Aula 5

C.E.S.A.R school

NExT

Nova Experiência de Trabalho

Novas oportunidades, novos desafios

Ao vivo | Nova edição

APOIO:

PortoDigital
Onde as conexões acontecem

SOFTEX RECIFE

C.E.S.A.R

DIAGRAMA DE CLASSE

UML, Classes e Dependências

O QUE É UML

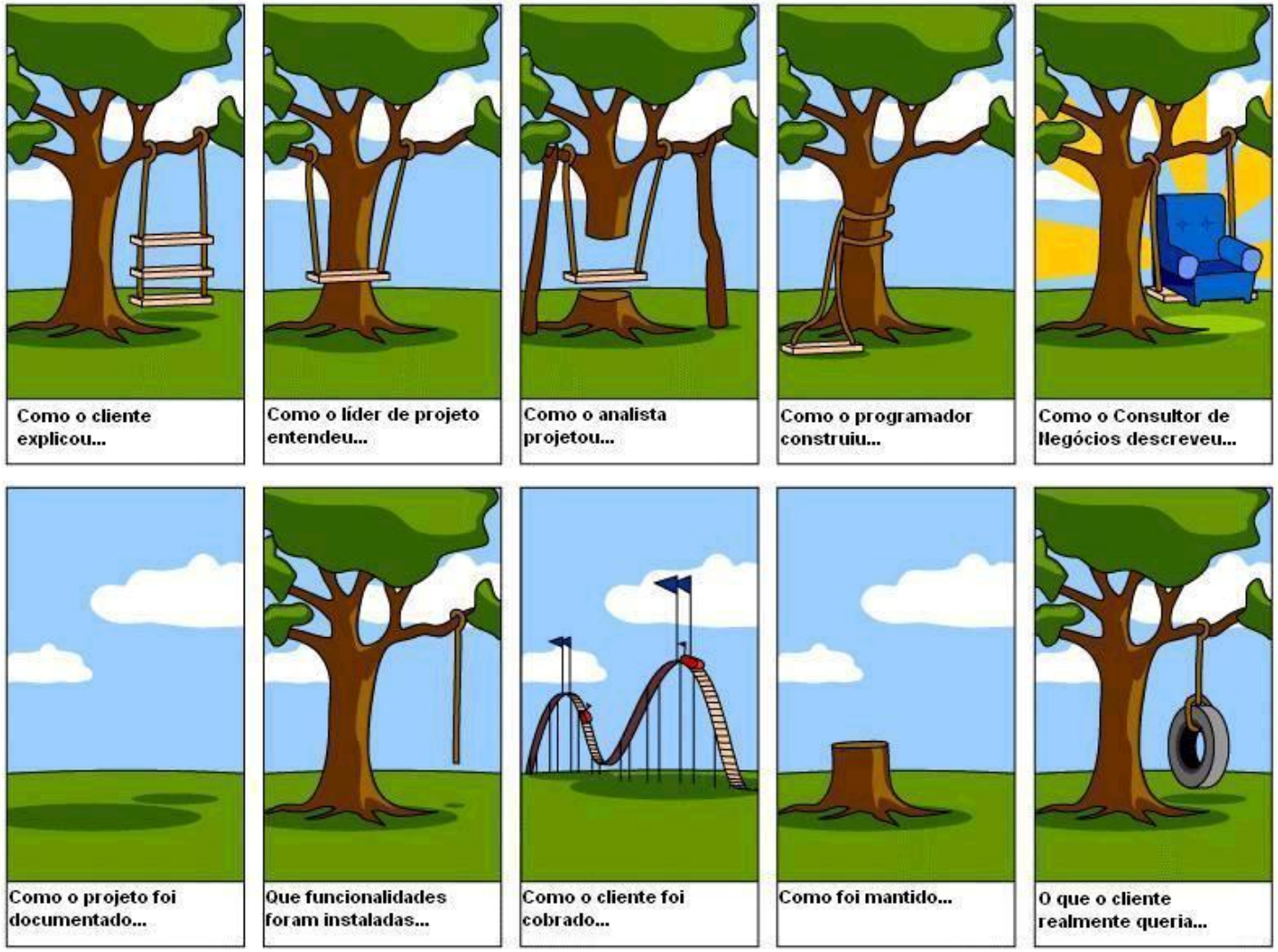
É Uma linguagem ou notação de diagramas para:

- Especificar, visualizar e documentar modelos de software
- Não é um método de desenvolvimento
- Não indica o que fazer primeiro
- Ajuda a visualizar o produto e a comunicação

PARA QUE USAR UML?

