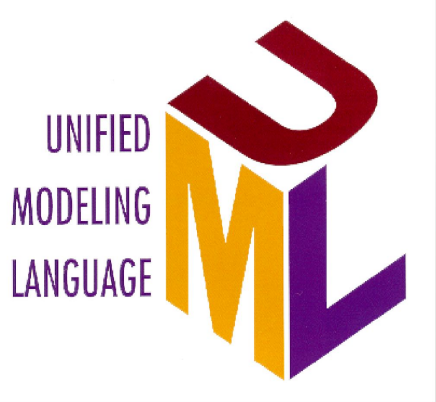
Engenharia de Software II

- **Diagrama de Classes**
- **Diagrama de Objetos**

Profa. Renata Neves



Engenharia de Software II

Diagrama de Classes

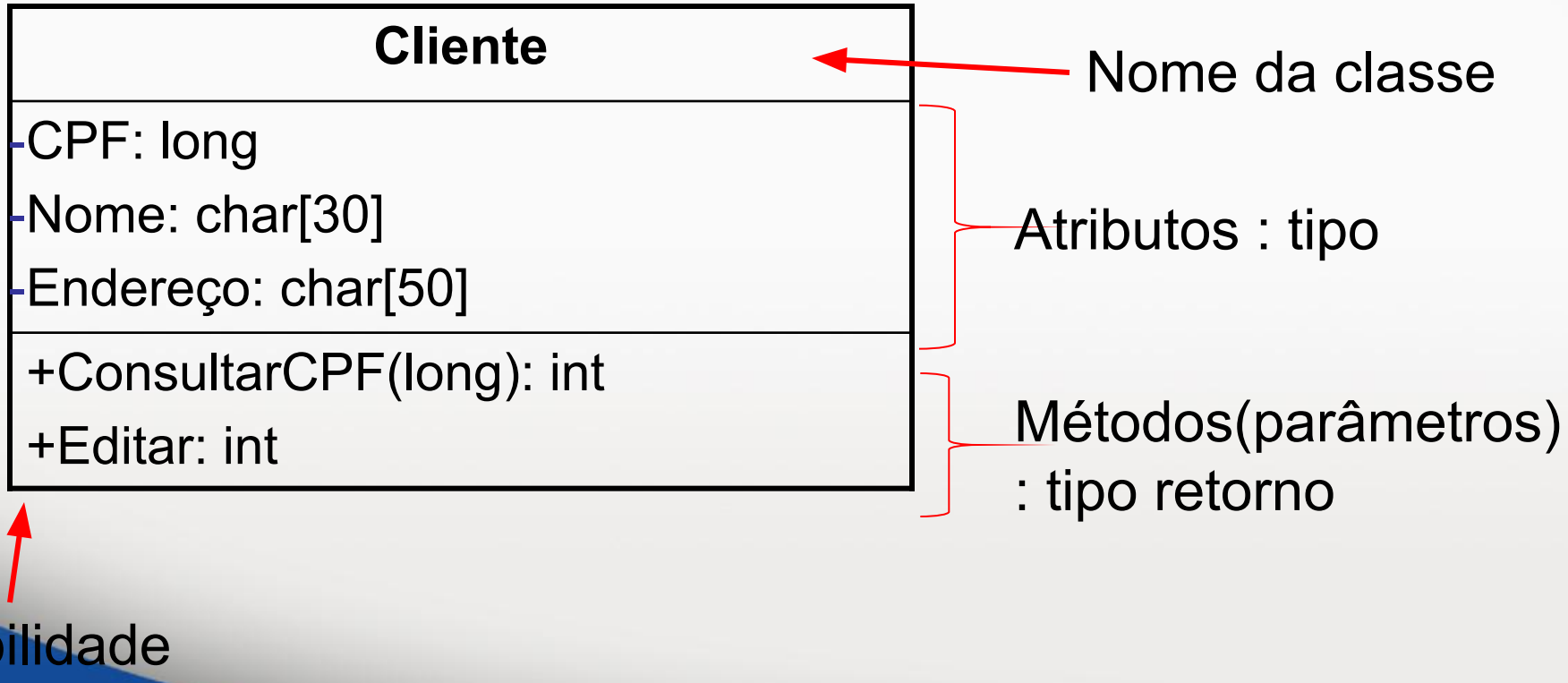
Função: Visualizar as classes que irão compor o sistema, seus atributos e respectivos métodos, bem como o relacionamento entre elas. Pode ser utilizado para modelar tabelas a serem salvas, sendo um avanço em relação ao Modelo Entidade-Relacionamento

Fornece:

- **Visualizar a comunicação entre as classes;**
- **Identificação da classe, atributos e seus tipos, métodos e parâmetros;**
- **Base para a modelagem de outros diagramas.**

Engenharia de Software II

Diagrama de Classes



Engenharia de Software II

Diagrama de Classes

Não é obrigatório que a classe apresente sempre 3 divisões, pois pode haver classes que não tenham atributos, ou que não tenham métodos.

Alguns atributos ou métodos podem ser suprimidos para não deixar o diagrama muito "poluído".

Engenharia de Software II

Orientação a objetos - Visibilidade

Visibilidade de atributos e métodos:

- Pública (+): pode ser utilizado por outras classes;
- Protegida (#): somente a classe possuidora e as classes derivadas da possuidora podem usar;
- Privado (-): somente a classe possuidora pode usar.

Engenharia de Software II

Relacionamentos entre classes

- **Associações:**
  - **Unária ou Reflexiva;**
  - **Binária;**
  - **Ternária ou N-ária;**
  - **Agregação;**
  - **Composição;**
- **Especialização /Generalização;**
- **Dependência;**
- **Realização.**

Engenharia de Software II

Associação

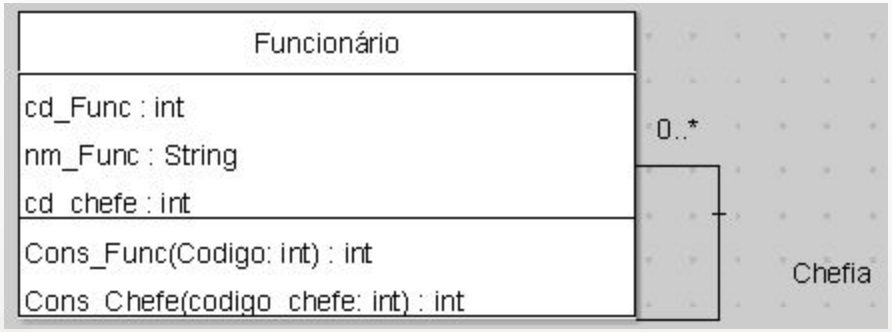
Definir como as classes estão unidas e interagem entre sí, compartilhando informações. É representada por uma linha ou seta (indicando o sentido da navegação).

