

Engenharia de Software I

Arquitetura de Software

Análise e Projeto de Sistemas

Orientado a Objetos e Modelagem

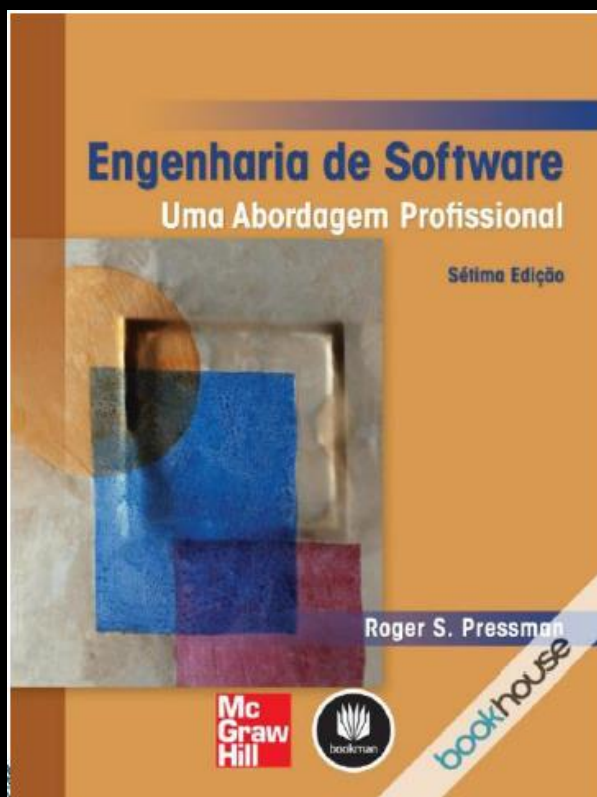
Professor Doutor Flávio Miranda de Farias

Curso Tecnólogo Sistemas para Internet

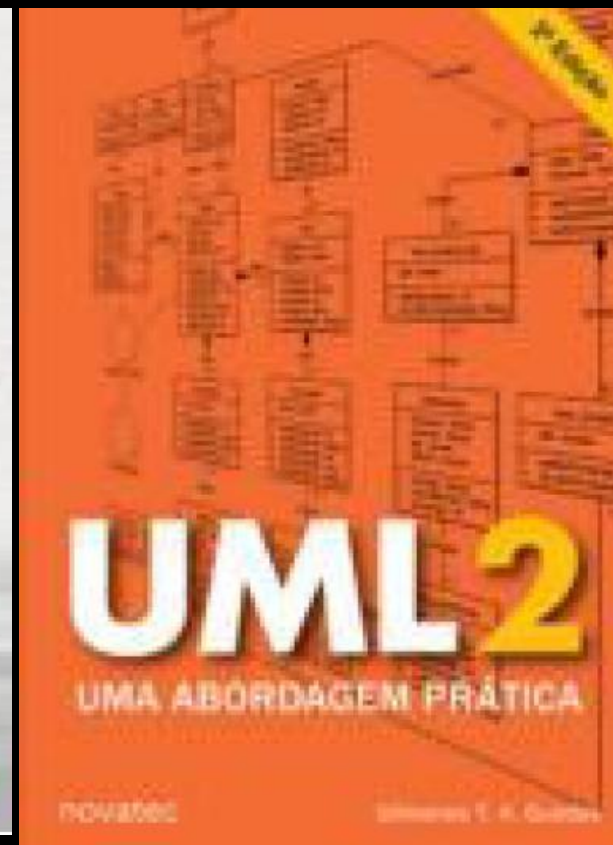
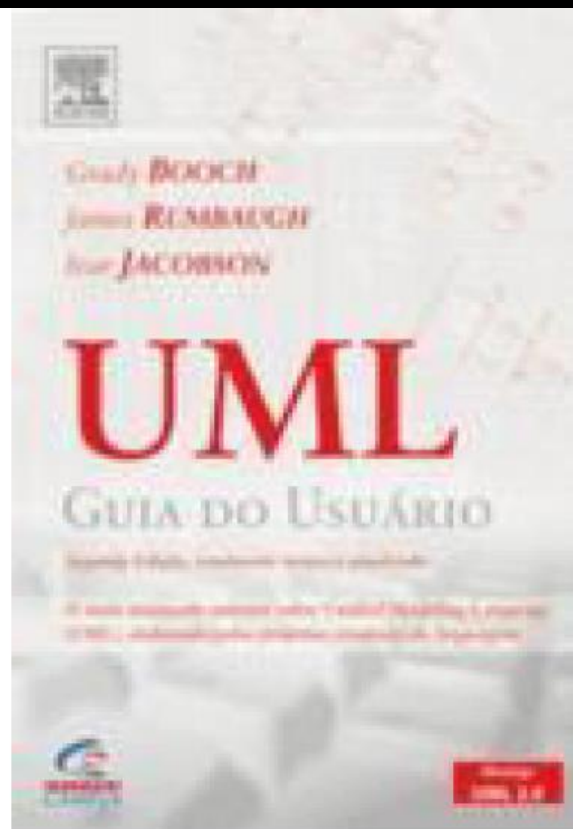
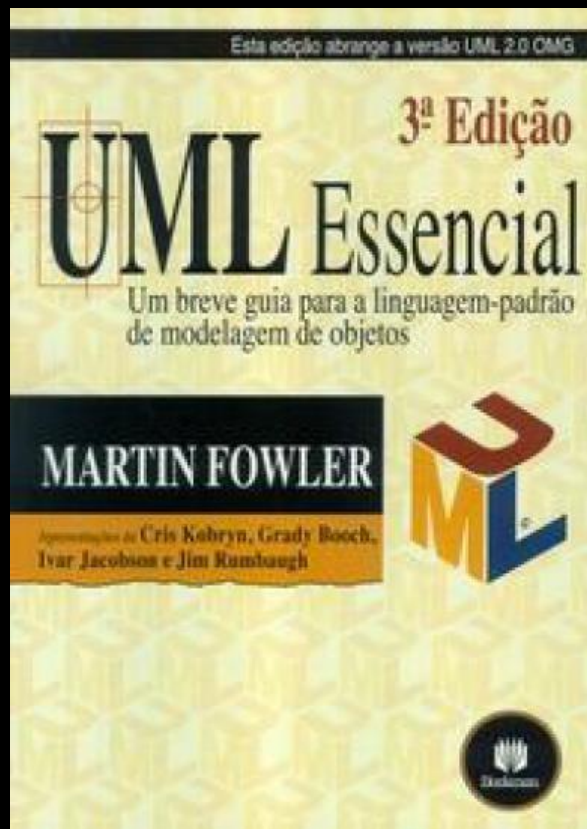
Instituto Federal de Ciência e Tecnologia do Acre - IFAC

2024-2

Livros Indicados e Utilizados: Pressman e Sommerville.
Principal Material Utilizado para os Slides: Curso do Site
www.euvoupassar.com.br



Livros Indicados e Utilizados: UML Essencial, UML Guia do Usuário, UML 2 Uma Abordagem Prática e O site oficial do UML



Análise e Projeto de Sistemas Orientado a Objetos e Modelagem

Conteúdo Programático.

- Análise e projeto de sistemas:
 - **Análise e projeto estruturado / Análise estruturada.**
 - Análise e projeto OO.
 - Conceitos fundamentais.
 - Análise.
 - Modelagem.
 - Padrões de projeto.
 - UML.

O que é Modelagem?