- Pensar antes de codificar
- Apresentar nossas ideias ao grupo
- Aumentar a participação e envolvimento do time
- Documentar as ideias quando já consolidadas
- Atender ao requisitos
- Reduzir esforço de manutenção
- Facilitar a alteração do software
- Reduzir retrabalho: reparos ocorrem a nível de projeto

PARA QUE USAR UML?

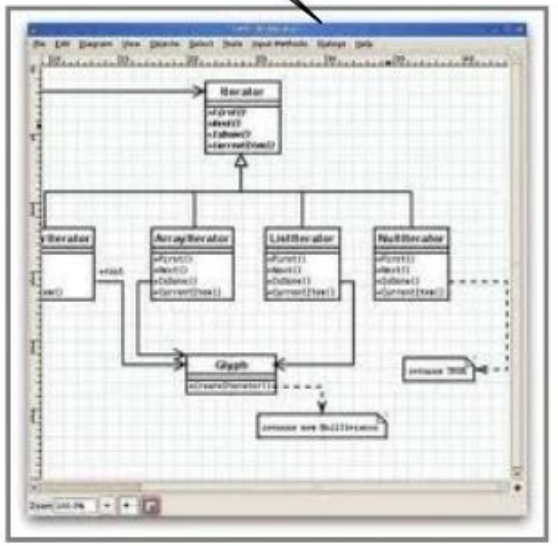


Falta de Alinhamento entre cliente, analista e time

PARA QUE USAR UML?

MODELAGEM

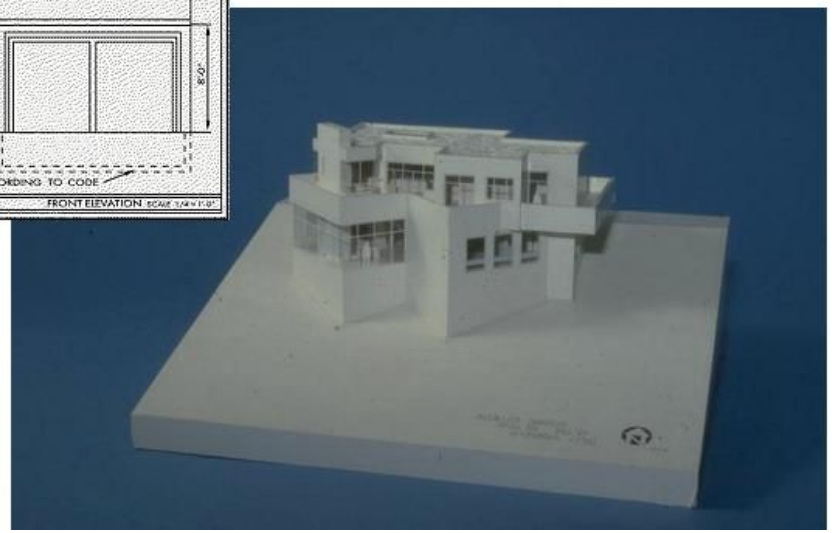
O QUE É PRECISO?



MODELAGEM VISUAL

NECESSIDADE DE UMA MV

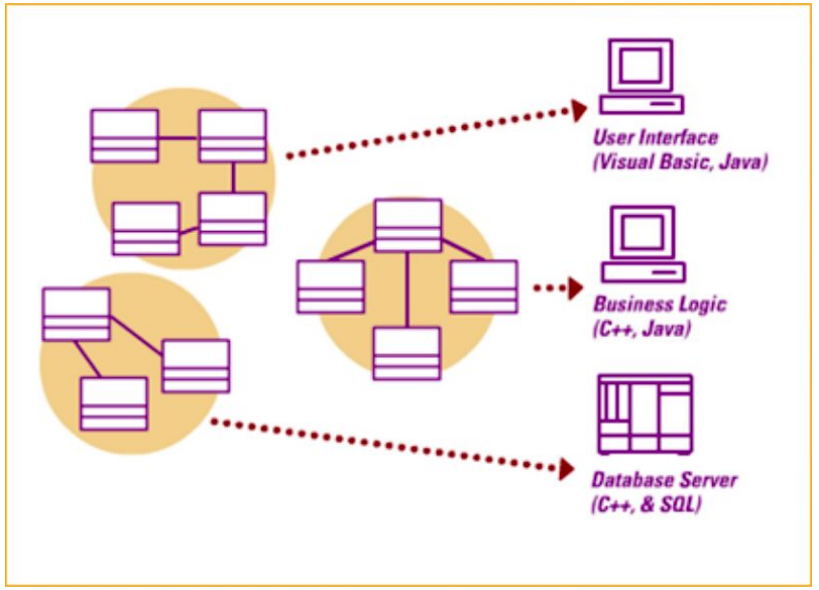
- Da mesma forma que é impossível construir uma casa sem primeiramente definir sua planta, também é impossível construir um software sem inicialmente definir sua arquitetura.