Possuem um título para indicar o tipo de vínculo estabelecido pela associação.

Engenharia de Software II

Associação – Unária ou Reflexiva

Relacionamento da classe para consigo mesma.



Neste exemplo, uma instância da classe funcionário (um funcionário) pode ser o chefe, tendo o atributo cd_func e cd_chefe iguais.

A associação tem um nome (Chefia) e a indicação da multiplicidade (0..*), indicando que uma instância da classe funcionário pode se relacionar com nenhuma (0) ou com muitas (*) .

Engenharia de Software II

Associação

Multiplicidades:

0..1 O Relacionamento não é obrigatório (0), mas se existir terá apenas uma instância se relacionando com outra (1);

1..1 Um objeto se relaciona com somente um outro;

0..* Pode ou não haver relacionamento e se existir de 1 para n;

*** Muitos: de n para n**

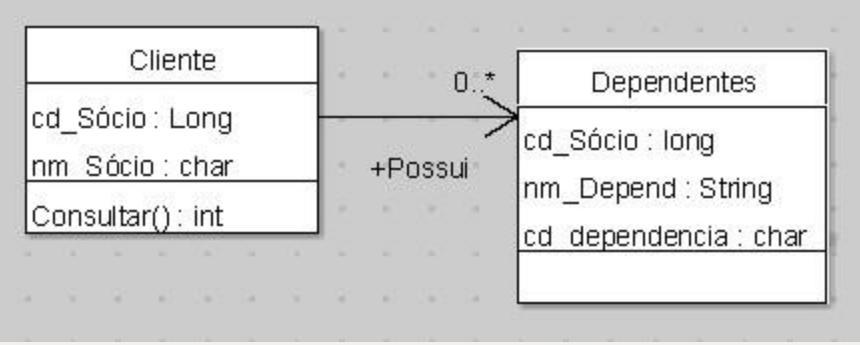
1..* Obrigatoriedade de relacionamento: no mínimo um e no máximo n;

3..5 Mínimo de 3 e máximo de 5

Engenharia de Software II

Associação – Binária

Relacionamento entre duas classes.



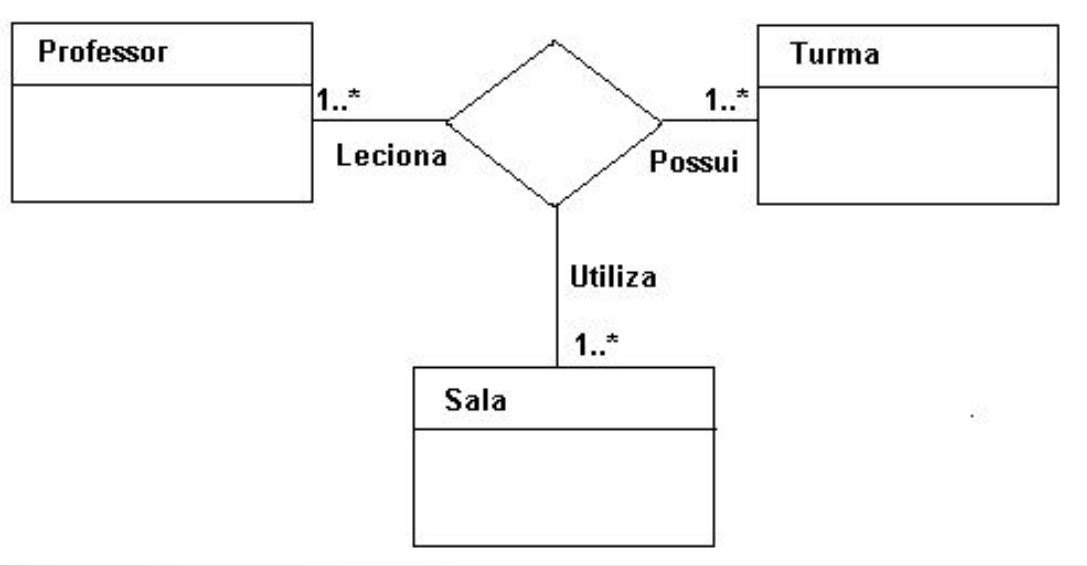
Uma instância da classe Cliente pode relacionar-se ou não com instâncias da classe Dependentes (0..*). Como do outro lado da associação não está declarado nada, dependentes só irá se relacionar

Com um objeto da classe Cliente e somente um. A seta e a descrição (possui) dão maiores indicações sobre a associação. Os atributos chaves (primária e estrangeira) podem ser abolidos do diagrama (implícitos).

Engenharia de Software II

Associação – Ternária ou N-ária

Relacionamento entre três ou mais classes.

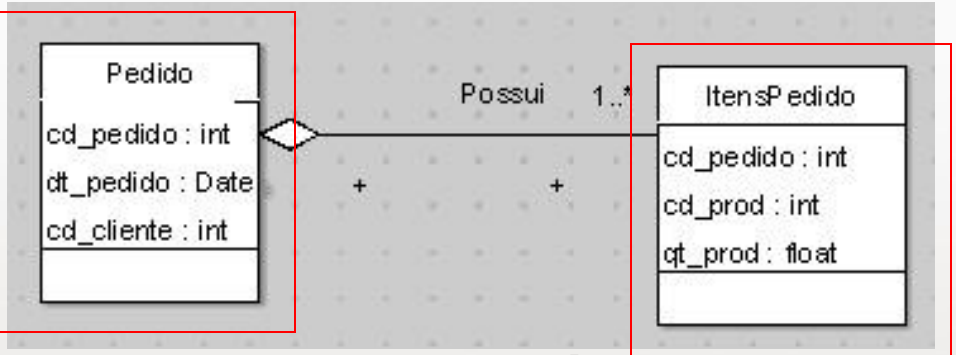


Engenharia de Software II

Associação – Agregação

Relacionamento entre duas ou mais classes, onde um objeto de uma classe (objeto-todo) precisa ser complementado por atributos de um ou mais objetos de outras classes (objetos-parte)

