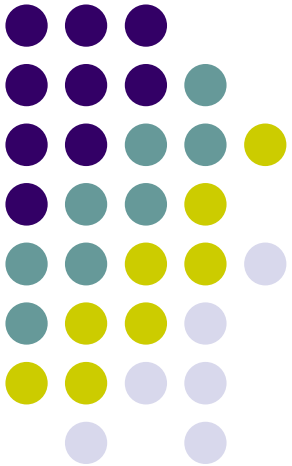
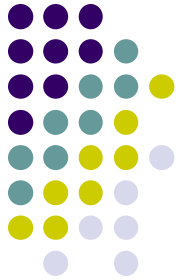
Diagrama de Classes



2019

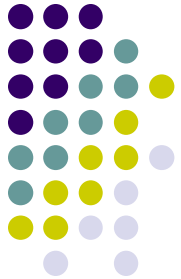
Diagrama de Classes



- *Visa permitir a visualização das classes que comporão o sistema junto com os respectivos atributos e métodos, bem como mostrar como as classes se relacionam, complementam e transmitem informações entre si.*

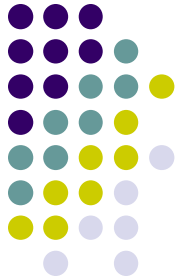
- *Diagrama mais IMPORTANTE*
- *Diagrama mais UTILIZADO*
- *Serve como BASE como para os demais diagramas*
- *Evolução do E-R*
- *Pode ser levado para um Banco de Dados*

Diagrama de Classes



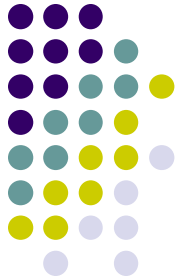
- Um diagrama de classes serve para modelar o vocabulário de um sistema, do ponto de vista do utilizador/problema ou do implementador/solução
  - Ponto de vista do utilizador/problema – na fase de captura e análise de requisitos, em paralelo com a identificação dos casos de utilização
  - Vocabulário do implementador/solução – na fase de projeto (*design*)
- Construído e refinado ao longo das várias fases do desenvolvimento do software, por analistas, projetistas (*designers*) e implementadores
- Também serve para:
  - Especificar colaborações (no âmbito de um caso de utilização ou mecanismo)
  - Especificar esquemas lógicos de bases de dados
  - Especificar vistas (estrutura de dados de formulários, relatórios, etc.)
- Modelos de objetos de domínio, negócio, análise e *design*

Diagrama de Classes



- No desenvolvimento de software orientado por objetos, procura-se imitar no computador o mundo real visto como um conjunto de objetos que interagem entre si
- Muitos objetos computacionais são imagens de objetos do mundo real
- Dependendo do contexto (análise ou projeto) podemos estar a falar em objetos do mundo real, em objetos computacionais ou nas duas coisas em simultâneo
- Exemplos de objetos do mundo real:
  - o Sr. João
  - a aula de ES no dia 11/10/2000 às 11 horas
- Exemplos de objetos computacionais:
  - o registro que descreve o Sr. João (imagem de objeto do mundo real)
  - uma árvore de pesquisa binária (objeto puramente computacional)

Diagrama de Classes



- ***PERSISTÊNCIA***

- *Visa preservar de maneira permanente os objetos de uma classe – "gravar em disco"*
- *Nem toda classe é/precisa ser persistente ...*
- *Necessário explicitamente definir através de um esteriótipo/restrição*

Diagrama de Classes

- ***CLASSES, ATRIBUTOS E MÉTODOS***

- *Atributos – Armazenam os dados dos objetos*
- *Métodos – Funções que uma instância da classe pode executar*

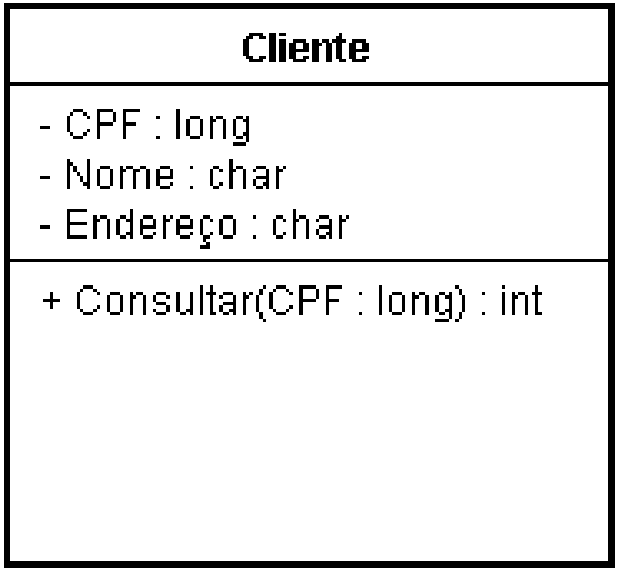


Diagrama de Classes

- ***VISIBILIDADE***

- *"+" = Visibilidade pública – pode ser utilizado por qq classe*
- *"#" = Visibilidade protegida – somente a própria classe ou suas subclasses podem ter acesso*
- *"-" = Visibilidade privada – Somente a classe possuidora do atributo poderá utilizá-lo*