- A abstração do sistema de *software* através de modelos que o descrevem é um poderoso instrumento para o entendimento e comunicação do produto final que será desenvolvido.
 - A maior dificuldade nesta atividade está no equilíbrio (*tradeoff*) entre simplicidade (favorecendo a comunicação) e a complexidade (favorecendo a precisão) do modelo.

Em que se Divide a Modelagem?

- Para gestão de controle de projetos de sistemas, foram apresentados várias modelagens para atender esta necessidade, a seguir as três principais:
 - **Análise estruturada**, criada por Gane & Searson;
 - **Análise Essencial**, criada por Palmer & McMenamin e Ed. Yourdon;
 - **UML**, criada por Grady Booch, Ivar Jacobson & Jaimes Rumbaugh. É hoje o método mais comum para o paradigma orientado a objetos.

Analise Estruturada

- É uma atividade de construção de modelos. Utiliza uma notação que é própria ao método de análise estruturada para com a finalidade de retratar o fluxo e o conteúdo das informações utilizadas pelo sistema, dividir o sistema em partições funcionais e comportamentais e descrever a essência daquilo que será construído.

Analise Estruturada - Modelo Ambiental

- O modelo ambiental descreve o ambiente no qual o sistema se insere, ou seja, descreve o contexto do sistema, que deve ter 3 componentes:
 - **Definição de objetivos** → Finalidade de sistema;
 - **Lista de eventos** → Os acontecimentos que ocorrem no exterior e que interagem com o sistema;
 - **Diagrama de contexto** → Representa o sistema como um único processo e as suas interações com o meio ambiente.

Analise Estruturada - Modelo comportamental

- Descreve as ações que o sistema deve realizar para responder da melhor forma aos eventos definidos no modelo ambiental. Técnicas utilizadas:
 - Diagrama de fluxos de dados (DFD);
 - Dicionário de dados (DD);
 - Diagrama de entidades e associações (ou relacionamentos) (Diagrama entidade relacionamento [DER] ou Modelo de entidades e relacionamentos [MER]);
 - Especificação de processos (EP) - (DESENHO);
 - Diagrama de transição de estados (DTE).

Analise Estruturada - Diagramas

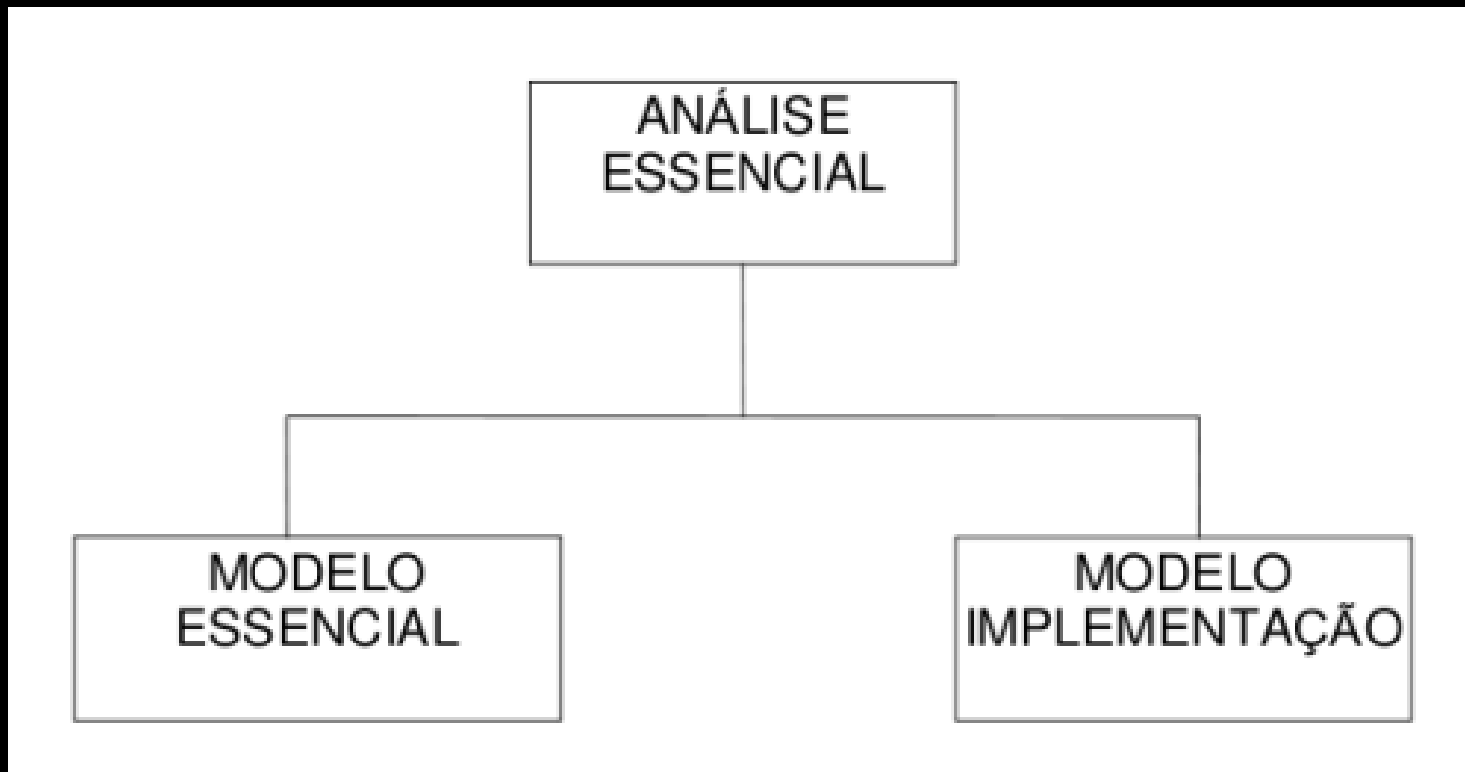
- A analise estruturada pode ser trabalhada com os seguintes diagramas:
 - Diagrama de contexto
 - Diagrama de fluxos de dados
 - Diagrama entidade relacionamento
 - Lista de eventos
 - Tabela de decisão
 - Árvore de decisão
 - Diagrama de transição de estados

Análise Essencial

- Ela propõe o particionamento do sistema por eventos. A rigor, o valor de um sistema está na sua capacidade de responder com eficácia a todos os estímulos a que for submetido. Assim, um sistema é construído para responder a estímulos. A cada estímulo, o sistema deve reagir produzindo uma resposta predeterminada.
- A expressão *Essential Analysis*, traduzida por Análise Essencial, foi proposta em 1984 por McMenamim e Palmer para refletir a introdução dos novos conceitos que estavam sendo incorporados à Análise Estruturada clássica.

Análise Essencial

- Na Análise Essencial existem dois modelos, denominados de Modelo Essencial e Modelo de Implementação.