Objeto-todo



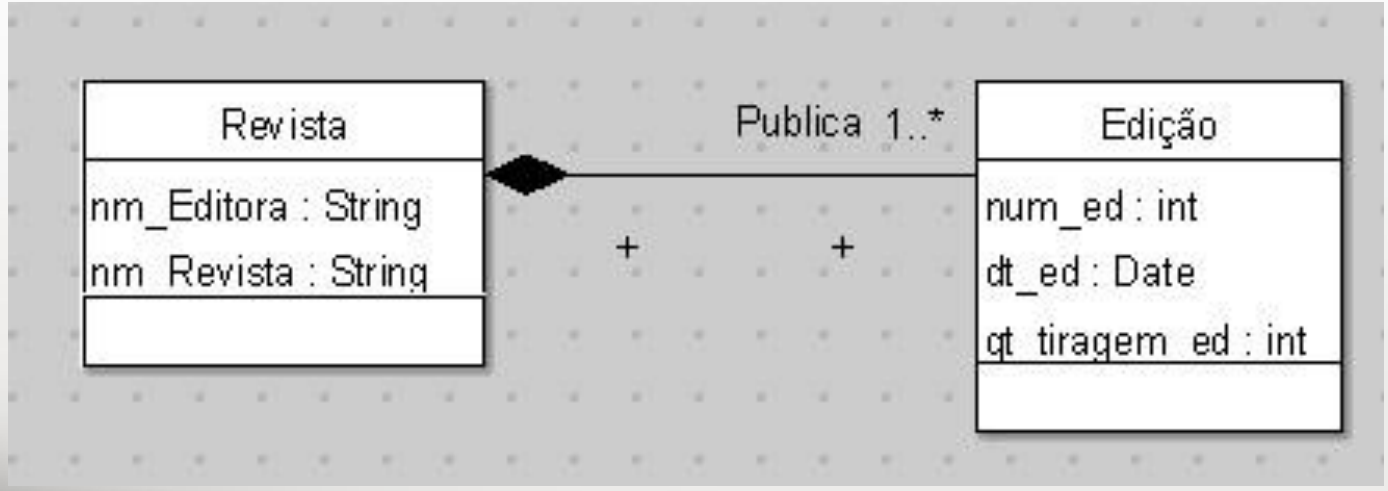
Objeto-parte

O objeto parte não pode ser destruído senão por métodos do objeto-todo (integridade referencial)

Engenharia de Software II

Associação – Composição

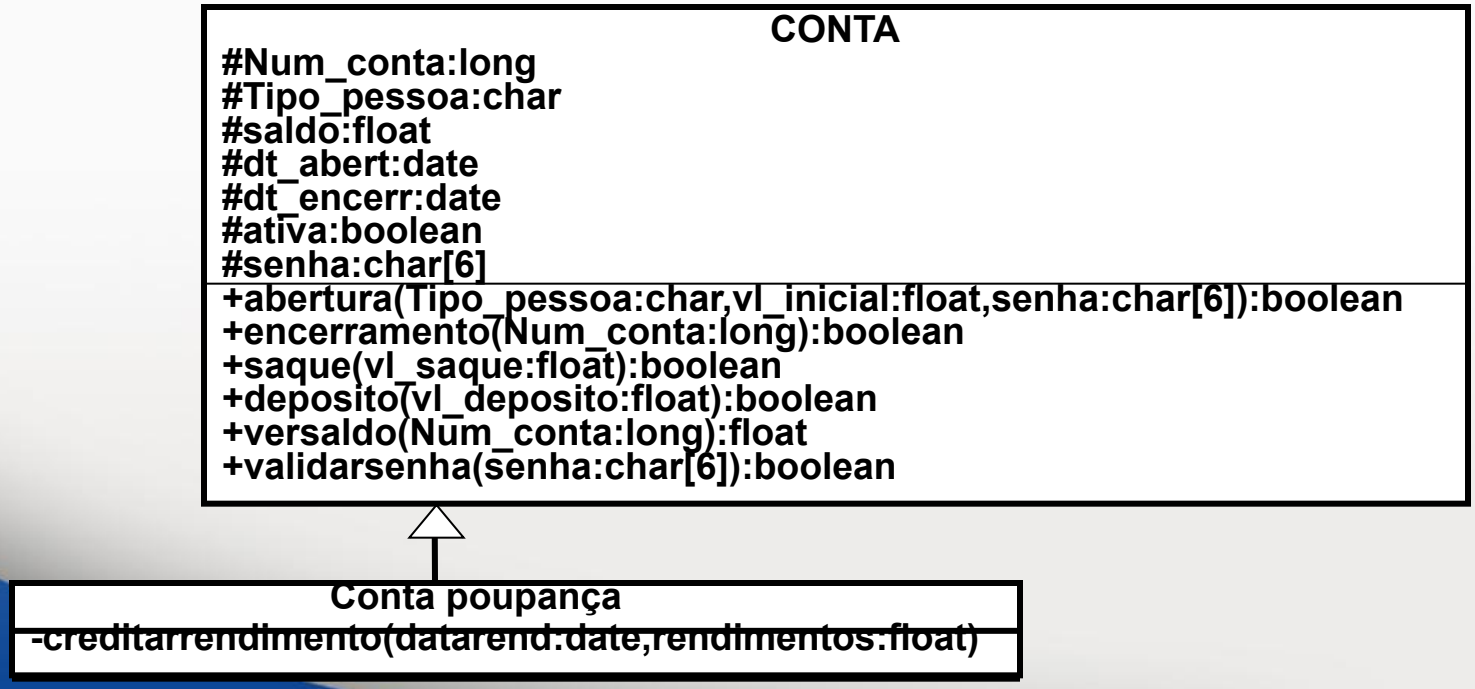
Semelhante à Agregação, porém o objeto-parte só poderá se relacionar exclusivamente com um único objeto-todo