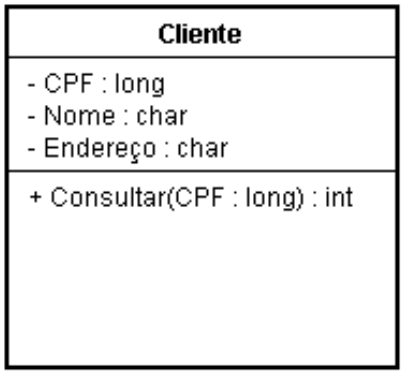
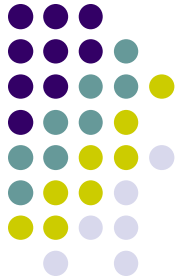


Diagrama de Classes

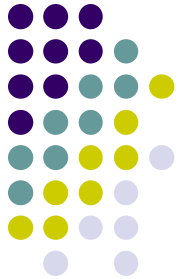


- ***RELACIONAMENTOS***

- *As classes costumam ter relacionamentos entre si com o intuito de compartilhar informações e colaborarem umas com as outras para permitir a execução dos processos*

- *Associações*
- *Especialização/Generalização*

Diagrama de Classes



ASSOCIAÇÕES

- *Descreve um vínculo que ocorre normalmente entre duas classes (binária),entre uma classe com ela mesma (unária) e entre várias classes (ternária/N-ária)*

- *Determinam-se que instâncias de uma classe estão de alguma forma ligadas às instâncias de outra classe – podendo haver troca de informações e compartilhamento de métodos*