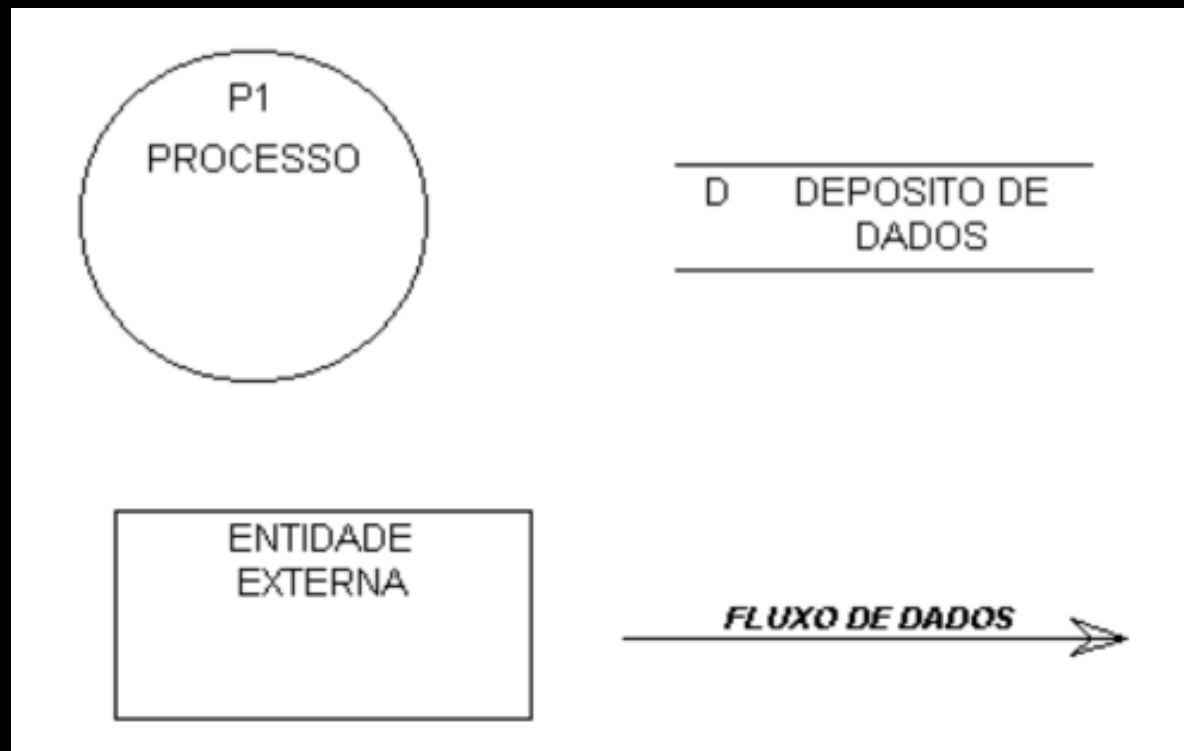


Análise Essencial – Modelo Essencial

- Apresenta o sistema em um nível de abstração completamente independente de restrições tecnológicas.
- O Modelo Essencial é formado por:
 - Modelo Ambiental: Define a fronteira entre o sistema e o resto do mundo.
 - Modelo Comportamental: Define o comportamento das partes internas do sistema necessário para interagir com o ambiente.

Análise Essencial – Modelo Implementação

- Tem como objetivo definir a forma de implementação do sistema em um ambiente técnico específico. Apresenta o sistema num nível de abstração completamente dependente de restrições tecnológicas.



Qual usar?

- Iremos adotar a terceira opção que é a **UML** que a mais usada atualmente, principalmente para linguagens Orientadas a Objetos e além disto, possui mais material didático disponível no mercado.
- Estaremos posteriormente estudando a UML em conjunto a conceitos de Orientação a Objetos.

Exercícios

- Pesquisar modelo de sistemas reais na internet com base na **Análise Estruturada** e **Análise Essencial** e trazer para próxima aula.

Conteúdo Programático.

- Análise e projeto de sistemas:
 - Análise e projeto estruturado / Análise estruturada.
 - **Análise e projeto OO.**
 - Conceitos fundamentais.
 - Análise.
 - Modelagem.
 - Padrões de projeto.
 - UML.

Analise de projeto Orientado a Objetos

Análise e projeto OO

Conceitos Gerais - OO

- 🌐 Análise Estruturada não representa o mundo real.
- 🌐 Necessidade de maior aproximação entre sistemas de informação e o mundo real.
- 🌐 O ser humano pensa em objetos.
- 🌐 Carro.
 - 🌐 4 rodas no mínimo.
 - 🌐 2 portas no mínimo.
 - 🌐 Farol.
 - 🌐 Freio.
 - 🌐 Anda.
 - 🌐 Transporta pessoas.



Características



Operações

| Nome |
|-----------|
| Atributos |
| Métodos |

Análise e projeto OO

Conceitos Gerais - OO

- Fronteiras entre Engenharia de Software e Desenvolvimento.
- RUP independente mas orientado à UML e OO.
- UML completamente inserida no conceito de OO.
- Simula 67 (Ole Johan Dahl e Kristen Nygaard).
- Smaltak 1970 (Alan Kay).
- PHP, C++, Delphi, Java, Object Pascal.

Análise e projeto OO





Conceitos Gerais - OO

- Modela os objetos do mundo real com correspondentes em software.
- Fornece uma maneira mais natural e intuitiva de ver o processo de programação modelando objetos do mundo real, seus atributos e seus comportamentos.
- Busca a estrutura do problema, e não apenas da informação.
- Identifica em objetos, os elementos importantes do domínio do problema que tratam com dados e possuem funções que podem operar os dados.



Análise e projeto OO

Conceitos Gerais - OO

Vantagens:

-  Maior componentização.
-  Facilidade de manutenção.
-  Maior reaproveitamento de código.
-  Melhor gestão do desenvolvimento (UML -> Classe e Pacotes).

Desvantagens:

-  Entendimento de conceitos extremamente complexos.
-  Desempenho em tempo de execução.

Componente -> Independente de ambiente
e executável



São diferentes

Objeto -> Precisa de estrutura pré-
configurado, preciso de estímulo externo

Análise e projeto OO

Conceitos Gerais - OO





Abstração:

-  Enfocar os aspectos mais importantes de um objeto (visão externa).
-  Ignorar suas características internas (visão interna).

Análise e projeto OO

Conceitos Gerais - OO







Tipo:

-  Segue uma mesma estrutura e definem o que chama-se de um tipo abstrato de dado.
-  É uma estrutura de dados com um conjunto de definições de operações que afetam esta estrutura.
-  Serve para se definir um padrão de modelo.
-  A implementação de um tipo é uma classe que cria com suas operações um componente autônomo para o sistema.

Análise e projeto OO

Conceitos Gerais - OO

Classe:

-  Representa uma categoria.
-  Modelo inicial a ser utilizado.
-  Abstração que define um tipo de objeto e o que determinado tipo de objeto tem dentro dele.
-  Identifica um grupo de objetos com as mesmas características.
-  Trata-se de um conceito.
-  Você moraria na planta de uma casa?

| Nome |
|-----------|
| Atributos |
| Métodos |

Análise e projeto OO

Conceitos Gerais - OO

Objetos:





- São os membros ou exemplos dessa categoria.
- Os objetos podem executar seus métodos.
- São instâncias das classes.
- Possuem os atributos relativos à classe.
- Caracterizado por um conjunto de operações e um estado que armazena os efeitos das operações que o objeto é capaz de realizar.
- Aqui está a casa.



Análise e projeto OO

Conceitos Gerais - OO

Atributos X Propriedades:

-  Classes possuem propriedades ou características específicas.
-  Atributos são estas propriedades.
-  Representados na segunda divisão da classe.
-  Nome: tipo de dado que o objeto armazena (não obrigatório).

| Casa |
|----------------------------------|
| 30° andar surla garagem |
| |

Classes - > Propriedades

Objetos -> Atributos

Análise e projeto OO

Conceitos Gerais - OO

- 🌐 Métodos, Operações ou Comportamentos.
 - 🌐 A UML utiliza o termo Operação.
 - 🌐 Atividade que um objeto de uma classe pode executar.
 - 🌐 Podem receber parâmetros e retornar valores (não obrigatoriamente).
 - 🌐 Conjunto de instruções que são executadas quando o método é chamado.

| Casa |
|------------------|
| cor |
| andar |
| porta |
| garagem |
| abrigarFessoas() |

Exercício

- Criar classes relacionadas a geladeira, prefeito, torradeira, cadeira, cachorro, aluno, computador e humano.