Engenharia de Software II

Especialização/Generalização

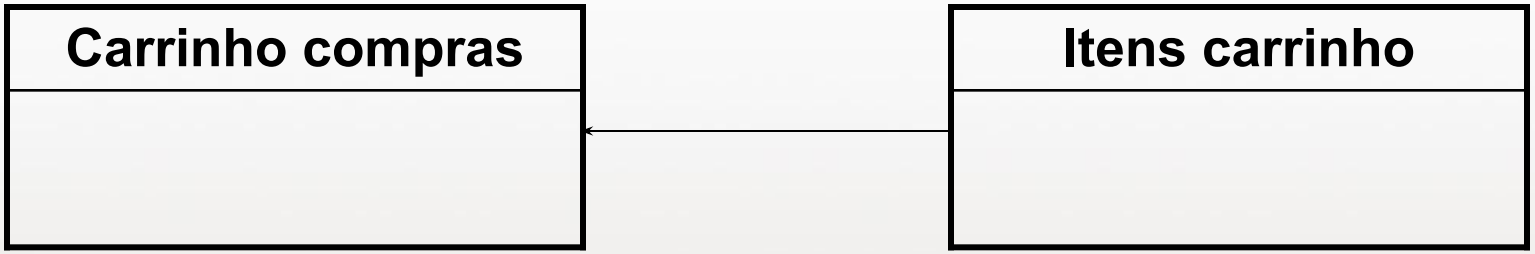
Igual ao diagrama de Casos de Uso: herança



Engenharia de Software II

Dependência

Pouco utilizado, indica uma dependência de uma classe em relação à outra: sempre que ocorre uma mudança na classe "tutora", esta mudança deve ser refletida na classe dependente

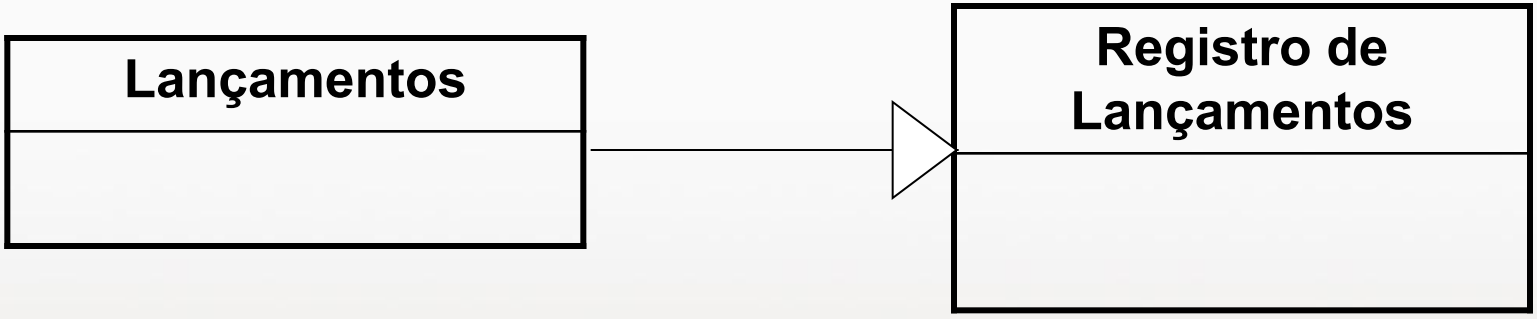


No exemplo acima, ocorrendo uma mudança na classe "Itens carrinho", ocorrerá uma mudança na classe "Carrinho compras

Engenharia de Software II

Realização

Uma classe realizará uma tarefa, a pedido de uma outra classe

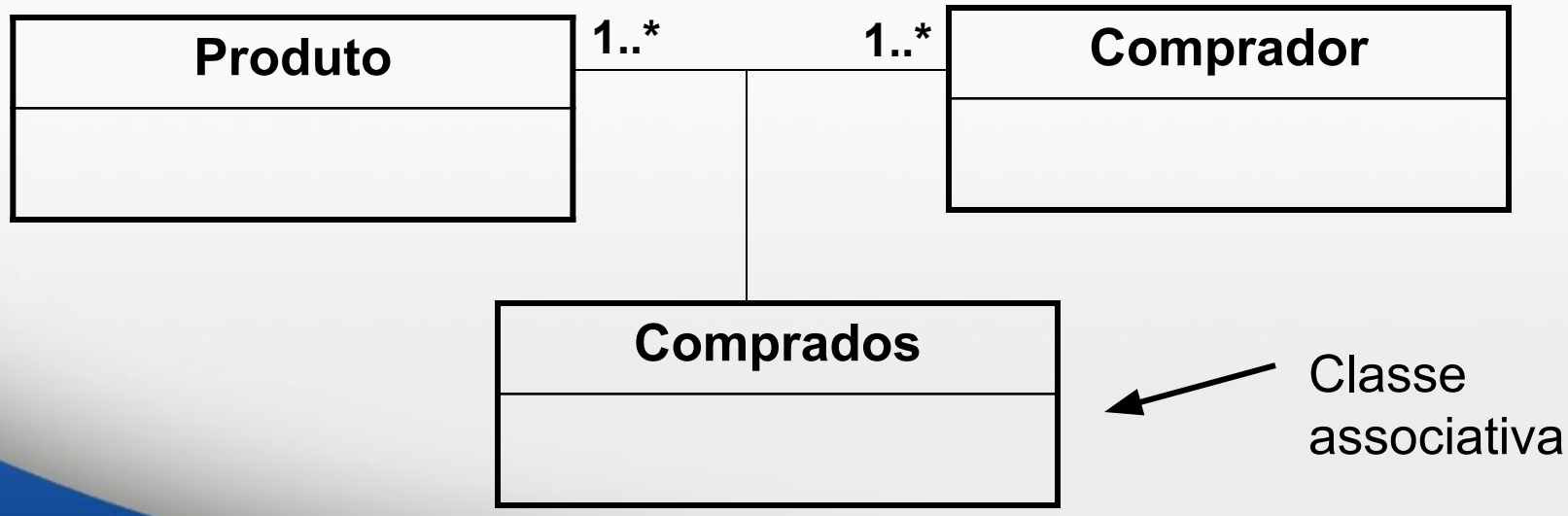


No exemplo acima, Lançamentos pode ser enxergado como uma interface para receber os dados e Registro de Lançamentos a classe que realmente efetuará um processo.

