Unified Modeling Language (UML)

Universidade Federal do Maranhão – UFMA
Pós Graduação de Engenharia de Eletricidade
Grupo de Computação
Assunto: Diagrama de Classes

Autoria:Aristófanes Corrêa Silva
Adaptação: Alexandre César M de Oliveira

3 Diagrama de Classes

3.1 Definições

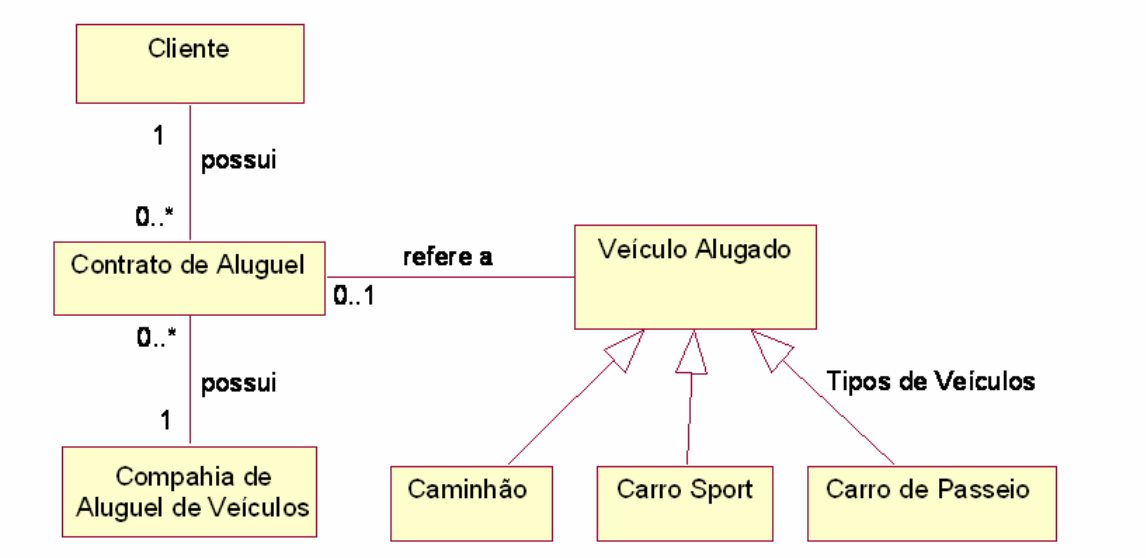
- Uma classe é a descrição de um tipo de objeto.
- Todos os objetos são instâncias de classes, onde a classe descreve as propriedades e comportamentos daquele objeto.
- Identificar as classes de um sistema pode ser complicado, e deve ser feito por *experts* no domínio do problema a que o software modelado se baseia.
- O diagrama de classes é considerado estático já que a estrutura descrita é sempre válida durante o ciclo de vida do sistema.
- Um sistema normalmente possui alguns diagramas de classes, já que não são todas as classes que estão inseridas em um único diagrama e uma certa classe pode participar de vários diagramas de classes.
- As classes devem ser extraídas do domínio do problema e serem nomeadas pelo que elas representam no sistema.

3.2 Relacionamentos

- Classes podem se relacionar com outras através de diversas maneiras: associação (conectadas entre si), dependência (uma classe depende ou usa outra classe), especialização (uma classe é uma especialização de outra classe), ou em pacotes (classes agrupadas por características similares).

- Os relacionamentos são mostrados no diagrama de classes juntamente com as suas estruturas internas, que são os atributos e operações.

- Exemplo



3.3 Propósitos Básicos e Distintos

- Fazer modelagem do vocabulário do sistema
  - Indica quais abstrações fazem parte do sistema e quais estão fora do limite do mesmo
- Fazer a modelagem e colaboração simples
  - Mostra como as classes trabalham em conjunto permitindo a compreensão de uma semântica maior do que se estas mesmas classes fossem analisadas individualmente.
- Fazer a modelagem do esquema lógico de um banco de dados
  - Descreve, de forma orientada a objetos, o esquema lógico de um banco de dados que fisicamente pode ser orientado a objetos, relacional ou híbrido.