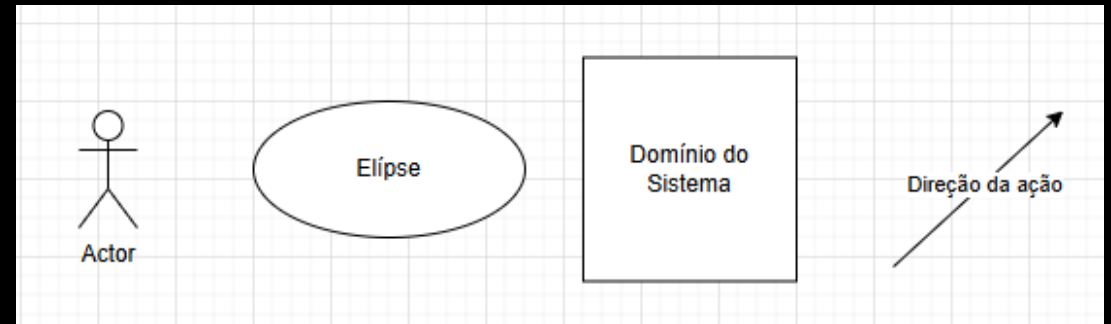
Análise e projeto OO

Conceitos Gerais – OO - Associação

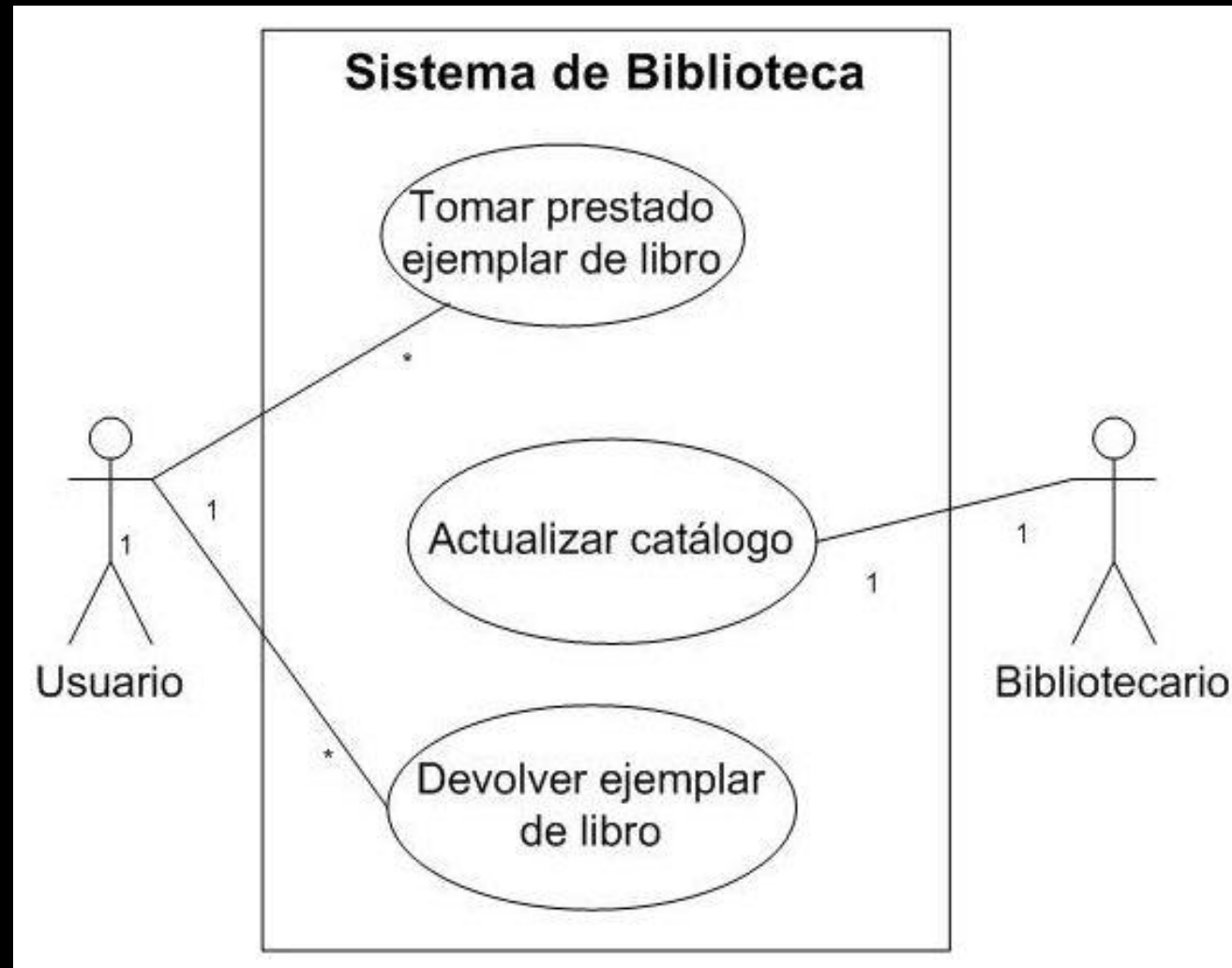
- Associações representam as interações ou relacionamento entre os atores que fazem parte do diagrama, entre os atores e os casos de uso ou os relacionamentos entre os casos de uso e outros casos de uso.
- Os relacionamentos entre casos de uso recebem nomes especiais (inclusão, extensão e generalização).