Diagrama de Classes

- ***ASSOCIAÇÃO UNÁRIA***

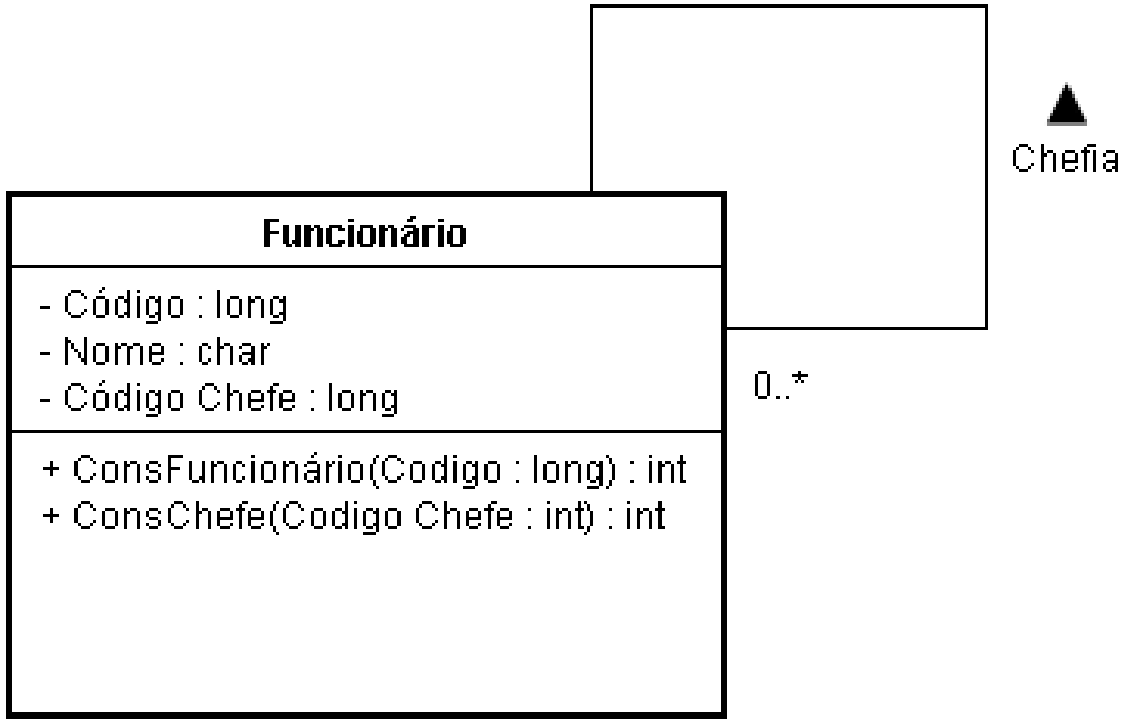


Diagrama de Classes

- ***ASSOCIAÇÃO UNÁRIA - exemplo***

*Associação "**Chefia**"*

Determina que um funcionário pode ou não chefiar outros funcionários

*Multiplicidade "**0..***" (semelhante a cardinalidade)*

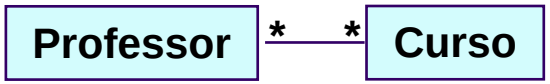
Indica que um determinado funcionário pode chefiar nenhum (0) ou muitos (*) funcionários

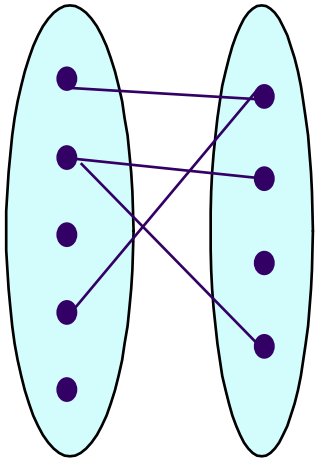
Diagrama de Classes

Multiplicidade	Significado
0..1	**No mínimo zero e no máximo um**. Indica que os objetos das classes não precisam obrigatoriamente estar relacionados mas, se houver, apenas uma instância da classe se relaciona
1..1	**Um e somente um**. Indica que apenas um objeto da classe se relaciona com a outra classe.
0..*	**No mínimo nenhum e no máximo muitos**. Indica que pode ou não haver instâncias da classe participante do relacionamento.
*	**Muitos**. Indica que muitos objetos da classe estão envolvidos
1..*	**No mínimo um e no máximo muitos**. Indica que há pelo menos um objeto envolvido no relacionamento, podendo haver muitos.
3..5	**No mínimo três e no máximo cinco**.

Multiplicidade de associações binárias

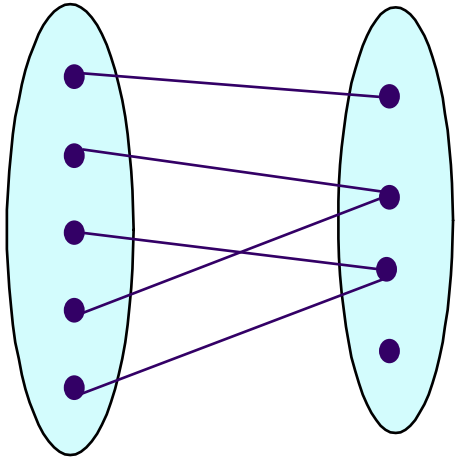
Muitos-para-Muitos



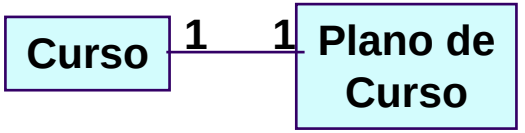


(sem restrições)

Muitos-para-1



1-para-1



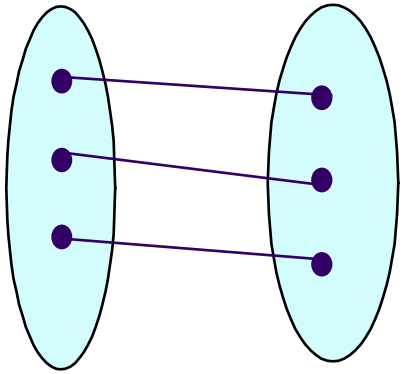
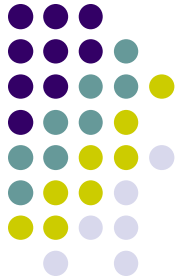