Engenharia de Software II

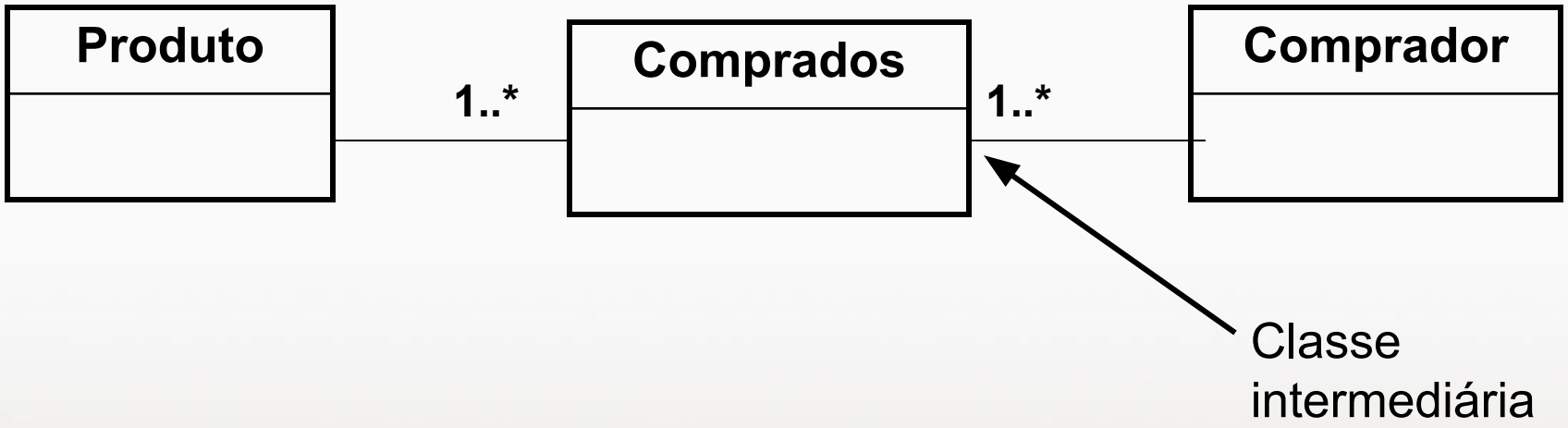
Classe Associativa

Classes produzidas por associações de multiplicidade * (muitos), onde não se pode armazenar em nenhuma das classes envolvidas, gerando um terceira classe.



Engenharia de Software II

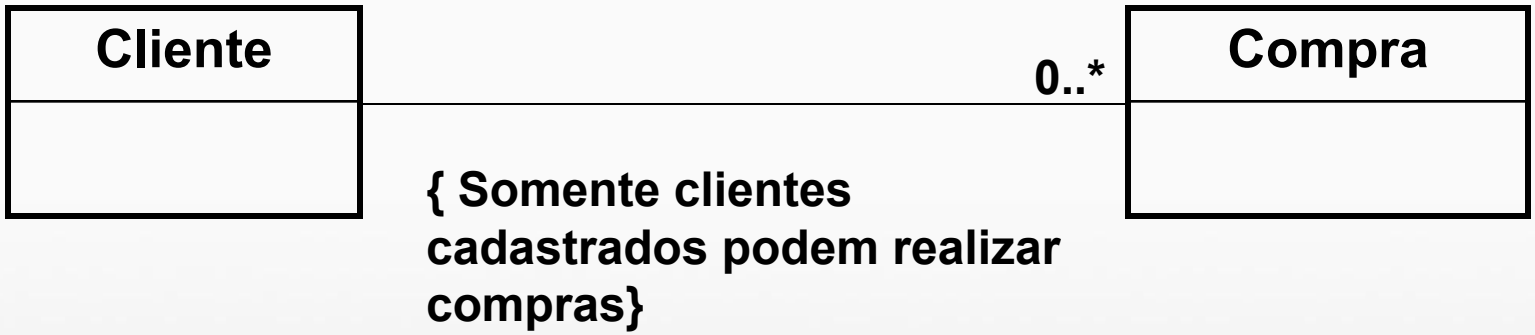
Classe Associativa



Os atributos chaves que mostram o relacionamento (os óbvios) não necessitam serem escritos.

Engenharia de Software II

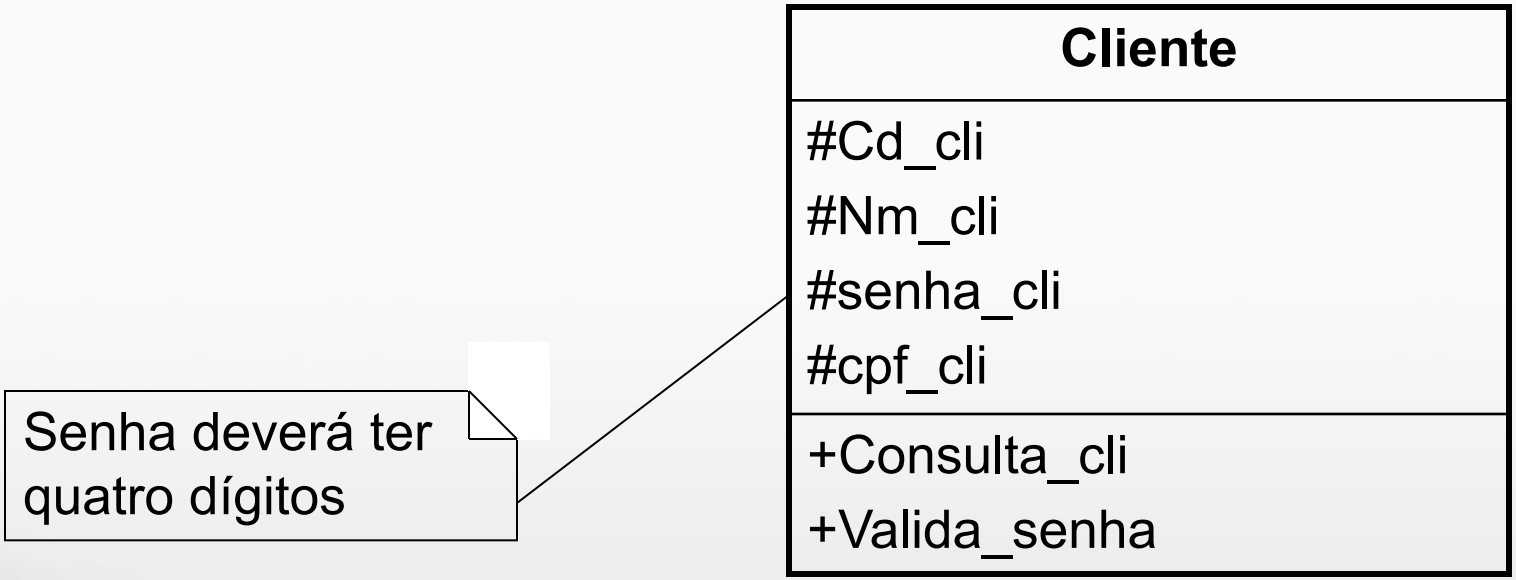
Restrição



Informações extras que definem restrições ou condições para o relacionamento (texto limitado por chaves)

