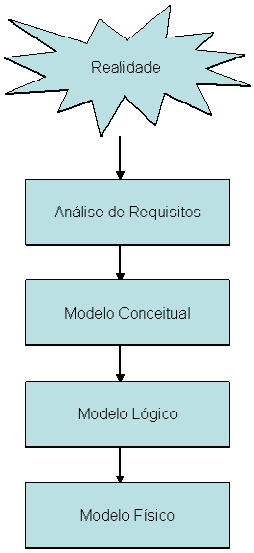
Projeto de Banco de Dados

Etapas:



Um projeto de banco de dados é desenvolvido em 3 etapas: modelagem de dados ou modelo conceitual, modelo lógico e modelo físico.

Modelo Conceitual => descreve a realidade da empresa ou de parte dela que se esta analisando, capturando as entidades e suas associações.

Modelo Lógico => descreve a implementação do modelo conceitual de acordo com o modelo de dados do SGBD (rede, hierárquico, relacional, orientado a objetos) do SGBD onde será feita a aplicação.

Modelo Físico => descreve as estruturas físicas de armazenamento tais como: tamanho de campos, índices, nomes etc.

MODELO CONCEITUAL:

Como é feito?

- Processo de análise;
- Através de entrevistas com os futuros usuários do sistema;
- Coletando-se documentos, relatórios existentes;
- Levantando-se as necessidades de informações dos usuários.
- Mapeado no Modelo de Classes do Domínio (UML)

MODELAGEM DE CLASSES DO DOMÍNIO

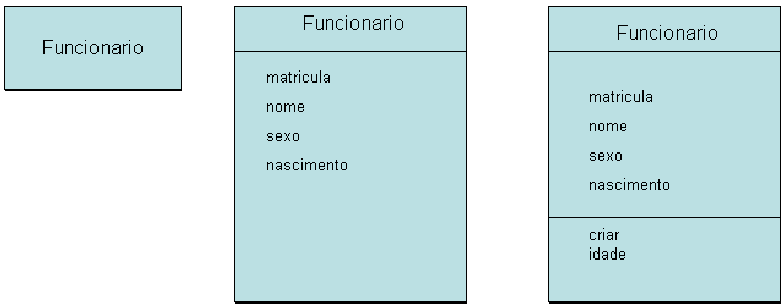
- Objetos do Sistema colaboram uns com os outros.
- Esta colaboração pode ser vista sob o aspecto estrutural estático e dinâmico.
- O aspecto dinâmico descreve a troca de mensagens entre os objetos.
- O aspecto estrutural estático permite compreender como o sistema esta estruturado internamente.
    - E estático porque não apresenta informações sobre como os objetos interagem no decorrer do tempo.

- E estrutural porque a estrutura das classes de objetos e as relações entre elas são representadas.
- Diagrama da UML usado para representar este aspecto e o diagrama de classes.
- O modelo de classes e composto por este diagrama e da descrição textual associada.

- Nomenclatura
  - Para identificadores são removidos do nome espaços em branco e preposições.
  - Para nomes de classes e nomes de relacionamentos as palavras componentes do nome são escritas começando por letra maiúscula. Ex: **Cliente, ItemPedido**
  - Para nomes de atributos e nomes de operações escrever a primeira palavra do nome em minúsculo e as palavras subseqüentes iniciando com letra maiúscula. Ex: **quantidade, precoUnitario**.

DIAGRAMA DE CLASSES

- Classes representam objetos que existem no mundo real com as quais é necessário guardar a informação. Exemplo: Pessoa, Livro, etc.
- Podem ser concretas (Pessoa, Livro) ou abstratas(Feriado, Conta)
- São descritas por atributos e seus valores (conjunto de domínio).
- Representação:

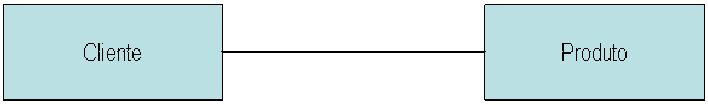


Objetos de uma mesma classe compartilham as mesmas operações.