NECESSIDADE DE UMA MV

- Um modelo deve ser criado independentemente de sua implementação.

- A utilização de uma modelagem visual facilita a visualização, e, por conseguinte, a criação de um melhor modelo (mais flexível, mais robusto e principalmente mais reutilizável).



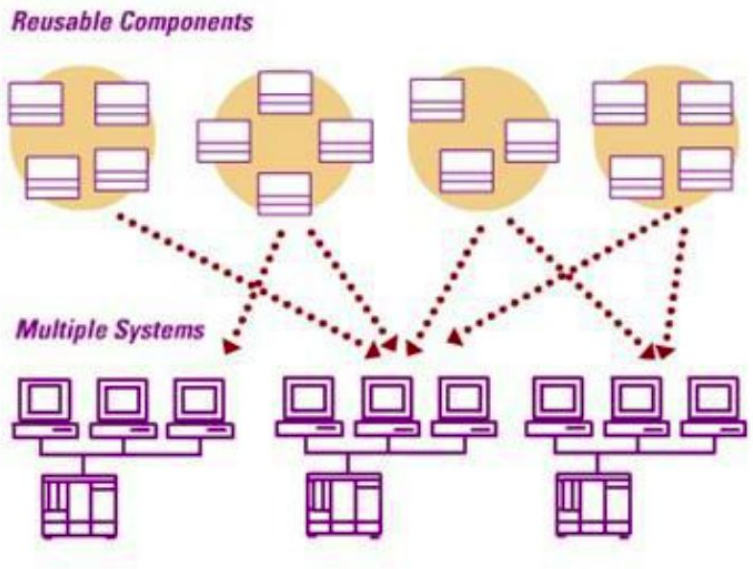


DIAGRAMA DE CLASSE

DIAGRAMA DE CLASSE

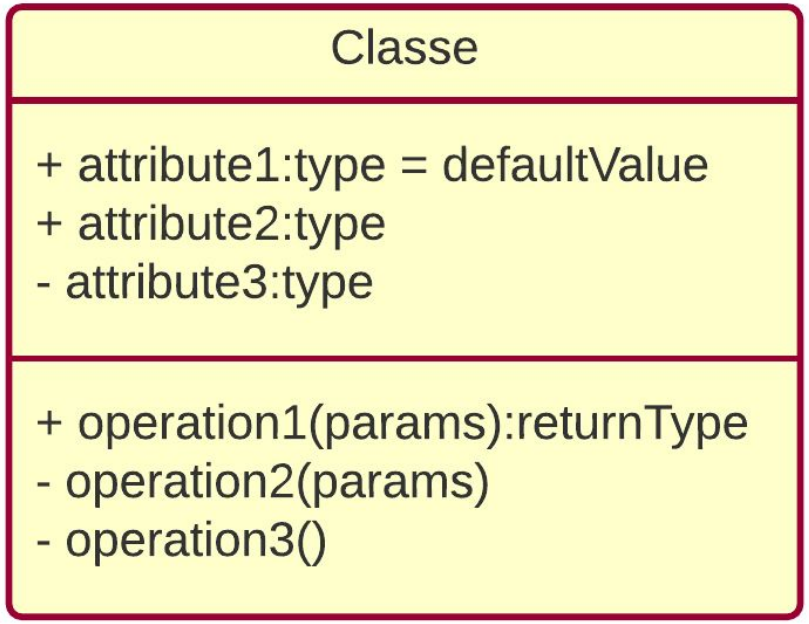
É um dos diagramas mais importantes da UML.

Objetivo:

- Descrever os vários tipos de objetivos no sistema e o relacionamento entre eles.
- Permitir a visualização das classes que irão compor o sistema com seus respectivos atributos e métodos.
- Demonstrar como as classes se relacionam, complementam e transmitem informações entre si.