3.4 Identificação de classes

- Questões que podem ajudar a identificá-las:
  - Existem informações que devem ser armazenadas ou analisadas? Se existir alguma informação que tenha que ser guardada, transformada ou analisada de alguma forma, então é uma possível candidata para ser uma classe.

- Existem sistemas externos ao modelado? Se existir, eles deverão ser vistos como classes pelo sistema para que possa interagir com outros externos.

- Existem classes de bibliotecas, componentes ou modelos externos a serem utilizados pelo sistema modelado? Se sim, normalmente essas classes, componentes e modelos conterão classes candidatas ao nosso sistema.

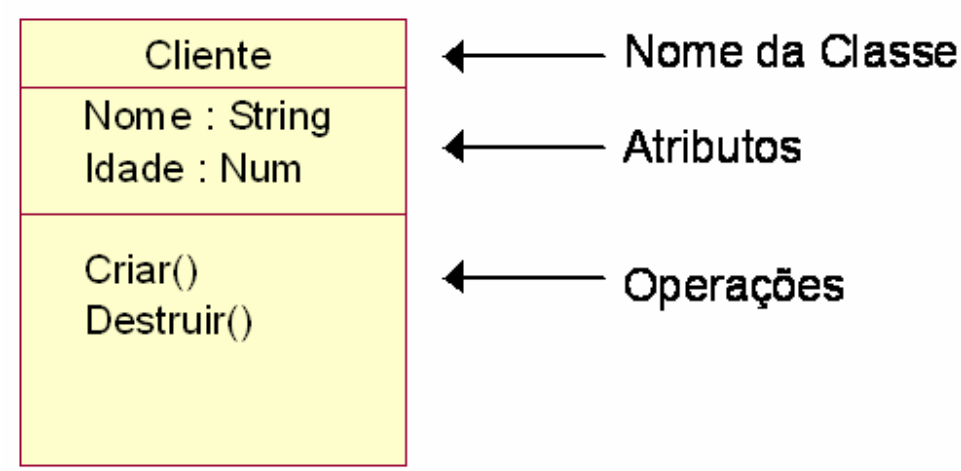
- Qual o papel dos atores dentro do sistema? Talvez o papel deles possa ser visto como classes, por exemplo, usuário, operador, cliente e daí por diante.

3.5 Representação

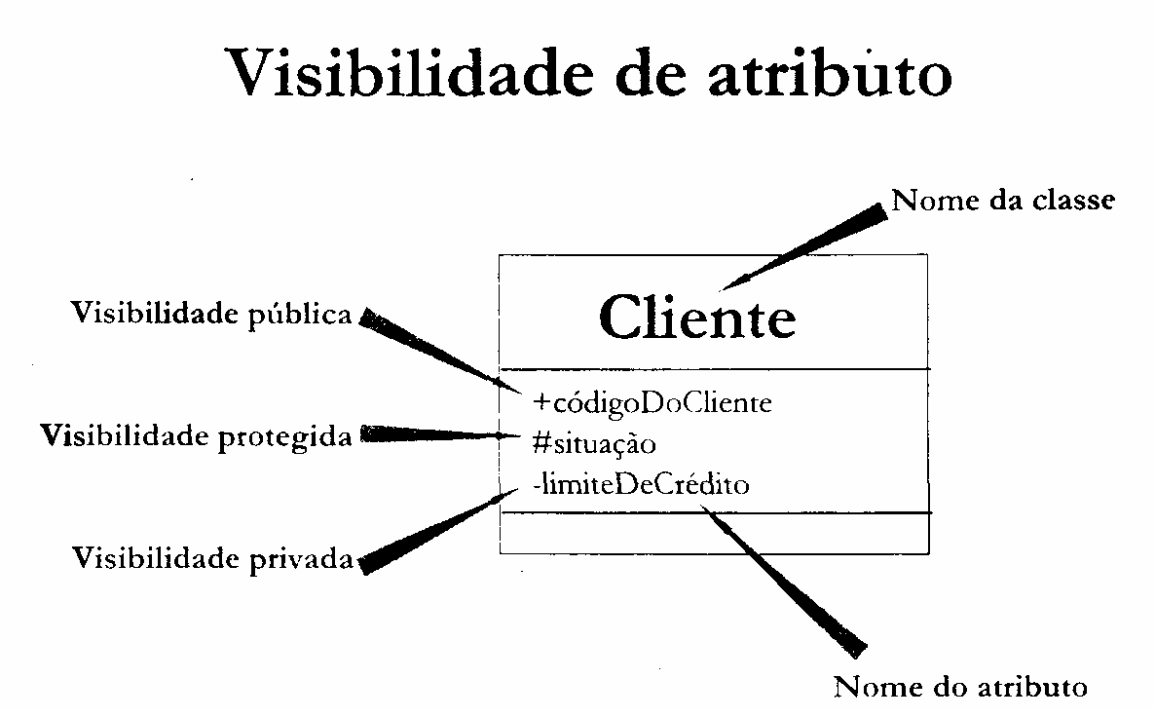
- Em UML, as classes são representadas por um retângulo dividido em três compartimentos: o compartimento de nome, que conterá apenas o nome da classe modelada, o de atributos, que possuirá a relação de atributos que a classe possui em sua estrutura interna, e o compartimento de operações, que serão os métodos de manipulação de dados e de comunicação de uma classe com outras do sistema.