Engenharia de Software II

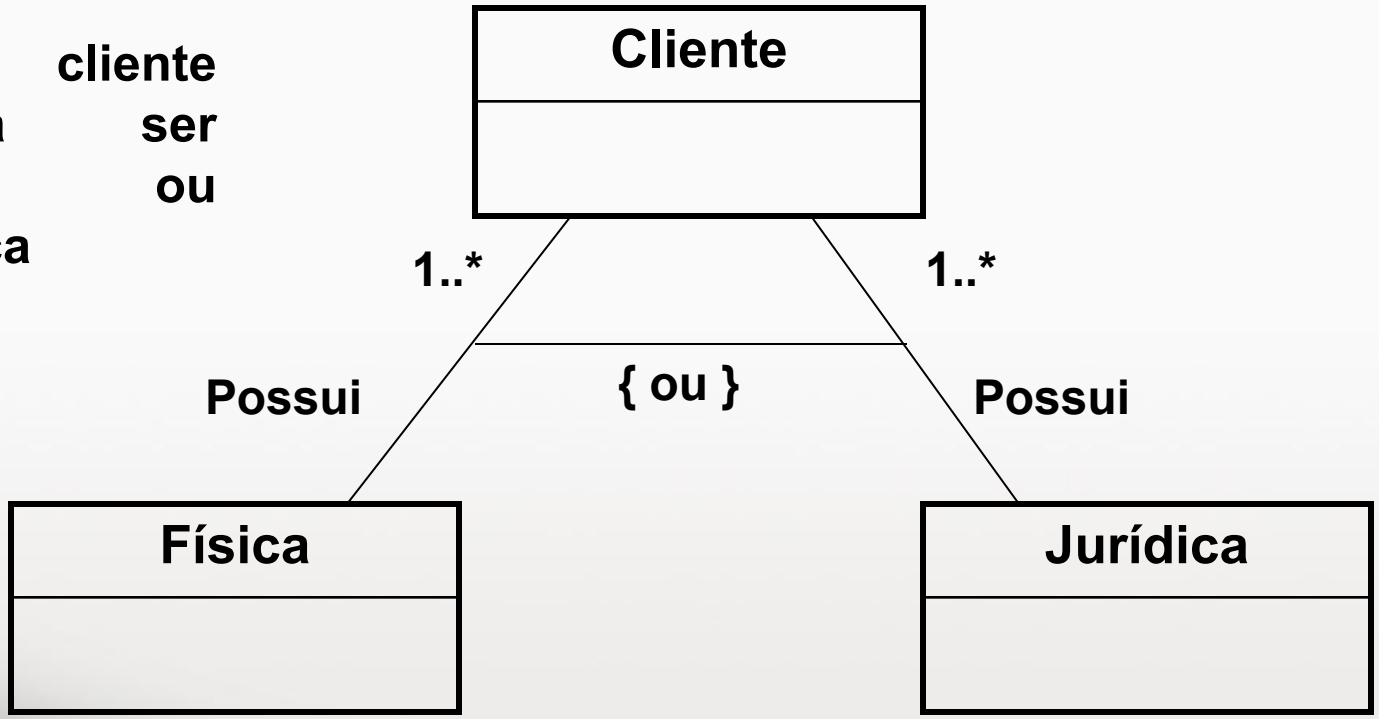
Restrição



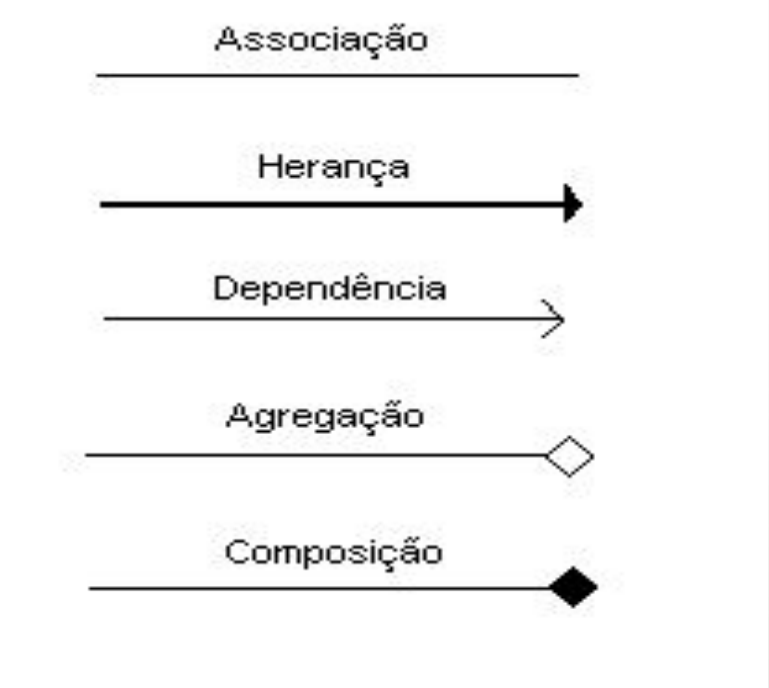
Engenharia de Software II

Restrição

Cada cliente poderá ser Física ou Jurídica



Engenharia de Software II



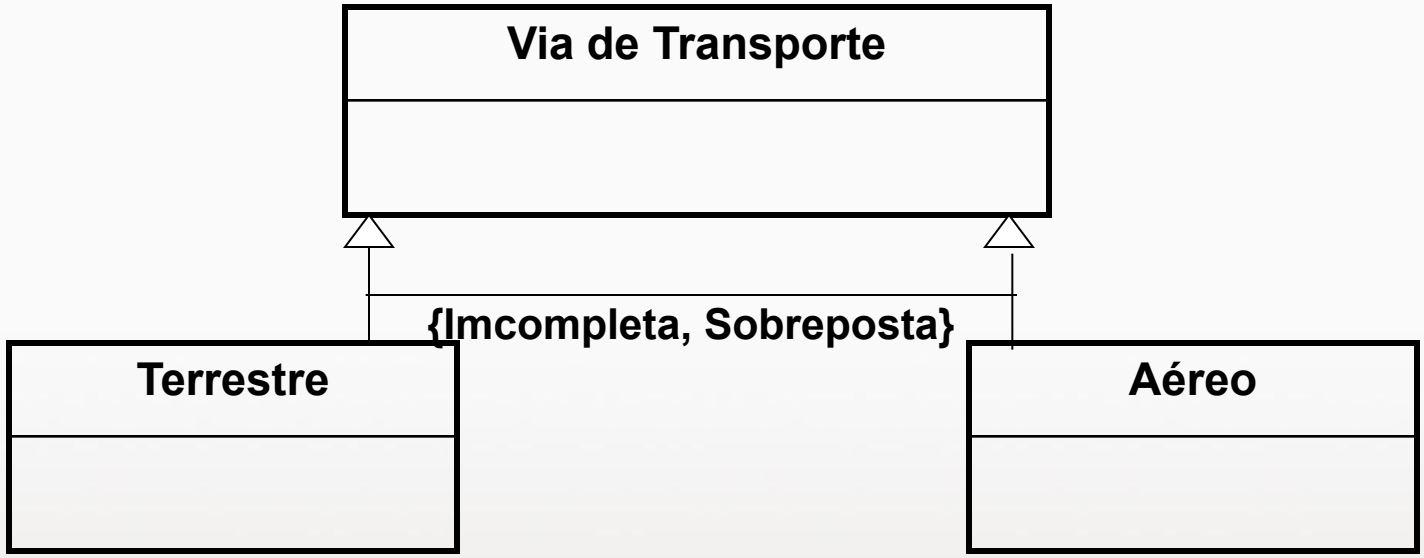
Engenharia de Software II

Restrição: definindo semântica das classes

- **Completa: as subclasses derivadas não podem derivar outras sub-classes;**
- **Incompleta: ainda é possível derivar;**