Exemplos de Associação para Casos de Uso

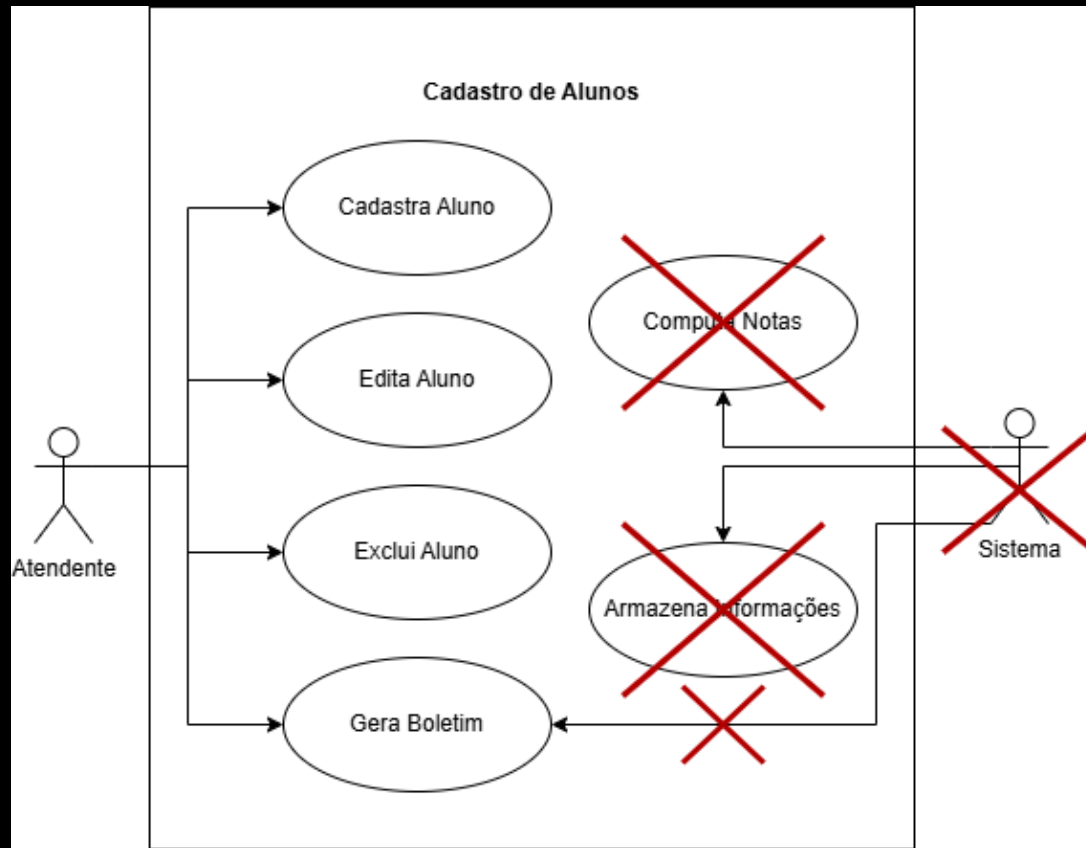
- UML de Caso de Uso é uma ferramenta usada para associar ações do usuário no sistema.
- Ela explica como os usuários do sistema utilizam o sistema como um todo ou partes do sistema.
- Normalmente usado para explicar o código para pessoas leigas, utiliza basicamente as ferramentas ao lado.



Exemplos de Casos de Uso



Erros Comuns em Diagramas de Caso de Uso



- Nos diagramas de Caso de uso, não se diagrama as funções realizadas por não humanos, o foco é em ações diretas realizadas por humanos no sistema.

Exercício

- Construa um exemplo simples de caso de uso com pelo menos 2 atores.

Análise e projeto OO

Conceitos Gerais - OO

🌐 Princípios/pilares da Orientação a Objeto:





- 🌐 Encapsulamento.
- 🌐 Herança.
- 🌐 Composição.
- 🌐 Polimorfismo.

Atenção
Sempre cai em
concursos

Análise e projeto OO

Conceitos Gerais – OO – Encapsulamento

Encapsulamento:

-  Capacidade de ocultar partes de implementação interno de classes do mundo exterior.
-  Possibilita a sua visão das apenas por seus métodos e não como eles são implementados.
-  A “cápsula” que protege o objeto que possui características de encapsulamento se chama interface que utiliza-se de mensagens para tal feito. (Motorista, Carro).
-  Implica na colaboração entre os objetos pela troca de mensagens. (TV, DVD).

Análise e projeto OO

Conceitos Gerais – OO – Encapsulamento


Encapsulamento:

- Facilita a reutilização.
- Windows media player, Winamp, MP3 veicular. **Ex.: Botões de mídia**
- O que caracteriza um encapsulamento eficiente é a definição precisa da interface.
- Mensagem.
 - Proporciona a comunicação entre objetos.
 - Chamada de uma função de um objeto, acionamento de uma operação encapsulada no objeto destino.
 - Feita do objeto de origem.
 - Por padrão toda mensagem tem uma resposta de retorno e pode transmitir uma informação nachamada e no retorno.

Análise e projeto OO

Conceitos Gerais – OO – Encapsulamento

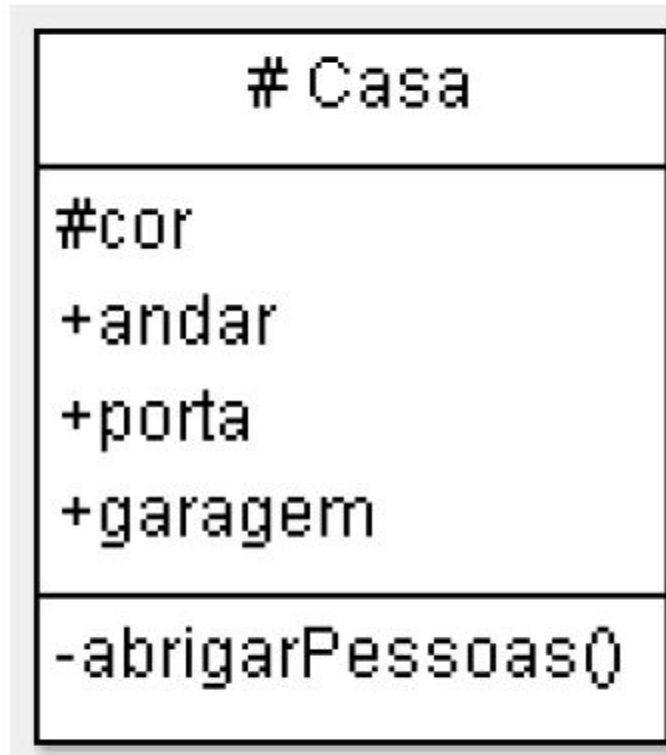
Visibilidade:

-  Utilizada para indicar o nível de acessibilidade de um atributo ou método.

| Modos de Visibilidade | Apenas Objetos da Classe | Apenas Objetos da Subclasse | Apenas Objetos do Pacote | Qualquer Objeto |
|---------------------------------------|--------------------------|-----------------------------|--------------------------|-----------------|
| Privada/ Private (-) | X | | | |
| Protegida/ Protected (#) | X | X | | |
| Pública / Public(+) | X | X | X | X |
| Pacote/ Package ou Friendly (~) | | | X | |

Análise e projeto OO




Conceitos Gerais – OO – Encapsulamento



Análise e projeto OO

Conceitos Gerais – OO – Composição ou Agregação

Composição ou Agregação:







-  Mecanismo de reaproveitamento de classes utilizado pela OO par aumentar a produtividade e a qualidade do desenvolvimento de software.
-  Quando uma classe é composta de outras classes ela pode tanto usar os objetos que são gerados pelas classes que a compõem, como pode também usufruir dos atributos e métodos dessas classes.
-  Quando uma classe usa a Composição para agregar outras classes, podemos dizer que ela tem um relacionamento chamado “Tem um”.