- A sintaxe usada em cada um destes compartimentos é independente de qualquer linguagem de programação, embora podem ser usadas outras sintaxes como a do C++, Java, e etc.

- Representação Gráfica



- Visibilidade



3.6 Tipos de relacionamentos

- Os relacionamentos ligam as classes/objetos entre si criando relações lógicas entre estas entidades. Os relacionamentos podem ser dos seguintes tipos:
  - **Associação**: É uma conexão entre classes que também significa uma conexão entre objetos daquelas classes. Em UML, uma associação é definida com um relacionamento que descreve uma série de ligações, onde a ligação é definida como a semântica entre as duplas de objetos ligados.
  - **Generalização**: É um relacionamento de um elemento mais geral e outro mais específico. O elemento mais específico pode conter apenas informações adicionais. Uma instância (ou objeto) do elemento mais específico pode ser usada onde o elemento mais geral seja permitido.
  - **Dependência e Refinamentos**: Dependência é um relacionamento entre elementos, um independente e outro dependente. Uma modificação de um elemento independente afetará diretamente elementos dependentes dele. Refinamento é um relacionamento entre duas descrições de uma mesma entidade, mas em níveis diferentes de abstração.

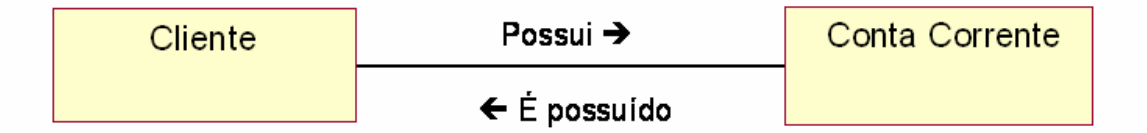
3.7 Associações

- Uma associação representa que duas classes possuem uma ligação (link) entre elas, significando, por exemplo, que elas "conhecem uma a outra", "estão conectadas com", "para cada X existe um Y" e assim por diante.
- Podem ser:
  - Normal, Recursiva, Qualificada, Exclusiva, Ordenada, Classe, Ternária e Agregação.

- **Associação Normal**
  - O tipo mais comum de associação é apenas uma conexão entre classes.
  - É representada por uma linha sólida entre duas classes. A associação possui um nome (junto à linha que representa a associação), normalmente um verbo, mas substantivos também são permitidos.
  - Pode-se também colocar uma seta no final da associação indicando que esta só pode ser usada para o lado onde a seta aponta. Mas associações também podem possuir dois nomes, significando um nome para cada sentido da associação.

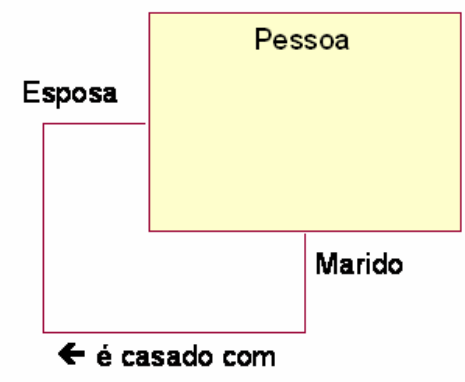
- Para expressar a multiplicidade entre os relacionamentos, um intervalo indica quantos objetos estão relacionados no link. O intervalo pode ser de zero para um (0..1), zero para vários (0..* ou apenas *), um para vários (1..*), dois (2), cinco para 11 (5..11) e assim por diante. É também possível expressar uma série de números como (1, 4, 6..12). Se não for descrita nenhuma multiplicidade, então é considerado o padrão de um para um (1..1 ou apenas 1).

