que implementam as operações permitidas. A estrutura de dados pode ser usada somente com os métodos. Essa restrição ao acesso é denominada encapsulamento, que protege os dados contra adulteração. Os dados do objeto não podem ser usados, exceto com esses métodos.

Classe

O termo classe refere-se à implementação de software de um tipo de objeto. Um tipo de objeto especifica uma família de objetos sem estipular como o tipo e o objeto são implementados. Os tipos de objetos são especificados durante a análise OO. Os tipos de objetos são implementados como módulos enquanto na linguagem orientada a objeto, os tipos de objetos são classificados como classes.

- Uma classe é uma implementação de um tipo de objeto.
- **Uma classe específica a estrutura de dados e os métodos operacionais permissíveis que se aplicam a cada um de seus objetos.**
- Uma classe pode ter sua própria estrutura de dados e métodos, bem como herda-lá de sua superclasse.

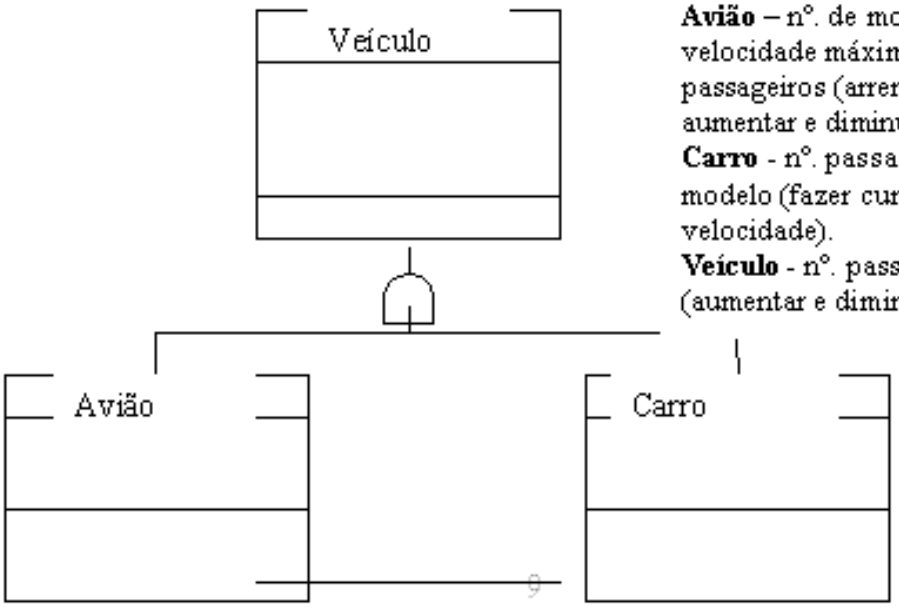
Exemplo: Classe - Objeto - Método - Atributo.



Classes Abstratas e Concretas

A classe abstrata é uma classe que não possui objetos instanciados a partir dela, enquanto a classe concreta possui objetos instanciados (*criados*) a partir dela.

Exemplo: No mundo real, por exemplo, existem automóveis e aviões, mas nada que seja simplesmente um veiculo (*em outras palavras, se não for um carro ou avião, não é de nosso interesse*). Isso significa que podemos instanciar objetos como carros e aviões, mas nunca iremos criar objetos veículos. As classes abstratas são criadas quando necessitamos de uma classe que implemente recursos comuns a duas ou mais classes.



Avião – nº. de motores, autonomia, velocidade máxima, altitude máxima, nº. passageiros (arremeter, aterrissar, glissar, aumentar e diminuir a velocidade).
Carro - nº. passageiros, autonomia, marca, modelo (fazer curvas, aumentar e diminuir a velocidade).
Veículo - nº. passageiros, autonomia (aumentar e diminuir a velocidade).

Herança

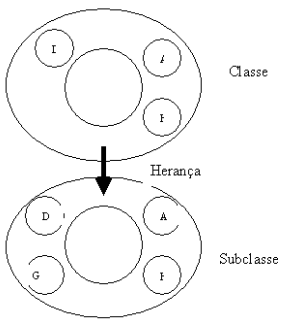
É comum haver similaridades entre diferentes classes. Frequentemente, duas ou mais classes irão compartilhar os mesmos atributos e/ou métodos. Como nenhum de nós deseja reescrever várias vezes o mesmo código, seria interessante se algum mecanismo pudesse tirar proveito dessas similaridades. A herança é esse mecanismo. Por intermédio da herança, é possível modelar relacionamentos do tipo **"é"** ou **"é semelhante"**, o que nos permite reutilizar rotinas e dados já existentes.

Uma subclasse herda as propriedades de sua classe-mãe; uma subclasse herda as propriedades das subclasses e assim por diante. Uma subclasse pode herdar a estrutura de dados e os métodos, ou alguns dos métodos, de sua superclasse. Ela também tem métodos e às vezes, tipos de dados próprios.