Diagrama de Classes



- ***ASSOCIAÇÃO BINÁRIA***

- *Associações entre duas classes*
- *Mais comum*

Diagrama de Classes

- ***ASSOCIAÇÃO BINÁRIA***

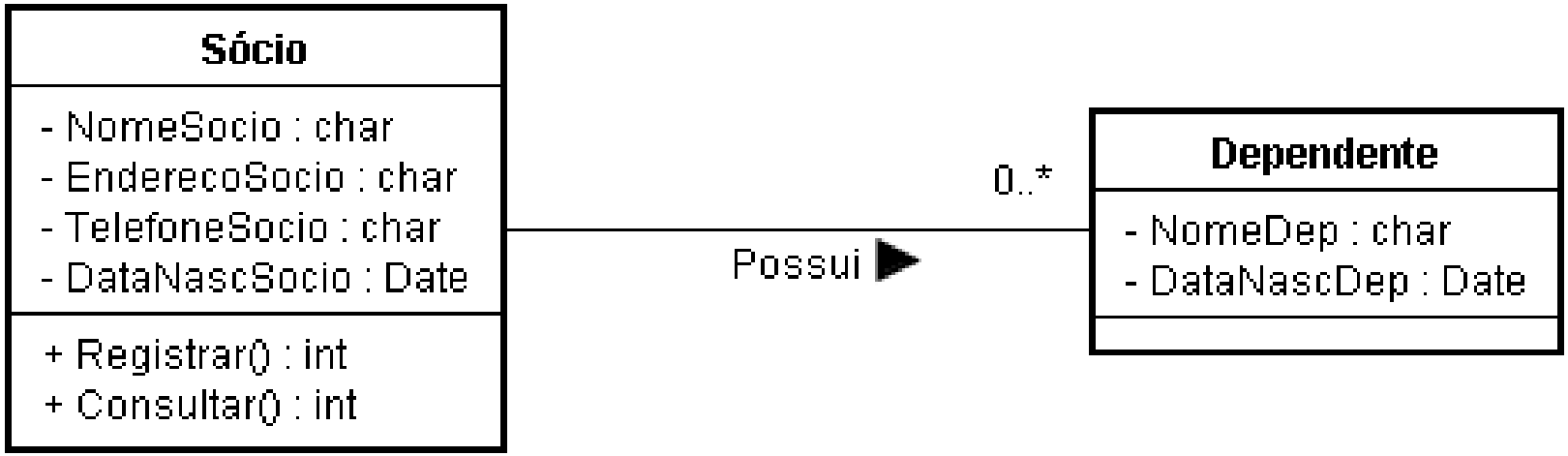
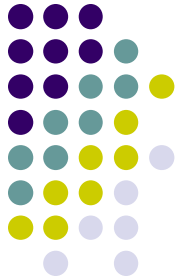


Diagrama de Classes



- ***ASSOCIAÇÃO TERNÁRIA ou N-ÁRIA***

- *Associações que conectam mais de duas classes*
- *São representadas por um losângulo para onde convergem todas as ligações de associação.*

Diagrama de Classes

- ***ASSOCIAÇÃO TERNÁRIA ou N-ÁRIA***

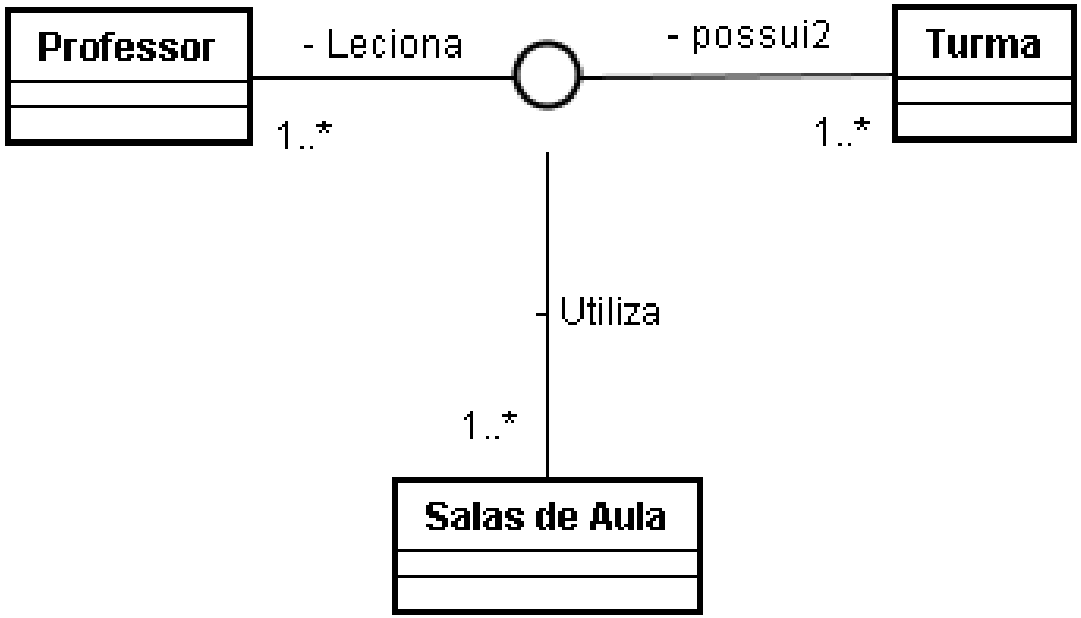
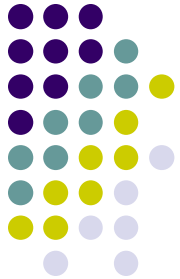