- Exemplo: um relacionamento entre as classes Cliente e Conta Corrente que se relacionam por associação.



- **Associação Recursiva**

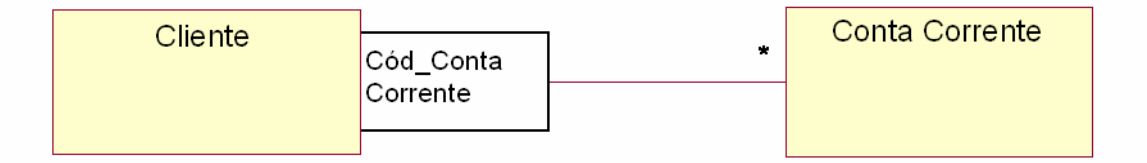
- Quando for necessário conectar uma classe a ela mesma através de uma associação e que ainda representa semanticamente a conexão entre dois objetos da mesma classe.



- **Associação Qualificada**

  - Associações qualificadas são usadas com associações de um para vários (1..*) ou vários para vários (*). O "qualificador" (identificador da associação qualificada) especifica como um determinado objeto no final da associação "n" é identificado, e pode ser visto como um tipo de chave para separar todos os objetos na associação.

  - O identificador é desenhado como uma pequena caixa no final da associação junto à classe de onde a navegação deve ser feita.



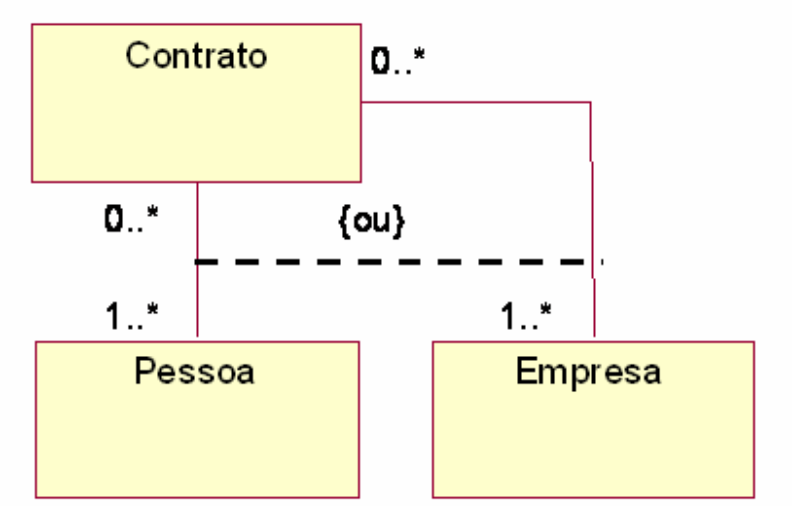
- **Associação Exclusiva**

  - Em alguns modelos nem todas as combinações são válidas, e isto pode causar problemas que devem ser tratados.

  - Uma associação exclusiva é uma restrição em duas ou mais associações.

  - Ela especifica que objetos de uma classe podem participar de no máximo uma das associações em um dado momento.

- Uma associação exclusiva é representada por uma linha tracejada entre as associações que são partes da associação exclusiva, com a especificação "{ou}" sobre a linha tracejada.
- Exemplo: No diagrama abaixo um contrato não pode se referir a uma pessoa e a uma empresa ao mesmo tempo, significando que o relacionamento é exclusivo a somente uma das duas classes.



- **Associação Ordenada**
  - As associações entre objetos podem ter uma ordem implícita.
  - O padrão para uma associação é desordenado (ou sem nenhuma ordem específica). Mas uma ordem pode ser especificada através da associação ordenada.

