Análise e Modelagem de Sistemas

Fundamentos da orientação a objetos

Você sabia que seu material didático é interativo e multimídia? Isso significa que você pode interagir com o conteúdo de diversas formas, a qualquer hora e lugar. Na versão impressa, porém, alguns conteúdos interativos ficam desabilitados. Por essa razão, fique atento: sempre que possível, opte pela versão digital. Bons estudos!

Nesta webaula vamos conhecer os fundamentos do paradigma de orientação a objeto e os conceitos fundamentais de orientação a objetos.

Paradigma

Na engenharia de software, consideramos um paradigma como um modelo que já foi testado e segue alguns princípios para a resolução de um problema computacional. Há uma grande vantagem em seguir um modelo, pois facilita o desenvolvimento e a compreensão da solução encontrada.

> [...] um paradigma de programação é um padrão de resolução de problemas que se relaciona a um determinado gênero de programas e linguagens.
> — Tucker e Noolan (2010, p. 3).

Historicamente, a busca por padrões foi um processo evolutivo na construção de programas e sistemas.

Crise do software

Período entre 1968 e 1969 – Busca por um padrão de desenvolvimento de software, um paradigma.

Utilização de uma estrutura adequada para construção de algoritmos, com o objetivo de melhorar os padrões e processos de programação e facilitar o entendimento pelos programadores quando eles realizassem uma manutenção (alteração).

Padrão estruturado

1970 – Paradigma estruturado passou a ser utilizado pelos desenvolvedores

O padrão estruturado era adequado às linguagens existentes.

Orientação a objetos

Com o surgimento de ambientes gráficos; o crescimento dos sistemas, que tornaram-se cada vez maiores e mais complexos; a crescente necessidade de integração entre sistemas; a solicitação, dia após dia, de melhorias para serem incrementadas nas aplicações existentes. Diante desses fatores, o padrão estruturado se mostrou ineficiente para algumas aplicações e assim, surgiu o paradigma orientado a objetos, que se tornou muito utilizado a partir de 1997, com a linguagem UML (*Unified Modeling Language*).

Com o paradigma orientado a objeto surgiu não só um novo padrão para o desenvolvimento de software, mas também uma nova forma de pensar como modelar os problemas do mundo real.

Esse conceito também foi estendido para a linguagem de programação, com a linguagem Smalltak 80 e posteriormente outras duas importantes linguagens de programação orientada a objetos, a Eiffel e a Java.

Conceitos básicos POO (Programação Orientada a Objetos)

Dessa forma, surgiram outros importantes conceitos básicos de orientação a objetos. Entre os conceitos e princípios básicos de POO:

Abstração

A modelagem do objeto inicia pela abstração, que é quando reconhecemos os objetos. Suponha que você ouviu o termo cadeira: você pensa na ideia de como é uma cadeira, e isso é uma abstração.

Classe

É a representação da abstração do objeto com suas características e comportamentos.

- **Atributos:** denominações técnicas para as características.
- **Métodos:** ações ou comportamentos.

Exemplo: classe Cadeira, atributos: a classe Cadeira tem pés, rodinhas, encosto, assento e cor, métodos: ela move o encosto, eleva o assento e anda.

Objeto

É a materialização de forma única de uma classe, e a esta materialização damos o nome de instância da classe. Portanto, um objeto é uma instância de uma classe. Os objetos são únicos e distinguem-se pelos valores dos seus atributos e ações (comportamentos), apesar de pertencerem a uma mesma classe.

Saiba Mais

Herança

Aplicamos o conceito de herança quando elaboramos uma classe que herda as características e operações de outra classe (classe: Cadeira; herda atributos: encosto, assento e pés).

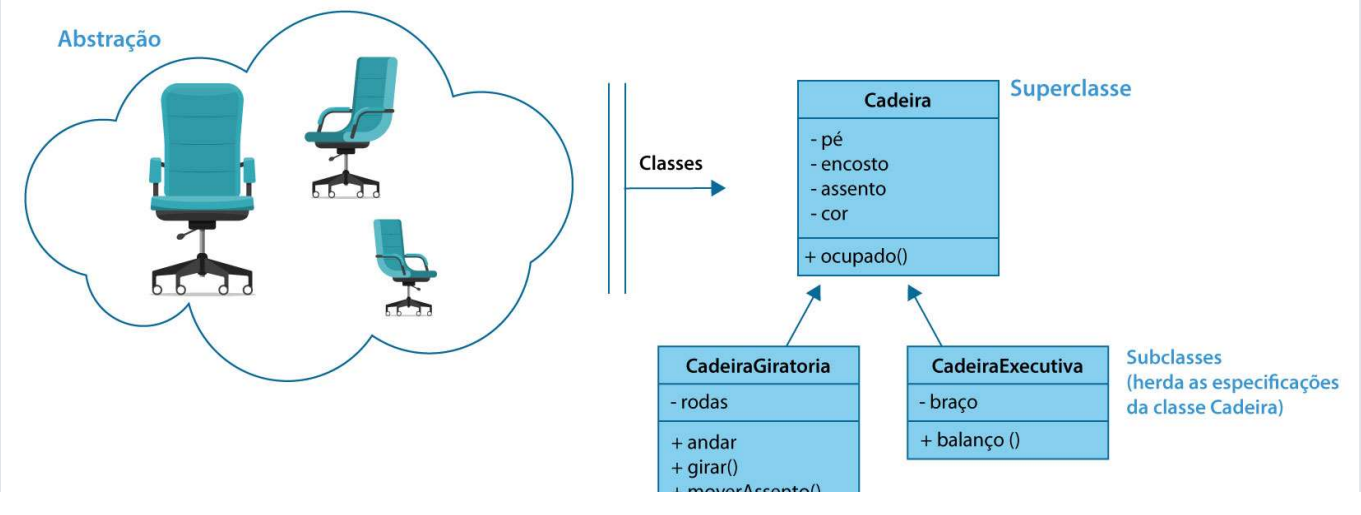
Durante o processo de modelagem serão refinadas e construídas subclasses que poderão herdar as características e comportamentos da classe genérica.