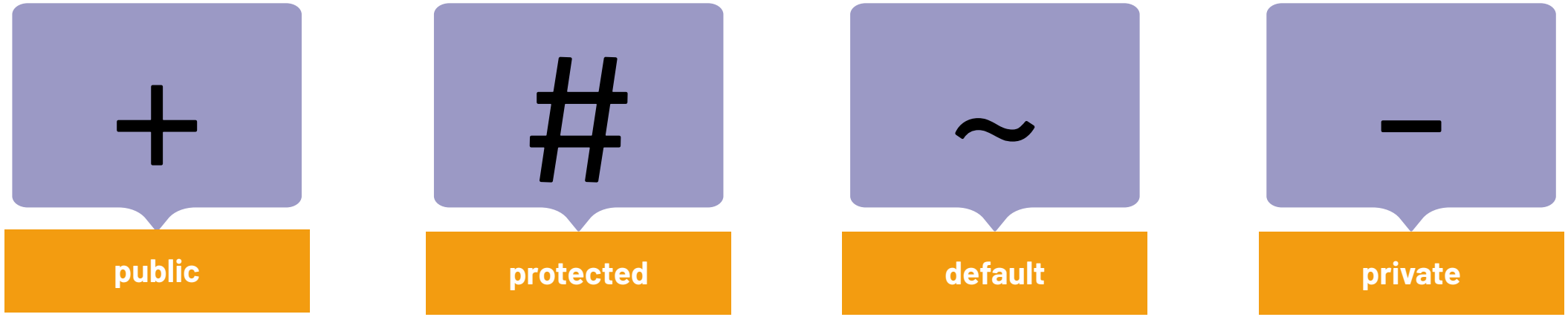
COMPONENTES BÁSICOS

- O diagrama de classes padrão é composto de três partes:
  - **Parte superior:** contém o nome da classe. Esta parte é sempre necessária, seja falando do classificador ou de um objeto.
  - **Parte do meio:** contém os atributos da classe. Use esta parte para descrever as qualidades da classe. É necessário somente quando se descreve uma instância específica de uma classe.
  - **Parte inferior:** inclui as operações da classe (métodos). Exibido em formato de lista, cada operação ocupa sua própria linha. As operações descrevem como uma classe interage com dados.



COMPONENTES BÁSICOS

- Todas as classes têm diferentes níveis de acesso, dependendo do modificador de acesso (visibilidade). Veja os níveis de acesso com seus símbolos correspondentes:

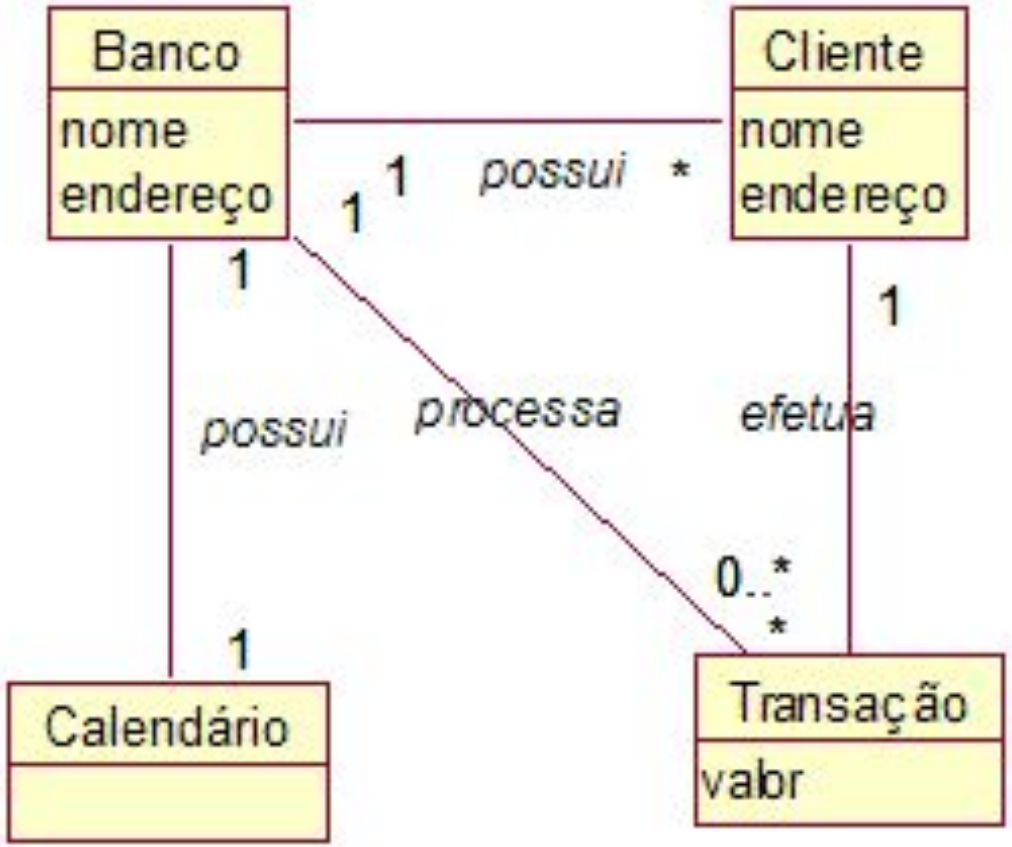


PERSPECTIVAS

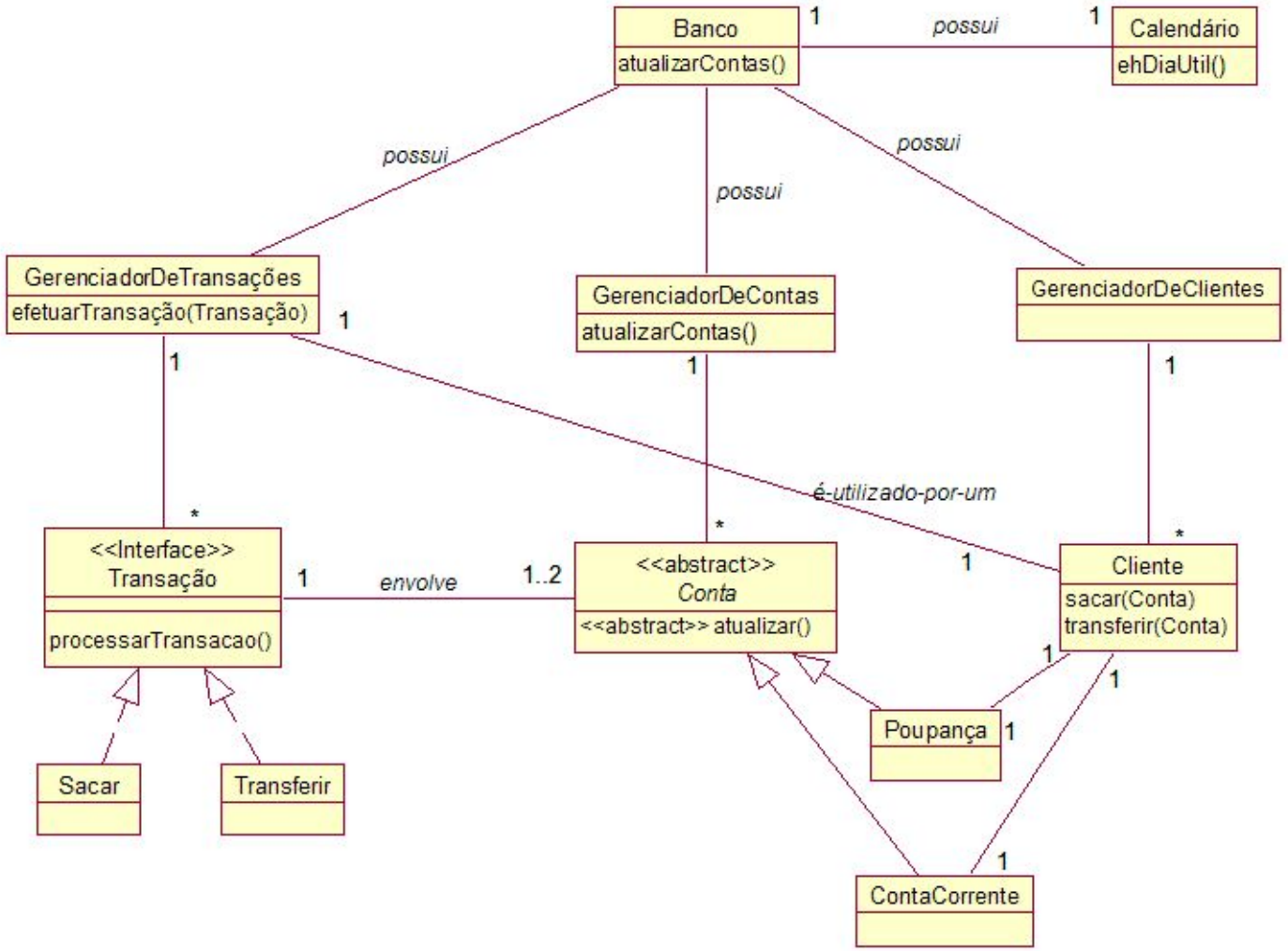
Um diagrama de classes pode oferecer três perspectivas, cada uma para um tipo de observador diferente. São elas:

- Conceitual
- Especificação
- Implementação - a mais utilizada de todas

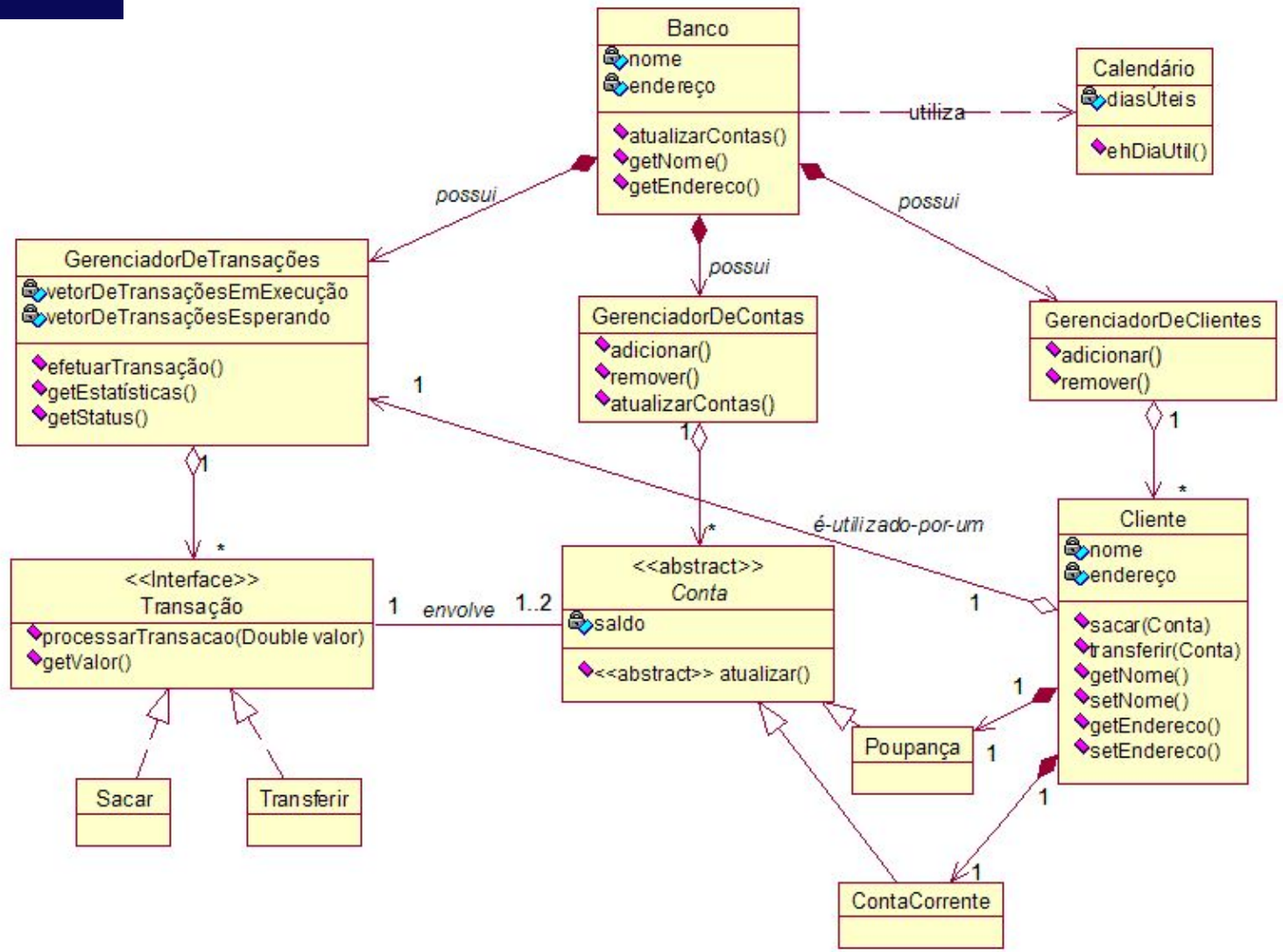
CONCEITUAL


Banco
nome
endereço
Cliente
nome
endereço
1
1
possui
*
1
1
possui
processa
efetua
0..*
*
1
Calendário
Transação
valor

ESPECIFICAÇÃO



IMPLEMENTAÇÃO

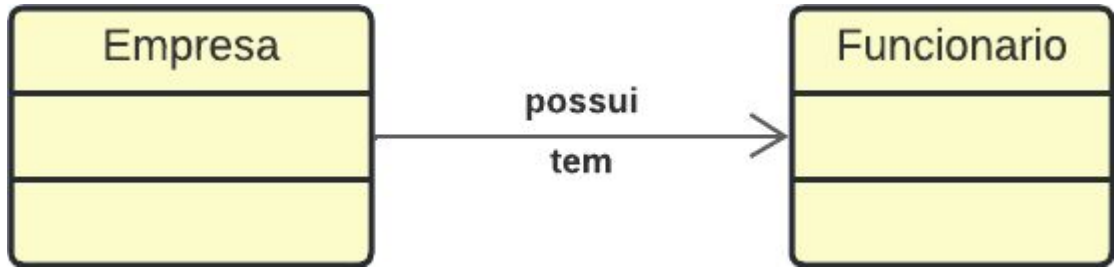


ASSOCIAÇÃO DE CLASSE

Entidades e Relacionamentos

ASSOCIAÇÃO DE CLASSE

- Descrevem um vínculo que ocorre entre os objetos de uma classe
- São representadas por uma linha que liga as classes envolvidas.
  - Algumas vezes, utiliza-se a seta