Diagrama de Classes



- ***AGREGAÇÃO***

Tipo especial de associação onde tenta-se demonstrar que as informações de um objeto precisam ser complementadas pelas informações contidas em um ou mais objetos de outra classe

Relação Todo-Parte

Diagrama de Classes

- ***AGREGAÇÃO***

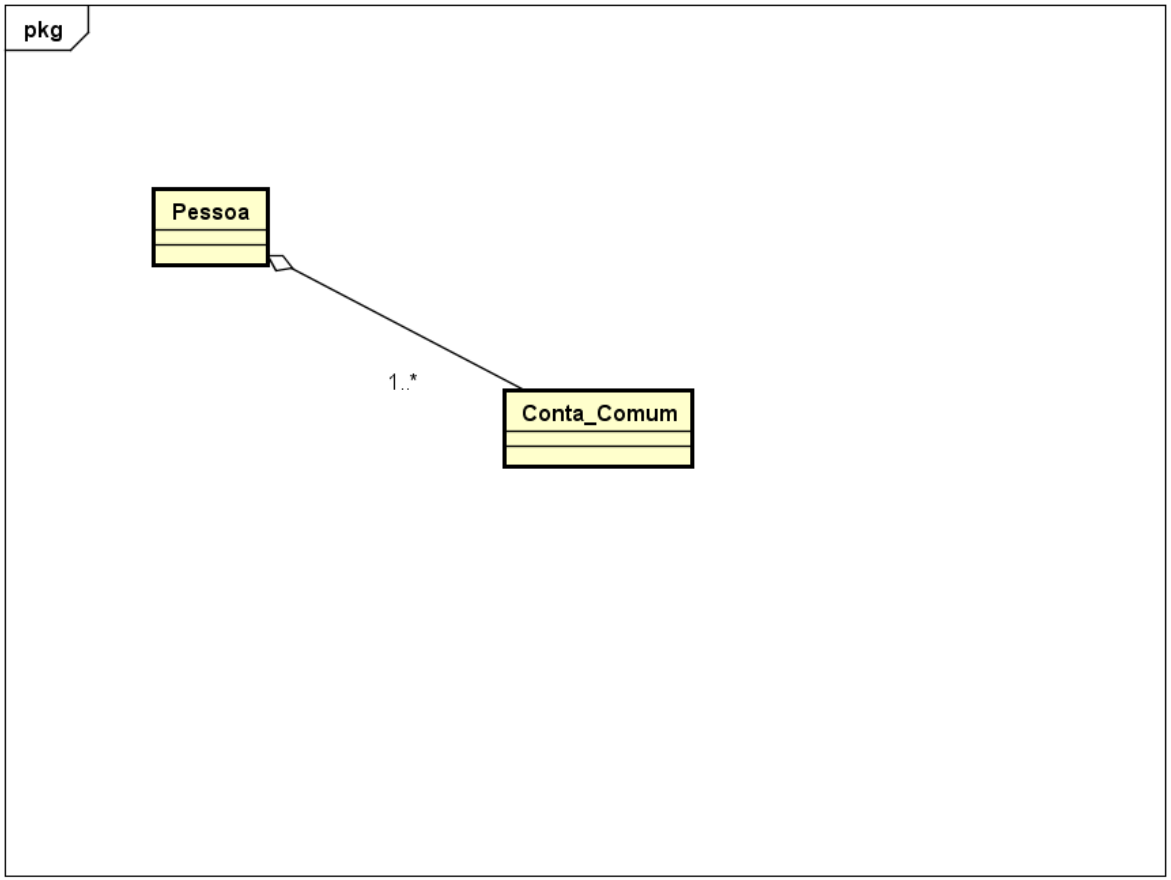
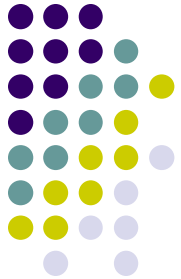