- Este tipo de associação pode ser muito útil em casos como este: janelas de um sistema têm que ser ordenadas na tela (uma está no topo, uma está no fundo e assim por diante).

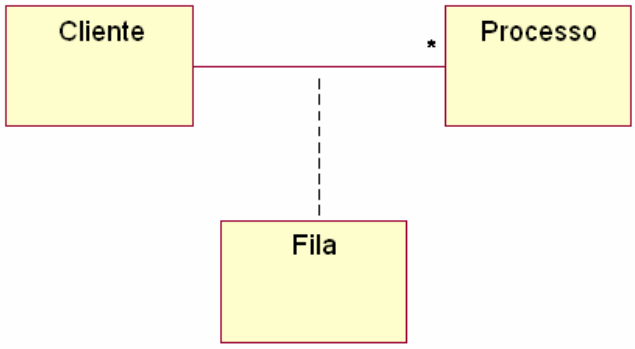
- A associação ordenada pode ser escrita apenas colocando "{ordenada}" junto à linha de associação entre as duas classes.

- **Associação de Classe**

  - Uma classe pode ser associada a uma outra associação, não sendo conectada nenhuma das extremidades da associação já existente, mas na própria linha da associação.

  - Esta associação serve para se adicionar informação extra a associação já existente.

  - Exemplo: A associação da classe Fila com a associação das classes Cliente e Processo pode ser estendida com operações de adicionar processos na fila, para ler e remover da fila e de ler o seu tamanho. Se operações ou atributos são adicionados a associação, ela deve ser mostrada como uma classe.

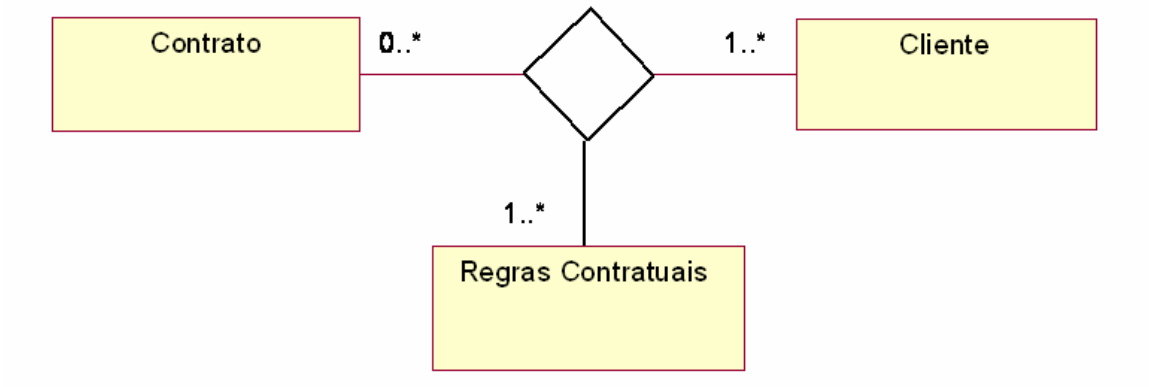


- **Associação de Ternária**

  - Mais de duas classes podem ser associadas entre si, a associação ternária associa três classes.

  - Ela é mostrada como uma grade losango (diamante) e ainda suporta uma associação de classe ligada a ela, traçar-se-ia, então, uma linha tracejada a partir do losango para a classe onde seria feita a associação ternária.

  - Exemplo: A associação ternária especifica que um cliente poderá possuir 1 ou mais contratos e cada contrato será composto de 1 ou várias regras contratuais.



- **Associação de Agregação**
  - A agregação é um caso particular da associação.
  - A agregação indica que uma das classes do relacionamento é uma parte, ou está contida em outra classe. As palavras chaves usadas para identificar uma agregação são: "consiste em", "contém", "é parte de".

  - Existem tipos especiais de agregação que são as agregações compartilhadas e as compostas.
  - Agregação Compartilhada:
    - É dita compartilhada quando uma das classes é uma parte, ou está contida na outra, mas esta parte pode estar contida na outras várias vezes em um mesmo momento.
    - Exemplo: uma pessoa pode ser membro de um time ou vários times e em determinado momento.