- **Separada ou disjunta: uma instância pertencerá a uma ou outra classe apenas;**
- **Sobreposta: uma instância poderá pertencer a mais de uma classe.**

Engenharia de Software II

Restrição: definindo semântica das classes



Incompleta: pode-se derivar de terrestre para trem ou caminhão, ou marítima, por exemplo;

Sobreposta: o transporte poderá se feito em parte por via aérea e em parte por terra.

Engenharia de Software II

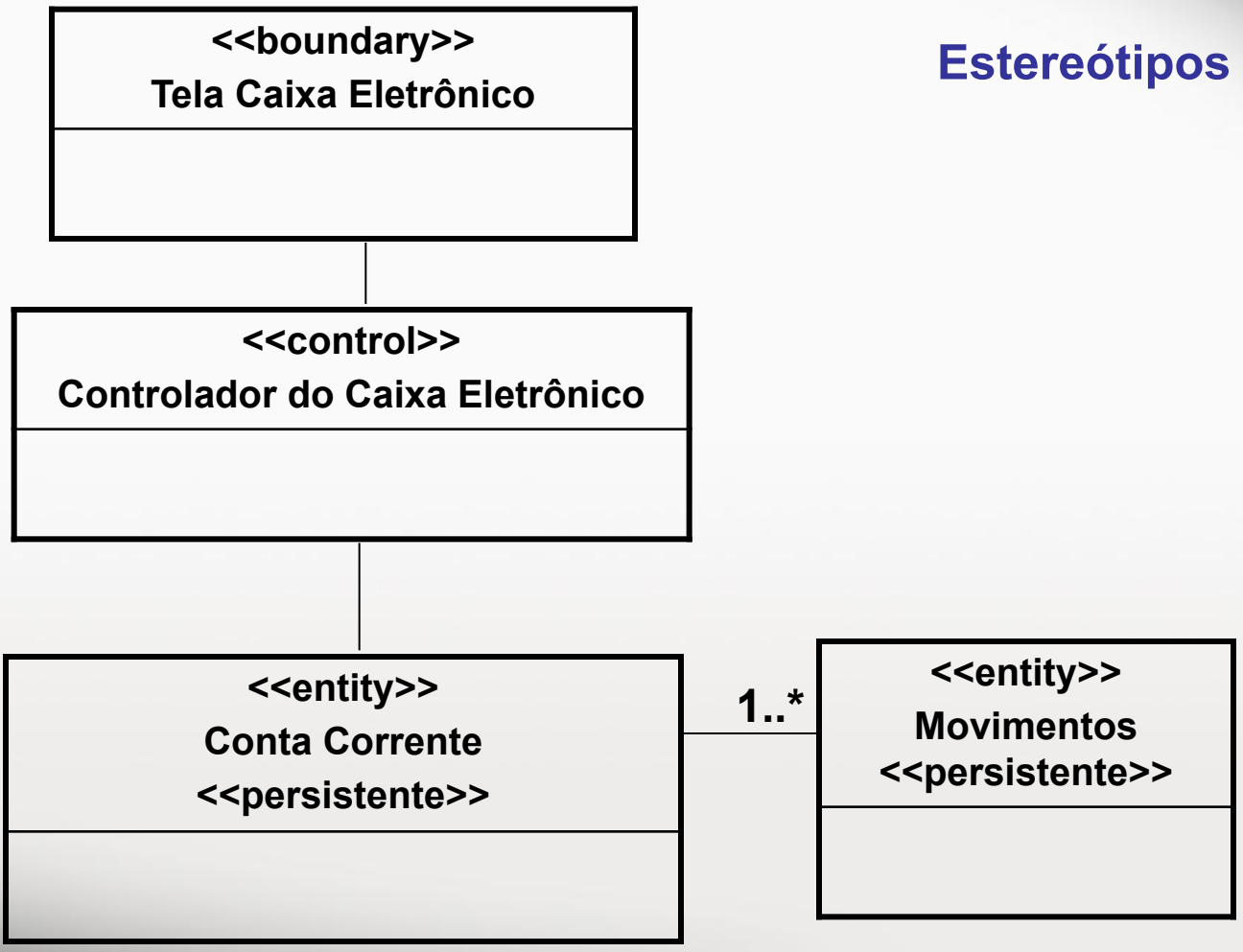
Estereótipos

Lugar-comum, clichê, rótulo.