Diagrama de Classes



COMPOSIÇÃO

- *Variação da associação de agregação.*
- *Vínculo mais forte entre Objetos-Todo e Objetos-Parte*
- *Objetos-Parte têm de pertencer exclusivamente a um Objeto-Todo*

Diagrama de Classes

- ***COMPOSIÇÃO***

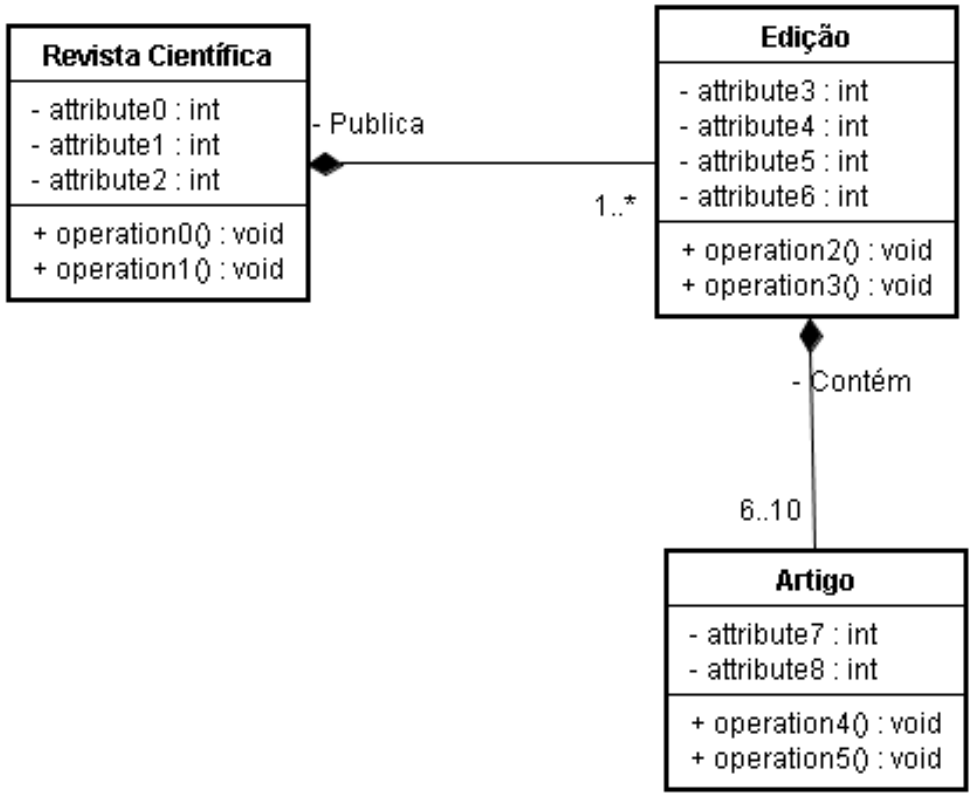
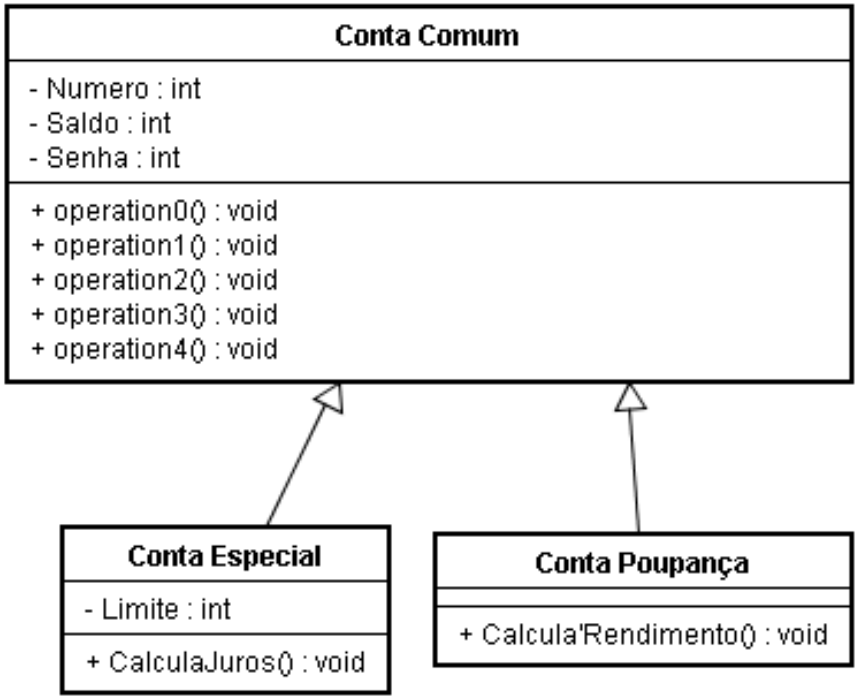