3.8 Generalização

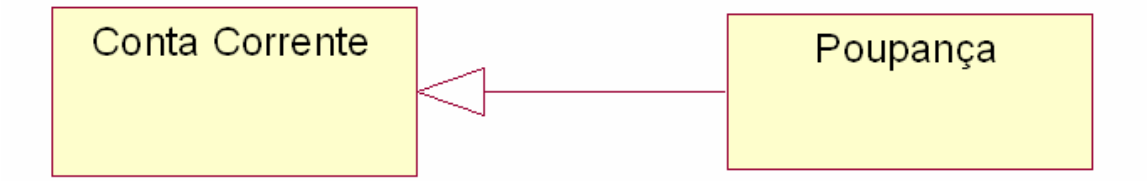
- A generalização é um relacionamento entre um elemento geral e um outro mais específico.
- O elemento mais específico possui todas as características do elemento geral e contém ainda mais particularidades.
- Um objeto mais específico pode ser usado como uma instância do elemento mais geral.
- A generalização, também chamada de herança, permite a criação de elementos especializados em outros.
- Existem alguns tipos de generalizações que variam em sua utilização a partir da situação. São elas:
  - Normal
  - Restrita
    - Sobreposição
    - Disjuntiva
    - Completa
    - Incompleta.

- **Generalização Normal**

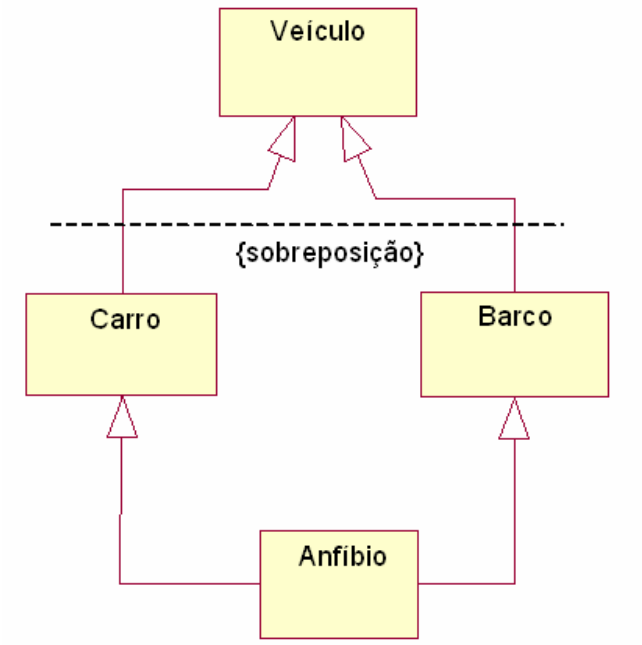
  - Na generalização normal a classe mais específica, chamada de subclasse, herda tudo da classe mais geral, chamada de superclasse. Os atributos, operações e todas as associações são herdados.

  - Uma classe pode ser tanto uma subclasse quanto uma superclasse, se ela estiver numa hierarquia de classes que é um gráfico onde as classes estão ligadas através de generalizações.

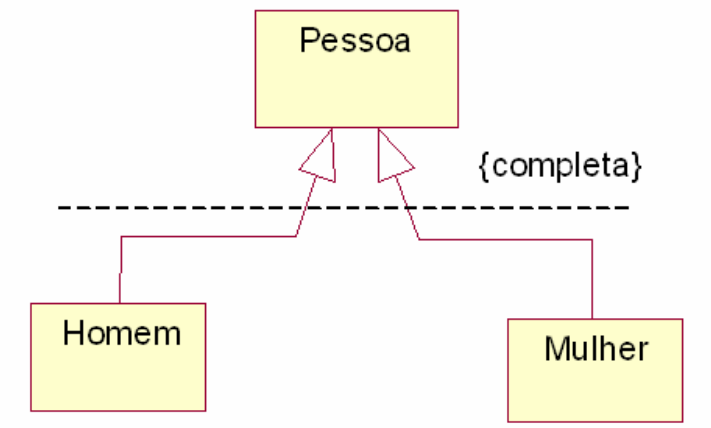
  - A generalização normal é representada por uma linha entre as duas classes que fazem o relacionamento, sendo que se coloca uma seta no lado da linha onde se encontra a superclasse indicando a generalização.