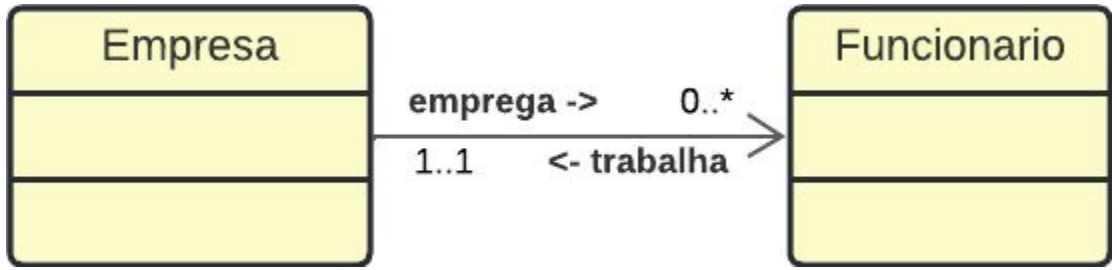


MULTIPLICIDADE

- Utilizado em todas as perspectivas de forma uniforme

Tipos	***Significa***
0..1	Zero ou uma instância. A notação n..m indica n para m instâncias.
0..* ou *	Não existe limite para o número de instâncias.
1	Exatamente uma instância.
1..*	Ao menos uma instância.

Exemplos:



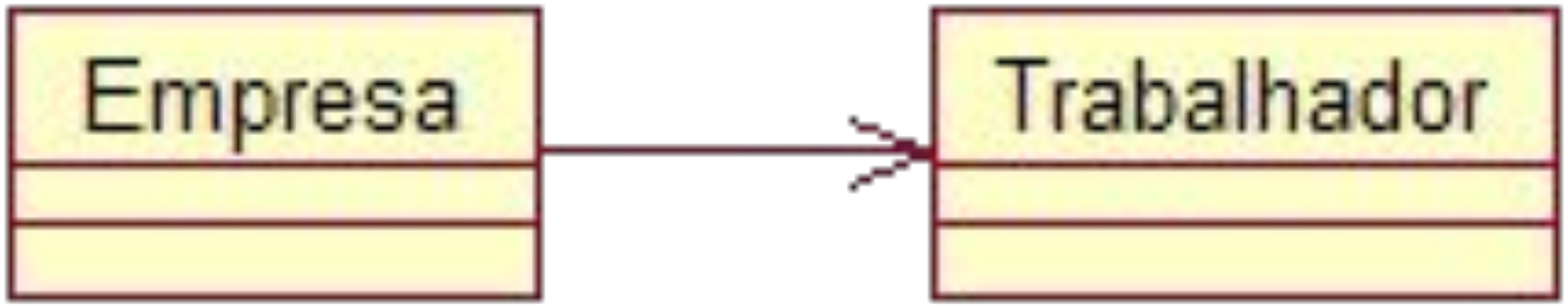
NAVEGABILIDADE

- utilizado apenas na perspectiva de implementação
- Um relacionamento sem navegabilidade implica que ele pode ser lido de duas formas, isto é, em suas duas direções. Ex.:

- Uma empresa possui um trabalhador, como também um trabalhador trabalha em uma empresa.

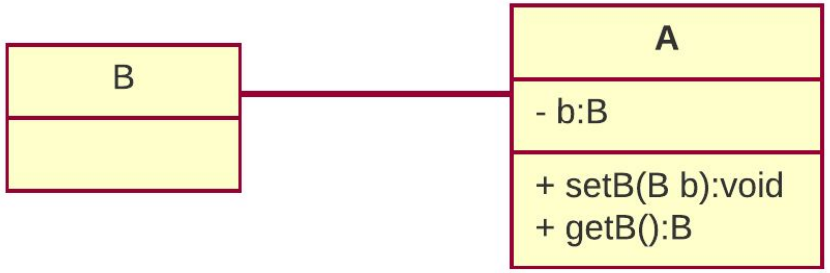
NAVEGABILIDADE

- Utilizando a propriedade de navegabilidade, podemos restringir a forma de ler um relacionamento. Isto é, em vez de termos duas direções, teremos apenas uma direção (de acordo com a direção da navegação). Ex.:



- Uma empresa possui um trabalhador.