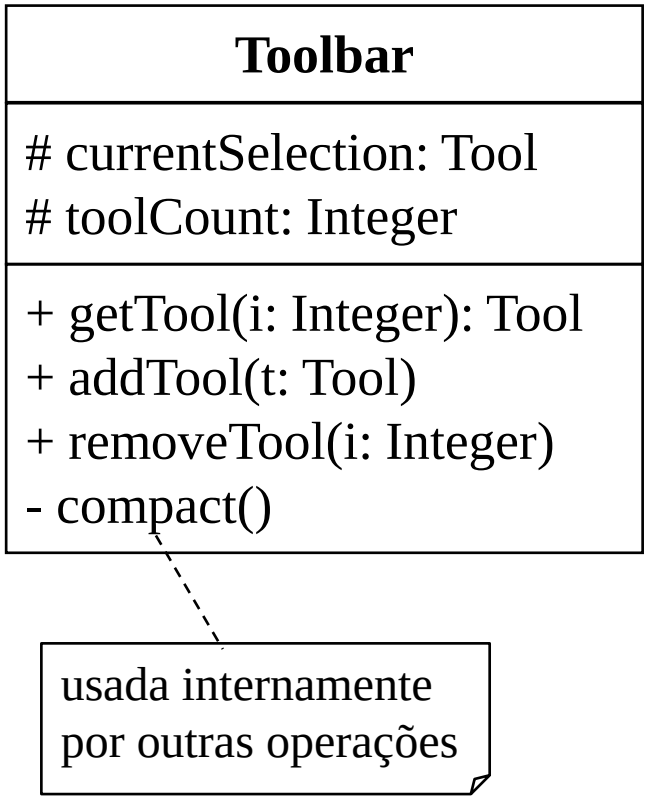
Diagrama de Classes

- ***ESPECIALIZAÇÃO/GENERALIZAÇÃO***



Visibilidade de atributos e operações

- Visibilidade:
  - \+ (public) : visível por todos
  - \- (private) : visível só por operações da própria classe
  - # (protected): visível por operações da própria classe e descendentes (subclasses)
- Princípio do **encapsulamento**: esconder todos os detalhes de implementação que não interessam aos clientes (utilizadores) da classe
  - permite alterar representação do estado sem afectar clientes
  - permite validar alterações de estado



Atributos de instância

- Atributos são listados num compartimento de atributos (opcional) a seguir ao compartimento com o nome da classe
- Uma classe não deve ter dois atributos com o mesmo nome
- Os nomes dos tipos não estão pré-definidos em UML, podendo-se usar os da linguagem de implementação alvo