- **Generalização Restrita**
  - Uma restrição aplicada a uma generalização especifica informações mais precisas sobre como a generalização deve ser usada e estendida no futuro.
  - Generalizações restritas com mais de uma subclasse:
  - Sobreposição e Disjuntiva:
    - Generalização de sobreposição ocorre quando subclasses herdam de uma superclasse por sobreposição e novas subclasses destas podem herdar de mais de uma subclasse.
    - A generalização disjuntiva é exatamente o contrário da sobreposição e a generalização é utilizada como padrão.

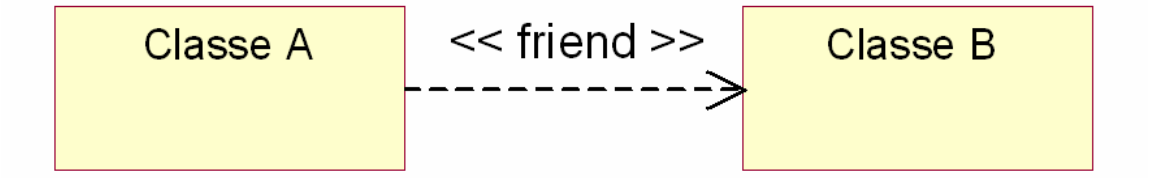


- Completa e Incompleta
  - Uma restrição simbolizando que uma generalização é completa significa que todas as subclasses já foram especificadas, e não existe mais possibilidade de outra generalização a partir daquele ponto.
  - A generalização incompleta é exatamente o contrário da completa e é assumida como padrão da linguagem.



3.9 Dependência e Refinamento

- Dependência é uma conexão semântica entre dois modelos de elementos, um independente e outro dependente.
- Uma mudança no elemento independente irá afetar o modelo dependente. Como no caso anterior com generalizações, os modelos de elementos podem ser uma classe, um pacote, um use-case e assim por diante.
- Quando uma classe recebe um objeto de outra classe como parâmetro, uma classe acessa o objeto global da outra. Nesse caso existe uma dependência entre estas duas classes, apesar de não ser explícita.
- Uma relação de dependência é simbolizada por uma linha tracejada com uma seta no final de um dos lados do relacionamento. E sobre essa linha o tipo de dependência que existe entre as duas classes.
- As classes "Amigas" provenientes do C++ são um exemplo de um relacionamento de dependência.



- Refinamentos são um tipo de relacionamento entre duas descrições de uma mesma coisa, mas em níveis de abstração diferentes, e podem ser usados para modelar diferentes implementações de uma mesma coisa (uma implementação simples e outra mais complexa, mas também mais eficiente).

- Os refinamentos são simbolizados por uma linha tracejada com um triângulo no final de um dos lados do relacionamento e são usados em modelos de coordenação.

- Em grandes projetos, todos os modelos que são feitos devem ser coordenados. Coordenação de modelos pode ser usada para mostrar modelos em diferentes níveis de abstração que se relacionam e mostram também como modelos em diferentes fases de desenvolvimento se relacionam.

3.10 Mecanismos gerais

- A UML utiliza alguns mecanismos em seus diagramas para tratar informações adicionais.
- **Ornamentos:**
  - São gráficos são anexados aos modelos de elementos em diagramas e adicionam semânticas ao elemento.
  - Um exemplo de um ornamento é o da técnica de separar um tipo de uma instância. Quando um elemento representa um tipo, seu nome é mostrado em negrito. Quando o mesmo elemento representa a instância de um tipo, seu nome é escrito sublinhado e pode significar tanto o nome da instância quanto o nome do tipo.
  - Outros ornamentos são os de especificação de multiplicidade de relacionamentos, onde a multiplicidade é um número ou um intervalo que indica quantas instâncias de um tipo conectado pode estar envolvido na relação.
- **Notas:**
  - Nem tudo pode ser definido em uma linguagem de modelagem, sem importar o quanto extensa ela seja.

- Para permitir adicionar informações a um modelo não poderia ser representado de outra forma, UML provê a capacidade de adicionar Notas.

- Uma Nota pode ser colocada em qualquer lugar em um diagrama, e pode conter qualquer tipo de informação.