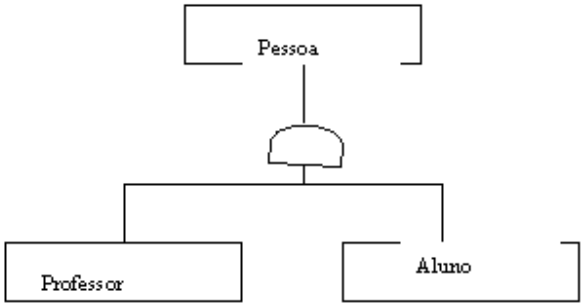
Subclasse – uma classe que é um subtipo de uma ou mais classes (*denominadas superclasses*). Como tal, ela herda todas as características de suas superclasses. Em outras palavras, todas as características de uma classe são reusáveis por suas subclasses. Se a classe B herda de A, então dizemos que B é uma subclasse de A.

Superclasse – Uma classe que é um supertipo de uma ou mais classes (*tb chamadas de subclasses*). Como tal, ela é uma classe a partir da qual todas as suas características são herdadas por suas subclasses. Em outras palavras, todas as características de uma superclasse são reusáveis por aquelas classes que são seus subtipos. Se a classe B herda de A, então dizemos que A é uma superclasse de B.

Exemplo 1: A figura abaixo mostra uma classe e uma subclasse. A subclasse tem os mesmos métodos de sua superclasse, mas tem também o método G. Ás vezes, uma classe herda propriedades de mais de uma superclasse. Isso é denominado herança múltipla.



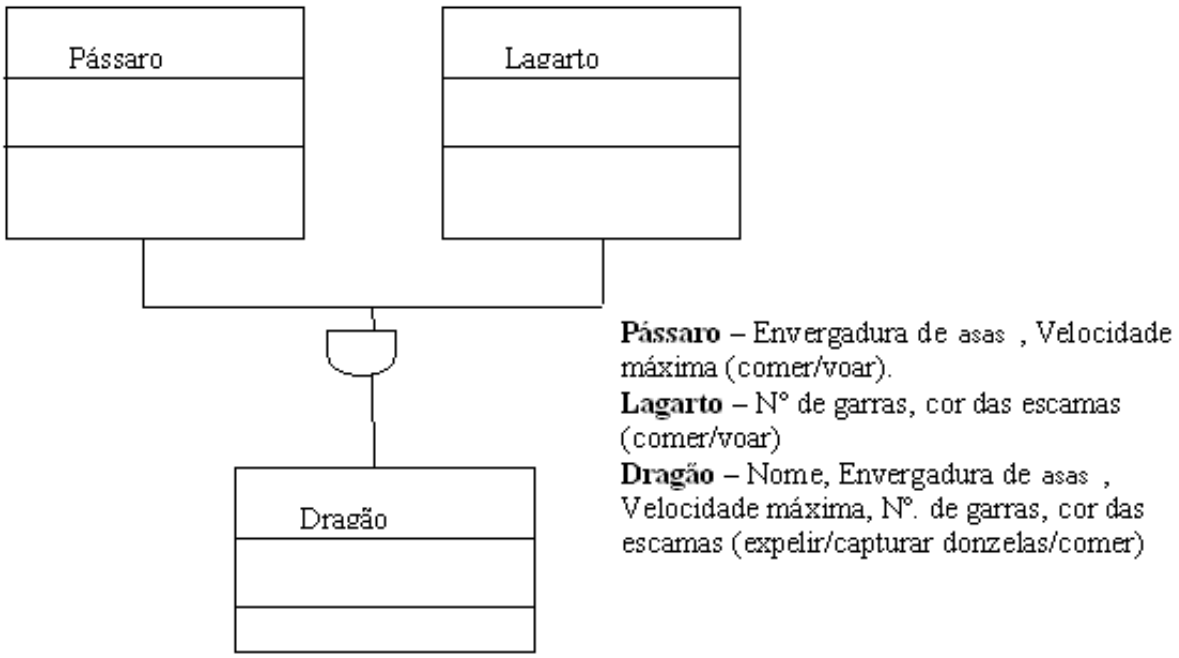
Exemplo 2: Pessoa – professor – aluno



OBS: As subclasses deverão sempre ficar abaixo da superclasse e o semicírculo deverá sempre apontar da subclasse para a superclasse

Herança Simples e Múltipla

Quando uma classe herda características somente de uma outra classe, dizemos que esta é uma herança simples. Quando uma classe herda de duas ou mais classes, temos um caso de herança múltipla. Em qualquer circunstância, o fato que você deverá lembrar é o seguinte: a subclasse herda todos os atributos e métodos das superclasses.



Hierarquias de Generalização

Um conjunto de classes relacionadas por meio da herança. Uma boa maneira de organizarmos os conhecimentos é organiza-lo em hierarquias do geral para o mais específico. Por exemplo, a figura ao lado descreve uma hierarquia com o conhecimento do tipo de objeto Pessoa no topo. Isso significa que Pessoa é um tipo de objeto mais geral do que empregado e estudante. Isso significa que empregado e estudante são subtipos de pessoa, ou, inversamente, que pessoa é um supertipo de empregado e estudante.