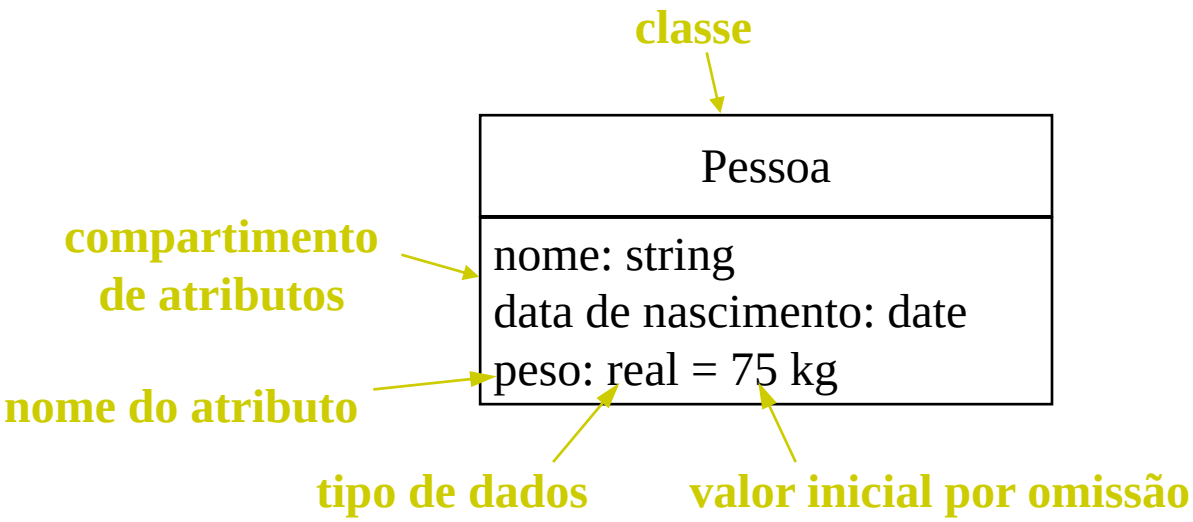
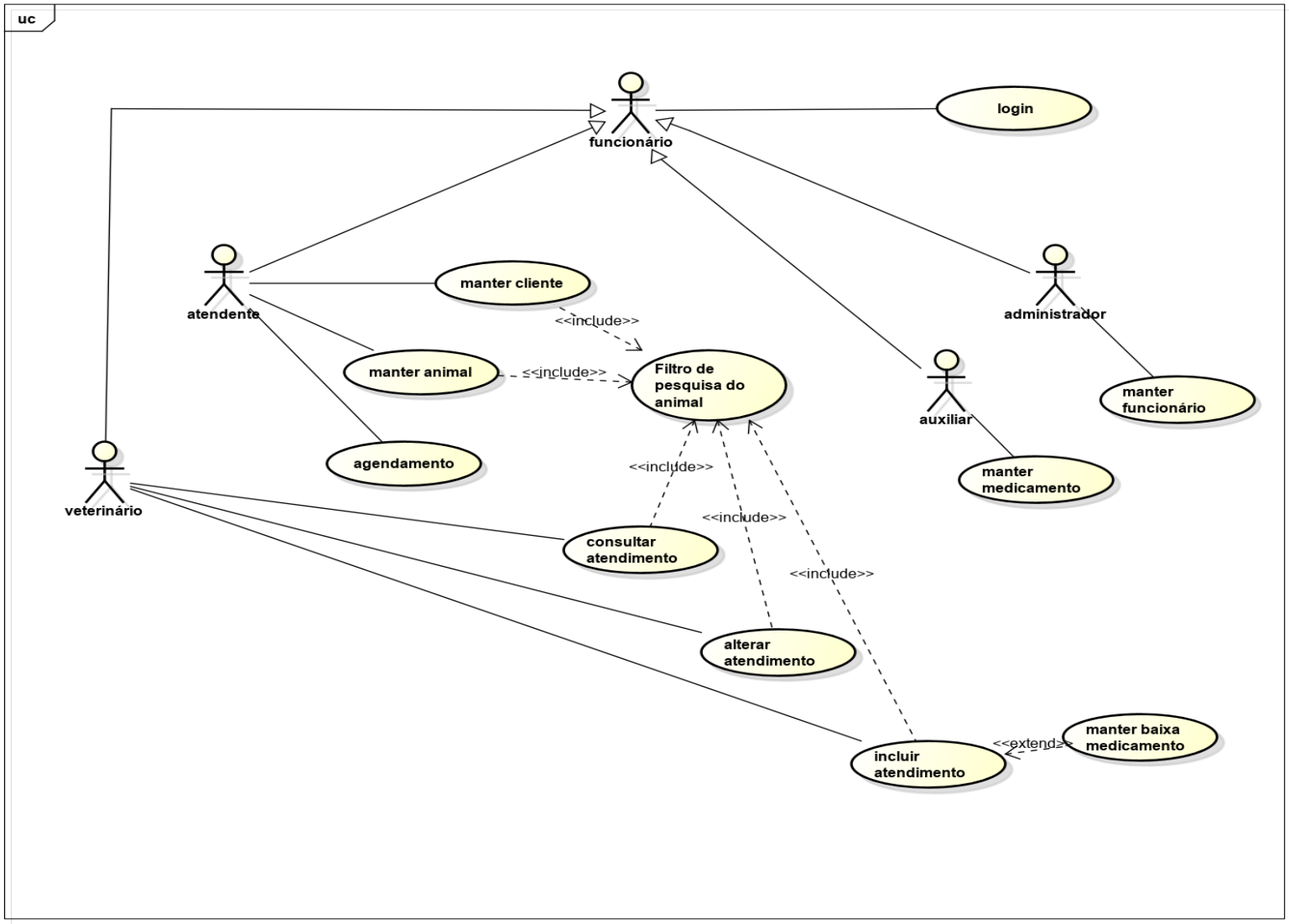
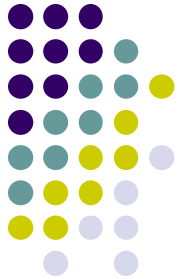


Diagrama de Classes





- Referência Bibliográfica:

- UML2. Guedes.Novatec