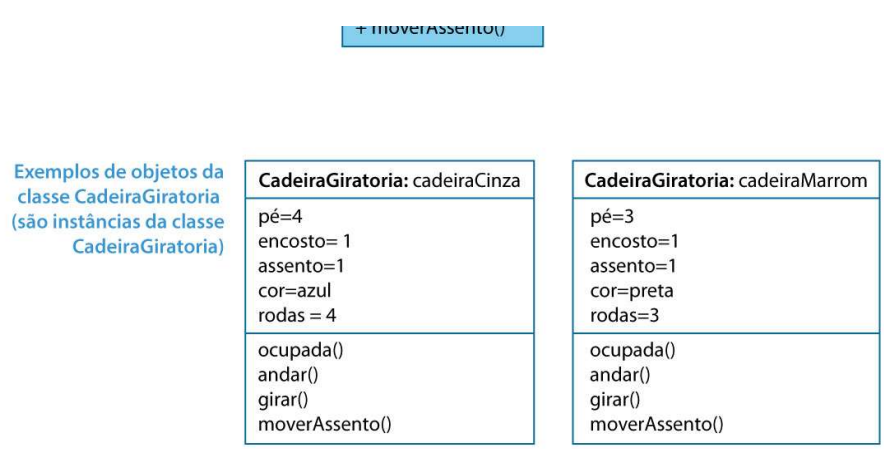
- **Superclasse:** é a classe genérica (Cadeira)
- **Subclasse:** é a classe que herda as características da superclasse (CadeiraGiratoria)

Exemplo: nova classe CadeiraGiratoria, acrescentando a ela o atributo rodas e os métodos andar, girar, mover assento.

Saiba Mais

POO – abstração, classe, herança e instanciação





Fonte: elaborada pela autora.

Encapsulamento

Na POO, o encapsulamento tem capacidade de tornar a visibilidade das informações e os detalhes da implementação dos métodos de uma classe oculta ou restrita. Isola os atributos da classe e suas ações, garantindo a integridade e a ocultação dos dados e ações. O relacionamento entre as classes passa a ser feito por mensagens.

Exemplo: você pressiona o comando abrir do objeto controle remoto do portão eletrônico e ele transmite a mensagem para o portão que executa o pedido, mas você não precisa ter conhecimento de como isso é feito.

No diagrama de classe usamos o símbolo "**-**" (um traço) para indicar que o atributo ou o método está encapsulado. Isto significa que somente o objeto da classe pode modificá-los ou acessá-los. Já o símbolo "**+**" (adição) indica que o atributo ou método é visível por outros objetos, ou seja, é público.

Poliformismo

Em uma classe, um método (operação) pode aparecer diversas vezes, pois pode ter comportamentos diferentes nas subclasses. Ou seja, objetos de um mesmo tipo de classe que apresentam comportamentos diferentes.

Exemplo: duas casas têm portões eletrônicos, porém em uma casa o portão abre para cima, e na outra, o portão é de correr na horizontal.

A orientação a objeto tornou se um paradigma muito utilizado no desenvolvimento de software e você pôde ter contato com os pilares deste paradigma: abstração, classe, objeto, herança, encapsulamento e polimorfismo.

Pesquise mais!

Desde o início da orientação a objeto, muitos trabalhos têm sido desenvolvidos e outros paradigmas vêm surgindo, portanto quanto mais você dominar este tema mais preparado você estará para desenvolver boas práticas de análise e desenvolvimento. Assim, recomendamos:

Para conhecer mais sobre Projeto Orientado a Objeto, leia o capítulo 14, págs. 208 a 214, do seguinte livro, disponível na Biblioteca Virtual:

SOMMEVILLE, I. **Engenharia de software**. 8. ed. São Paulo: Pearson - Addison Wesley, 2007.

Para melhorar ajudá-lo no processo de abstração e encapsulamento, recomendamos as páginas 26 a 38 sobre este assunto no livro a seguir, disponível na Biblioteca Virtual:

SINTES, T. **Aprenda Programação Orientada a Objetos em 21 dias**. Tradução: João Eduardo N. Tortello. São Paulo: Pearson Education do Brasil, 2002.