EXEMPLO




```java
public class A {
    private B b;
    public A( ){
    }
    public void setB( B b ){
         this.b = b;
    }
    public B getB( ) {
        return b;
    }
}

public class B {
    public B( ){
    }
}
```

Exercícios

Exercício 1

- Crie uma diagrama de classe contendo a classe Funcionario com os seguintes atributos: cpf, nome e dependente do tipo de uma classe Dependente.

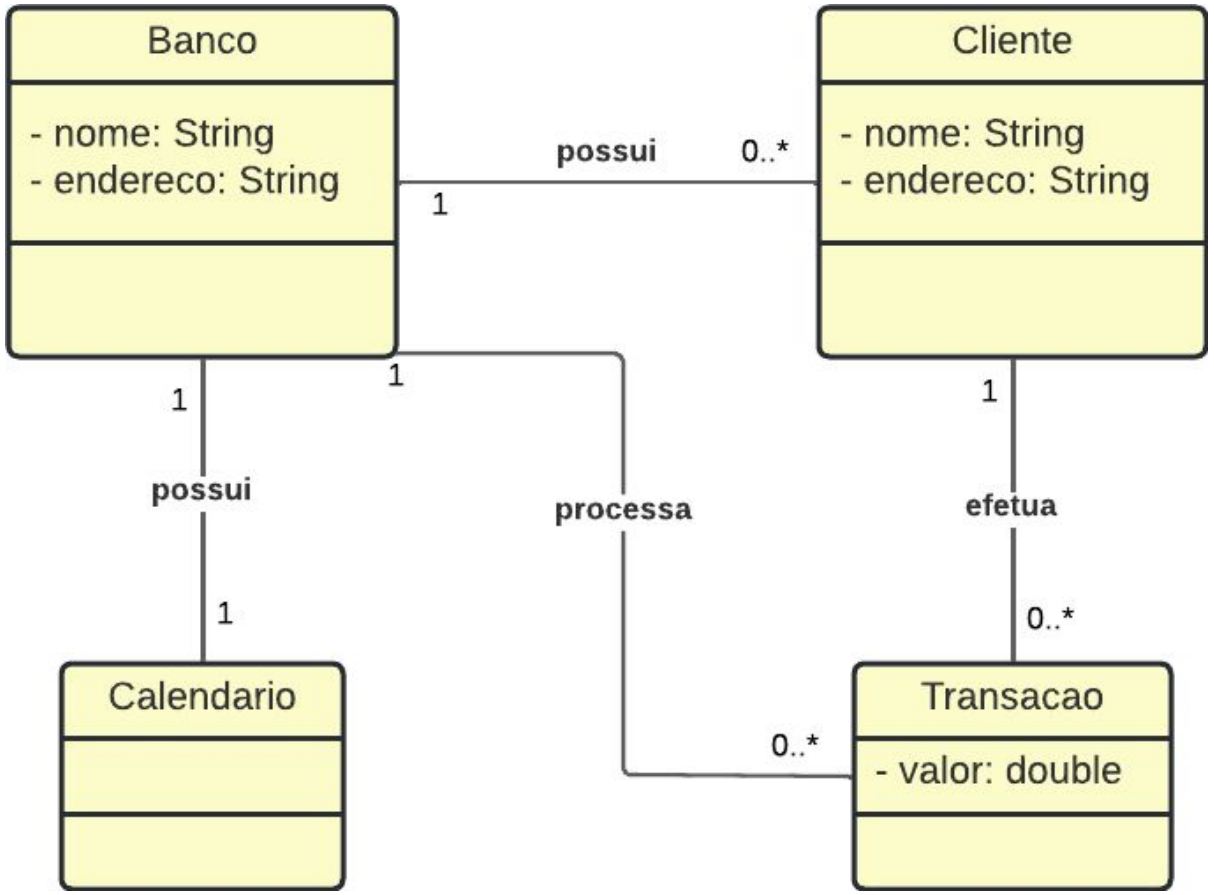
- Crie uma classe Dependente a qual se relaciona com a classe Funcionario

Exercício 2

- A partir do exercício anterior crie o código correspondente ao diagrama feito.

Exercício 3

- Crie um código correspondente ao diagrama a seguir:



C . E . S . A . R

school

Dúvidas? Entre em contato com a gente:

grs@cesar.school