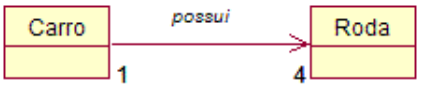
Relacionamentos

- Papel

  Descreve o relacionamento.



- Multiplicidade (utilizado em todas as perspectivas de forma uniforme)

  Notações possíveis

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:

- Associação (utilizado em todas as perspectivas)

  Representação Gráfica

  Associação

  Perspectiva:

  - Conceitual

    Define um relacionamento entre duas entidades conceituais do sistema.

  - Especificação

    Define responsabilidades entre duas classes. Implica que existem métodos que tratam desta responsabilidade.

  - Implementação

    Permite saber quem está apontando para quem, através da representação gráfica da navegabilidade. Além disto, é possível compreender melhor de que lado está a responsabilidade.

```
public class A {
```

```
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( ){
    }
}
```

- Herança ou Generalização (utilizado em todas as perspectivas)

  Representação Gráfica

  Perspectiva:

  Seja B uma generalização (extensão) de A.

  - Conceitual

    Considera que B é um subtipo ou um tipo especial de A. O que é válido para A, também é válido para B.

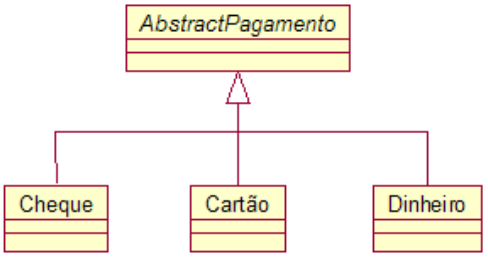
  - Especificação

    Ocorre uma herança de interface.

  - Implementação

    Ocorre uma herança de implementação.

  Exemplo de uma herança de implementação:

- 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.

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.

- Agregação (utilizado apenas na perspectiva de implementação)

  Definição

  Agregação é uma associação em que um objeto é parte de outro, de tal forma que a parte pode existir sem o todo.

  Em mais baixo nível, uma agregação consiste de um objeto contendo referências para outros objetos, de tal forma que o primeiro seja o todo, e que os objetos referenciados sejam as partes do todo.

  A diferença entre os relacionamentos de associação e agregação ainda é algo de bastante discussão entre os gurus. De forma geral, utiliza-se agregação para enfatizar detalhes de uma futura implementação (perspectiva de implementação).

  Representação gráfica

  Agregação com navegabilidade


```
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( ){
    }
}
```

- Composição (utilizado apenas na perspectiva de implementação)

  Definição

  Em mais baixo nível, em termos de passagem por parâmetro, seria uma *passagem por valor*. Enquanto que agregação seria uma *passagem por referência*.

  O todo *contém* as partes (e não *referências* para as partes). Quando o todo desaparece, todas as partes também desaparecem.

  Representação Gráfica


```
public class A {
    private B b;
    public A( ){
        b = new B();
    }
}
```

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

- Implementação (utilizado apenas na perspectiva de implementação)

  Em Inglês: *realization*

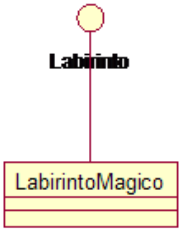
  Definição

  Utilizado para indicar que uma classe implementa uma interface

  Representação Gráfica

  Exemplo

  Implementação de uma interface representada por um círculo



  Implementação de uma interface representada por um retângulo