Ex.: Carro tem pneu
Carro tem porta

Análise e projeto OO

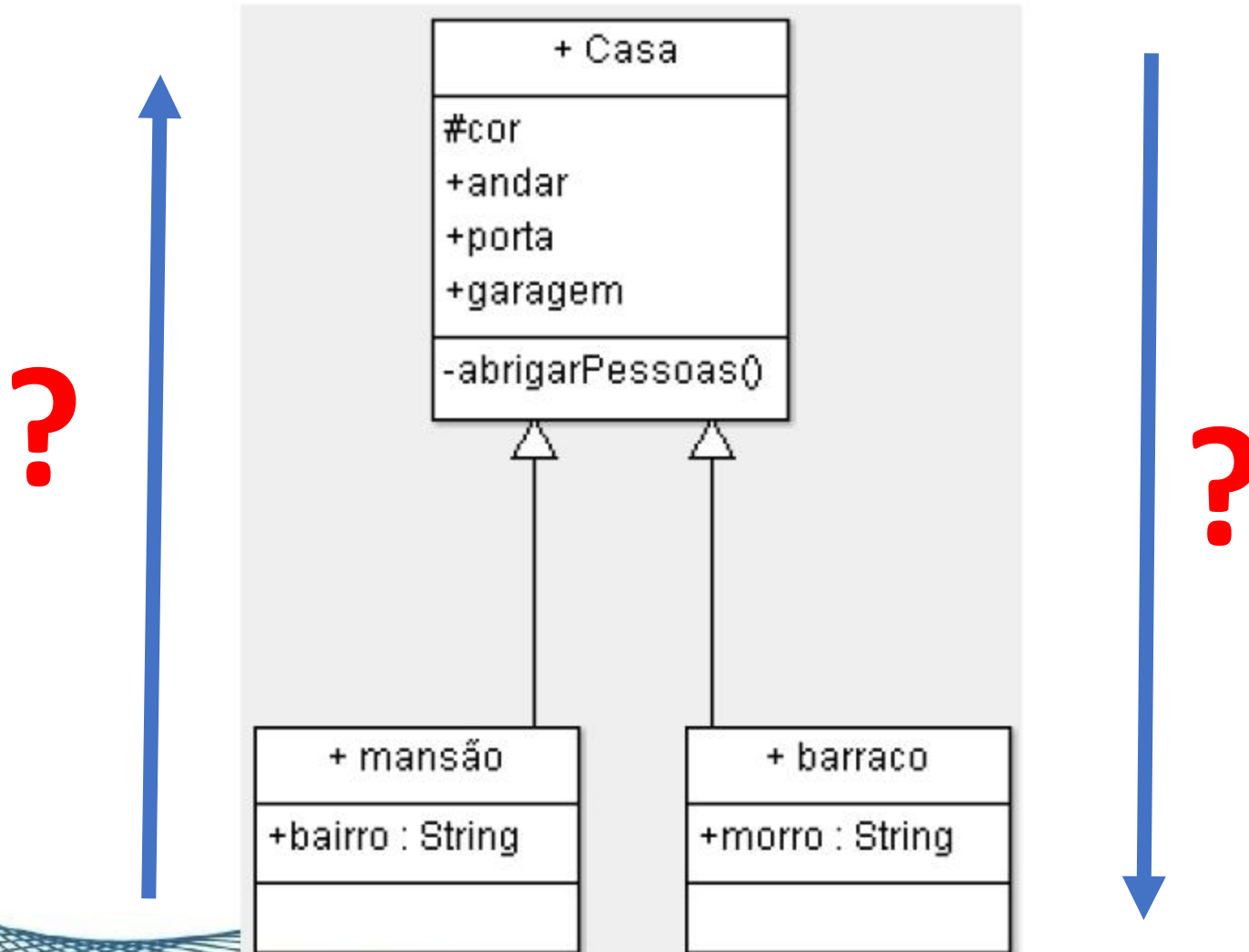
Conceitos Gerais – OO – Herança

Herança:

-  Permite reaproveitamento de atributos e métodos.
-  Permite a diminuição de linhas de código.
-  Superclasse ou classe mãe é uma classe que contém classes derivadas a partir dela, chamadas de subclasses ou classes-filha.
-  Quando declaramos atributos e métodos na classe mãe, não precisamos declarar na classe filha, pois ela herda automaticamente.
-  Declara-se no entanto os atributos e métodos exclusivos da classe filha.
-  Superclasses são genéricas e subclasses são especializadas.

Análise e projeto OO

Conceitos Gerais – OO – Herança



Análise e projeto OO

Conceitos Gerais – OO – Herança

```
class Casa {  
    protected cor;  
    public andar;  
    public porta;  
    public garagem;  
    private void abrigarPessoas() {  
        }  
}
```

Análise e projeto OO



Conceitos Gerais – OO – Herança

```
class mansão extends Casa {  
    public String bairro;  
}  
  
class barraco extends Casa {  
    public String morro;  
}
```

Análise e projeto OO

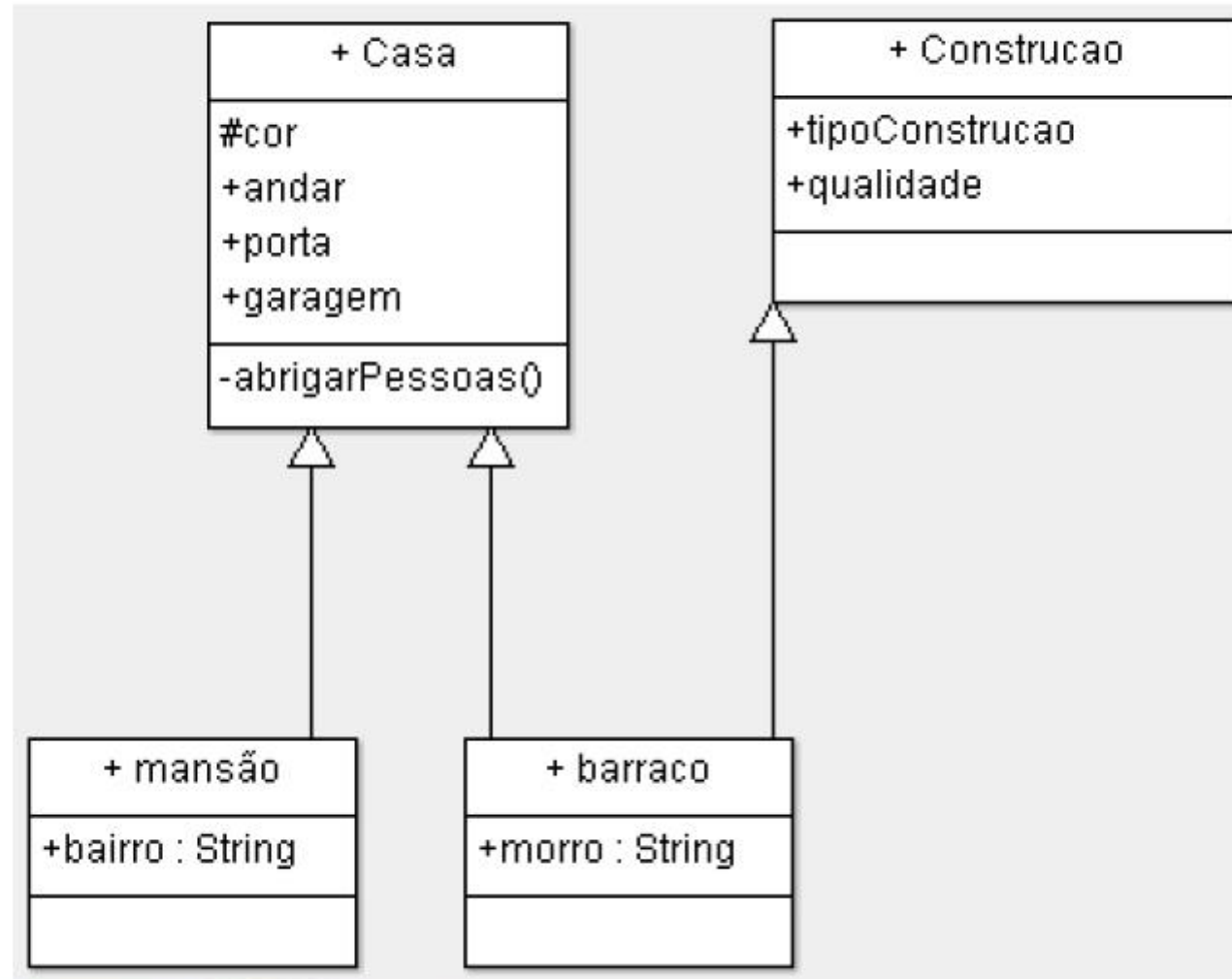
Conceitos Gerais – OO – Herança

Herança Múltipla:

-  Ocorre quando uma subclasse herda características de duas ou mais superclasses.
-  O Java não implementa herança múltipla, simula tal feito com o uso de interfaces.