Servem para destacar determinado componente do diagrama,indicando comportamentos ou funcionalidades específicas. Podem ser predefinidos em UML ou particulares:

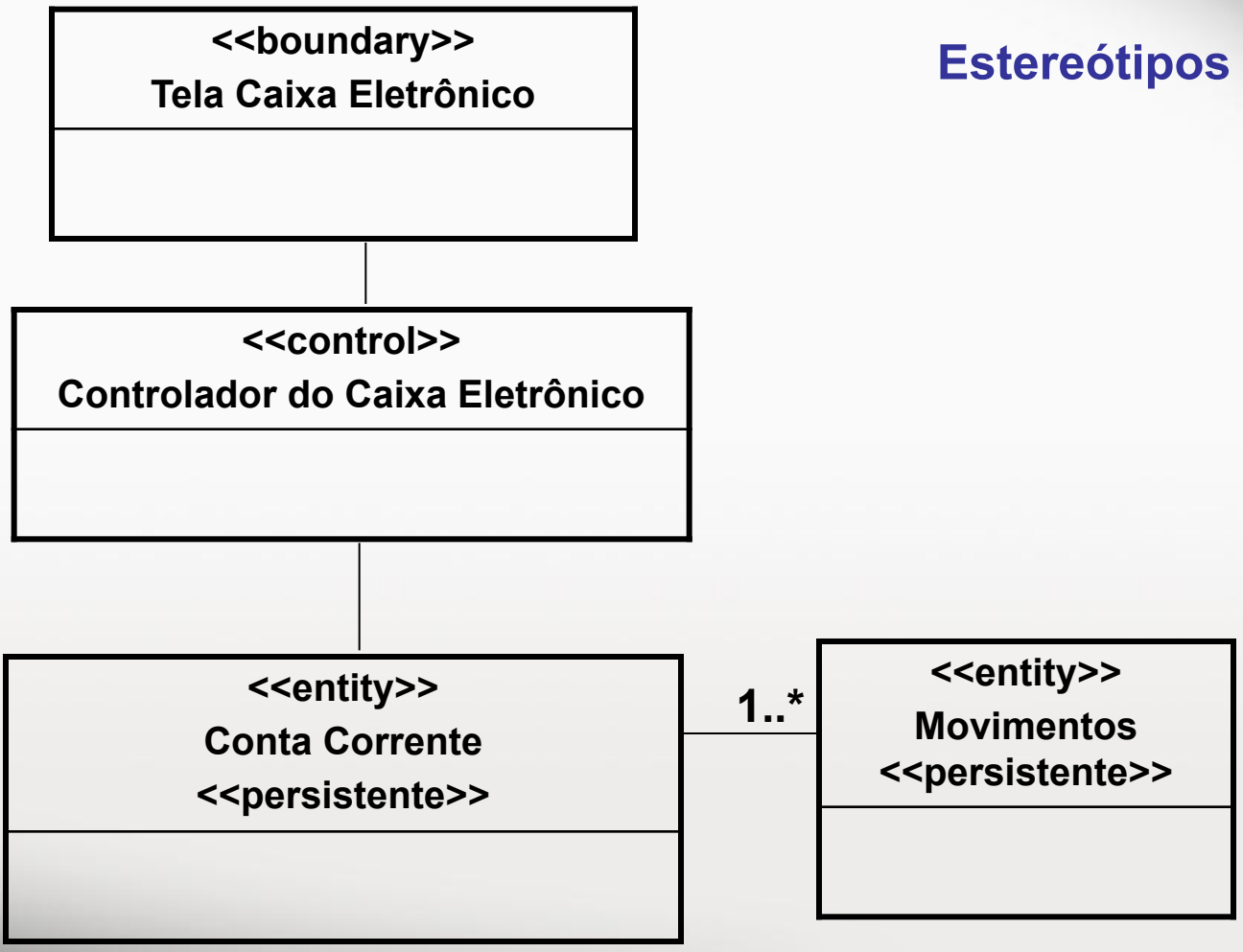
- **<<entity>> : a classe é uma entidade do sistema, recebidas ou geradas por este e serão usadas ao longo do sistena;**
- **<<persistente>> : particular, indica que tal instância deve ser preservada (gravada em disco, por exemplo);**
- **<<boundary>> : fronteira, identifica uma classe que faz a interface entre o sistema e um ator;**
- **<<control>> : servem de intermediárias entre as classes de interface e outras do sistema. Por exemplo, interpretar eventos ocorridos em uma interface e enviar os dados para outra classe**

Engenharia de Software II



Estereótipos

Engenharia de Software II



Estereótipos

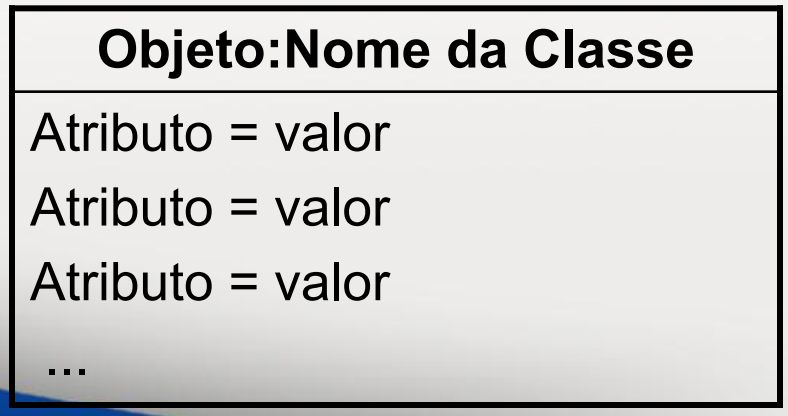
Engenharia de Software II

Diagrama de Objetos

Um complemento do Diagrama de Classses, fornece um exemplo da utilização das classes – os objetos (instâncias) - em um determinado momento da execução do sistema. São representados somente os atributos

Fornece:

- **Simulação dos objetos (dinâmico).**



Engenharia de Software II

Diagrama de Objetos