Análise e projeto OO

Conceitos Gerais – OO – Herança



Análise e projeto OO

Conceitos Gerais – OO – Herança

```
class Casa {  
    protected cor;  
    public andar;  
    public porta;  
    public garagem;  
    private void abrigarPessoas() {  
        }  
}
```



Análise e projeto OO




Conceitos Gerais – OO – Herança

```
class Construcao {  
    public tipoConstrucao;  
    public qualidade;  
}  
  
class mansão extends Casa {  
    public String bairro;  
}  
  
class barraco extends Casa, Construcao {  
    public String morro;  
}
```

Análise e projeto OO

Conceitos Gerais – OO – Polimorfismo



 Polimorfismo:

-  Está relacionado com herança, mas não é herança. **OH**
-  Teremos então 2 métodos com o mesmo nome e diferentes somente pela sua maneira de implementar.
-  O método polimórfico só irá se comportar de forma diferente nos objetos da classe que o modificou.



Análise e projeto OO

Conceitos Gerais – OO – Polimorfismo

Polimorfismo de sobrecarga:

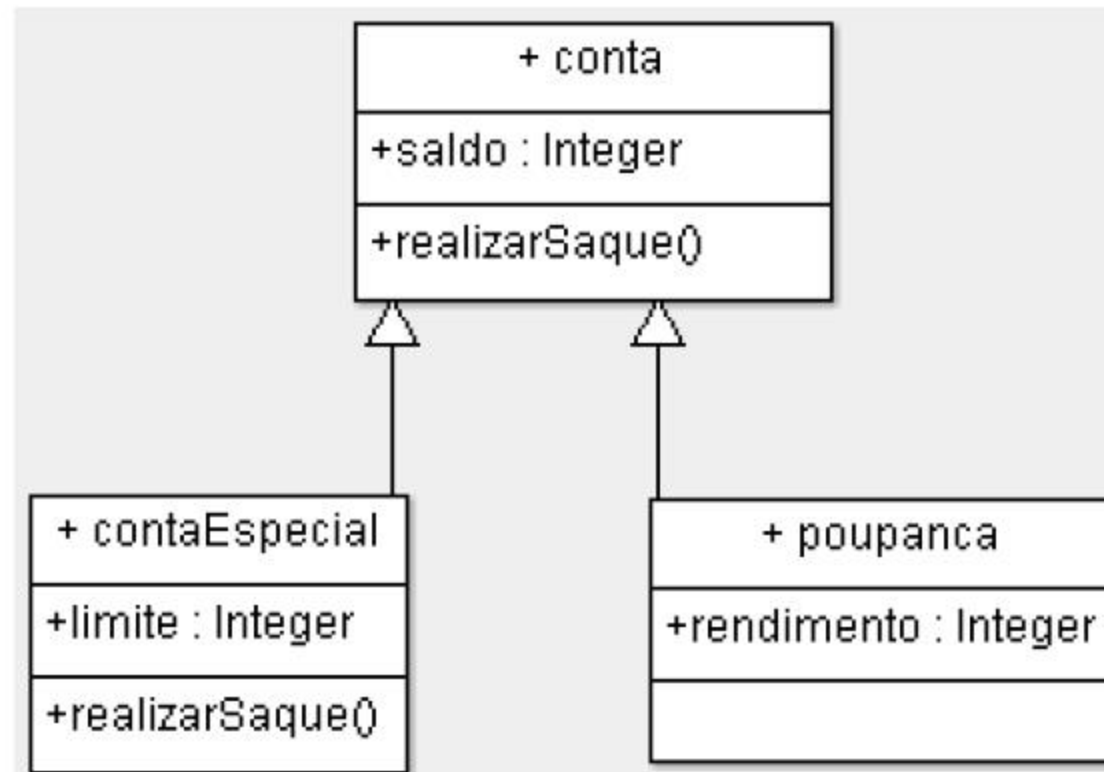
-  Permite que um método tenha comportamentos distintos, em função de diferentes parâmetros que ele recebe.
-  Ocorre normalmente nos métodos construtores.

Polimorfismo de sobreposição.

-  Redefinição de métodos em classes descendentes.
-  O método de uma classe filha com o mesmo nome de uma classe mãe irá sobrepo-lo.

Análise e projeto OO

Conceitos Gerais – OO – Polimorfismo



Análise e projeto OO




Conceitos Gerais – OO – Polimorfismo

```
class conta {  
    public Integer saldo;  
    public void realizarSaque() {  
    }  
}  
  
class contaEspecial extends conta {  
    public Integer limite;  
    public void realizarSaque() {  
    }  
}
```



Análise e projeto OO

Conceitos Gerais – OO – Polimorfismo

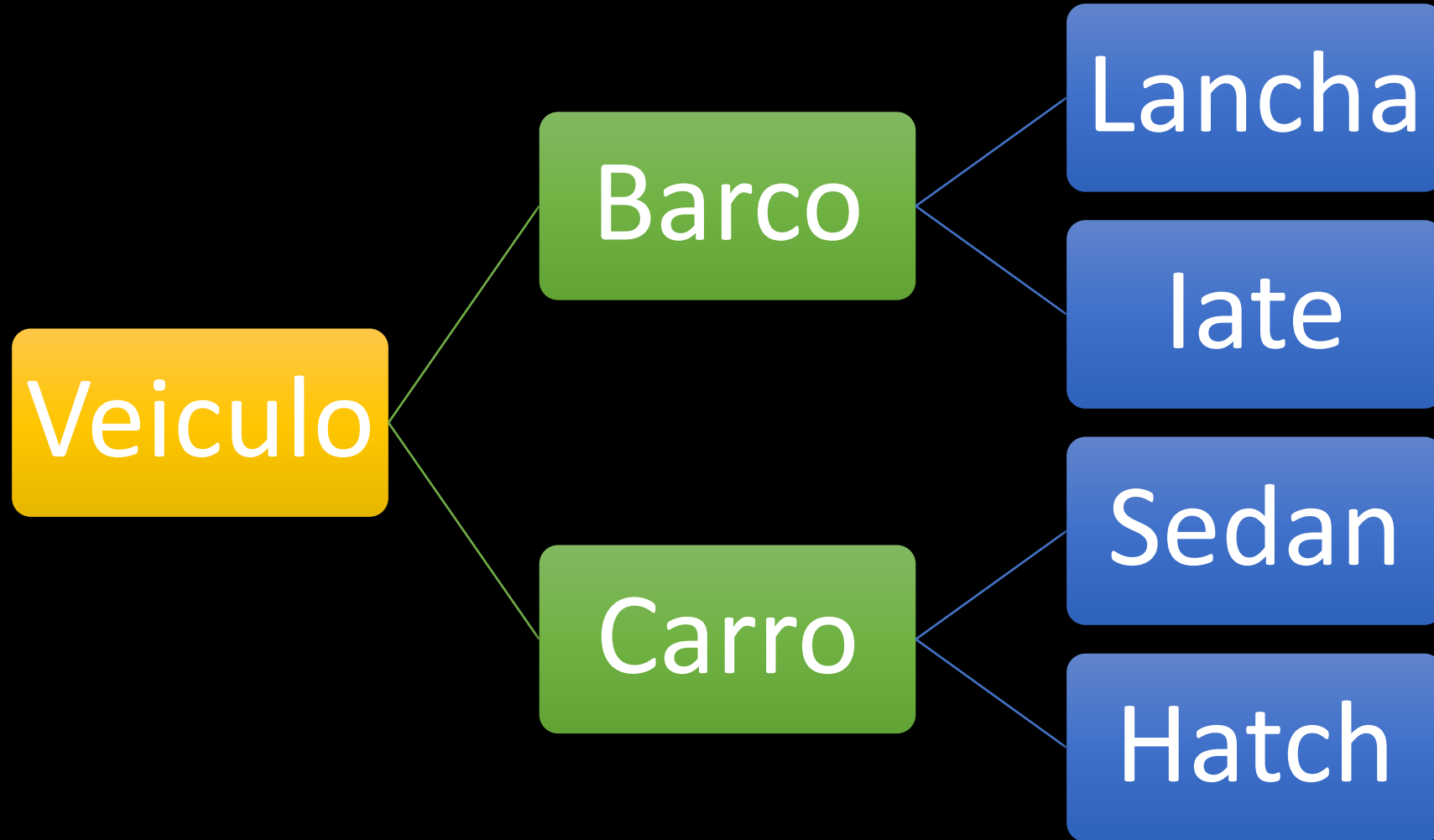
Classe Abstrata:

-  Representa uma coleção de características presentes em vários tipos de objetos, mas não existe isoladamente e automaticamente não pode ser instanciada.
-  É somente um conjunto de características em comum das subclasses. (Animal, Humano, Cachorro).
-  Aparece normalmente em *itálico* o seu nome.

Casting:

-  Utilizado quando queremos forçar uma classe a apresentar aquilo que ela não é (Animal, Humano, Cachorro).
-  Ex: Utilização de Array para Humano e Cachorro.

Abstração



Casting

Animal

?

?



Casting



Requisitos Funcionais e Não Funcionais de Sistemas

- O que são requisitos?
 - Declarações que definem o comportamento, as propriedades e as restrições de um sistema.
 - Dividem-se em funcionais e não funcionais.
- Importância:
 - Garantem que o sistema atenda às necessidades dos stakeholders.

Requisitos Funcionais

- Definição:
 - Descrevem o que o sistema deve fazer.
 - Focam nas funcionalidades e no comportamento esperado.
- Exemplos:
 - O sistema deve permitir o cadastro de usuários.
 - Deve enviar um e-mail de confirmação após o registro.
 - Deve gerar relatórios mensais de vendas.
 - Deve permitir buscas por produtos com base em categorias e palavras-chave.
 - Deve processar pagamentos online por meio de cartões de crédito e boleto bancário.

Requisitos Não Funcionais

- Definição:
 - Descrevem como o sistema deve se comportar ou operar.
 - Relacionam-se a propriedades de qualidade e restrições.
- Exemplos:
 - O sistema deve responder em menos de 2 segundos para 95% das requisições.
 - Deve suportar até 10.000 usuários simultaneamente.
 - Deve estar em conformidade com a LGPD (Lei Geral de Proteção de Dados). Deve garantir 99,9% de disponibilidade mensal.
 - Deve criptografar todos os dados sensíveis transmitidos entre o cliente e o servidor.

Diferenças entre Requisitos Funcionais e Não Funcionais

- Funcionais:
 - O foco está no que o sistema deve fazer.
 - Exemplos: funcionalidades, interações, processos.
- Não Funcionais:
 - O foco está em como o sistema opera.
 - Exemplos: desempenho, segurança, manutenção.

Sistema de Gerenciamento de Biblioteca Online

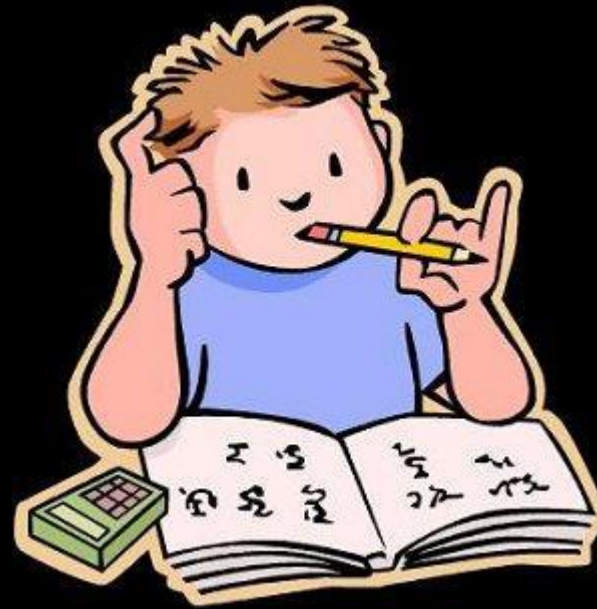
Requisitos Funcionais:

- O sistema deve permitir o cadastro de usuários com dados como nome, e-mail e senha.
- Deve possibilitar o login de usuários registrados.
- Deve permitir a busca de livros por título, autor ou gênero.
- Deve possibilitar o empréstimo e a devolução de livros online.
- Deve enviar notificações por e-mail sobre o prazo de devolução de livros.
- Deve permitir que administradores adicionem, editem ou removam livros do catálogo.
- Deve gerar relatórios de livros mais emprestados e usuários ativos.

Requisitos Não Funcionais:

- O sistema deve responder às buscas por livros em até 2 segundos.
- Deve suportar até 500 usuários simultâneos.
- Deve garantir 99% de disponibilidade mensal.
- Todas as informações sensíveis, como senhas, devem ser criptografadas.
- Deve estar em conformidade com a LGPD (Lei Geral de Proteção de Dados).
- A interface deve ser responsiva e compatível com dispositivos móveis.
- Deve realizar backups automáticos diários dos dados armazenados.

Exercícios



Exercícios

1. (TER-PI - 2016 – CESPE) Considerando o desenvolvimento de um projeto de software orientado a objetos, projetar a arquitetura do sistema envolve identificar os principais componentes do sistema e suas interações.
2. (SEE-DF - 2017 – Quadrix) Uma das desvantagens da programação orientada a objetos está no fato de que seus programas são de difícil manutenção, uma vez que esse tipo de abordagem lida com problemas complexos.
3. (CFO-DF - 2017 – Quadrix) Na programação orientada a objetos, é por meio dos objetos que se modela o software em termos reais.
4. (CFO-DF - 2017 – Quadrix) Desde que empregada corretamente, a UML, por meio de diagramas, consegue capturar a estrutura de sistemas orientados a objeto.

Exercícios

5. (CONTER - 2017 – Quadrix) A UML é composta de cinco diagramas, incluindo a descrição dos casos de uso.
6. (CFO-DF - 2017 – Quadrix) A UML é uma linguagem independente de linguagens de programação, mas não de processos de desenvolvimento de sistemas.
7. (NC-UFPRR - 2017 – ITAIPU) O Diagrama de Casos de Uso representa atores e casos de uso para modelar os comportamentos do sistema.
8. (NC-UFPRR - 2017 – ITAIPU) O Diagrama de Estados representa o estado final do objeto durante a troca sequencial de mensagens entre objetos.

Gabarito

- 1.V
- 2.F
- 3.V
- 4.V
- 5.F
- 6.F
- 7.V
- 8.V