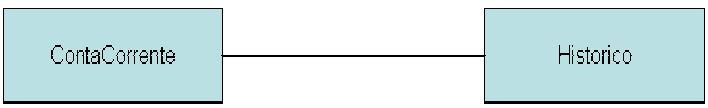
RELACIONAMENTOS (ASSOCIAÇÃO)

- É uma associação entre objetos das classes envolvidas na associação.
- Representação

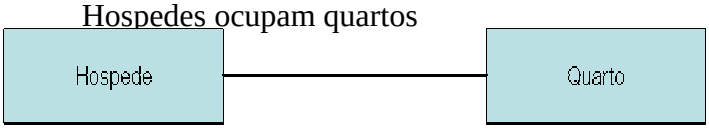
Um cliente compra produtos



Uma conta corrente possui um histórico de transações



Hospedes ocupam quartos



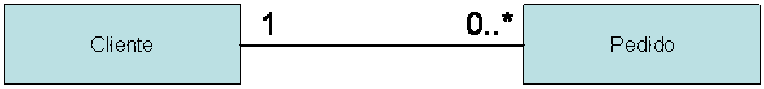
CARACTERÍSTICAS DOS RELACIONAMENTOS

1. **Multiplicidade**

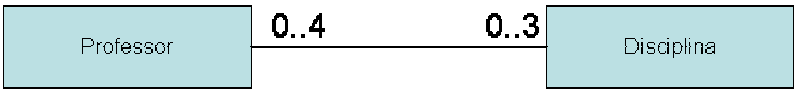
   As associações permitem representar a informação dos limites inferior e superior das quantidades de objetos aos quais um outro objeto pode estar associado. Estes limites são chamados de multiplicidade na terminologia UML. Cada associação possui duas multiplicidades, uma em cada extremo da linha de associação. A simbologia para representação é:

Nome	Simbologia
Apenas Um	1
Zero ou Muitos	0..*
Um ou Muitos	1..*
Zero ou Um	0..1
Intervalo Específico	$l_i..l_s$

   Exemplos:
   Um cliente pode fazer nenhum ou muitos pedidos. Um pedido está associado a um e somente um cliente.



   Um professor pode ensinar nenhuma ou no máximo 3 disciplinas e uma disciplina pode ser ensinada por nenhum ou no máximo 4 professores.



2. **Conectividade**

   É o tipo de associação entre duas classes. As associações podem ser agrupadas em três grandes tipos:

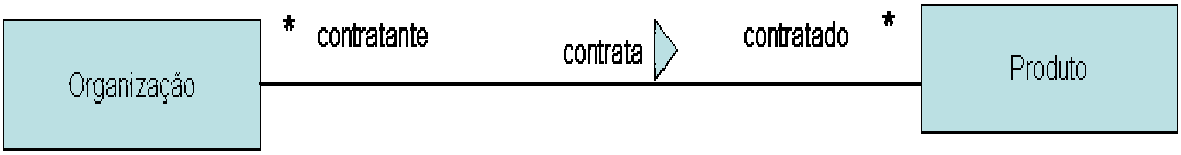
   - Um para Um: 0..1 ou 1 — 0..1, ou 1
   - Um para Muitos: 0..1 ou 1 — * , ou 0..*, ou 1..*
   - Muitos para Muitos: *, ou 0..*, ou 1..* — * , ou 0..*, ou 1..*

3. **Participação**

Indica a necessidade ou não da existência desta associação entre objetos. Pode ser obrigatória ou opcional. A multiplicidade (mínima) de valor 1 (obrigatória) indica que um objeto da classe **Pedido** só pode existir se houver um **Cliente** associado. A multiplicidade mínima de valor 0 (opcional) indica que um objeto **Cliente** pode existir sem ter um objeto **Pedido** associado.

4. **Nome da associação, direção de leitura e papeis**

O nome é posicionado no meio da linha de associação. A direção indica como o relacionamento deve ser lido. Uma característica complementar é o papel que um objeto representa na associação.



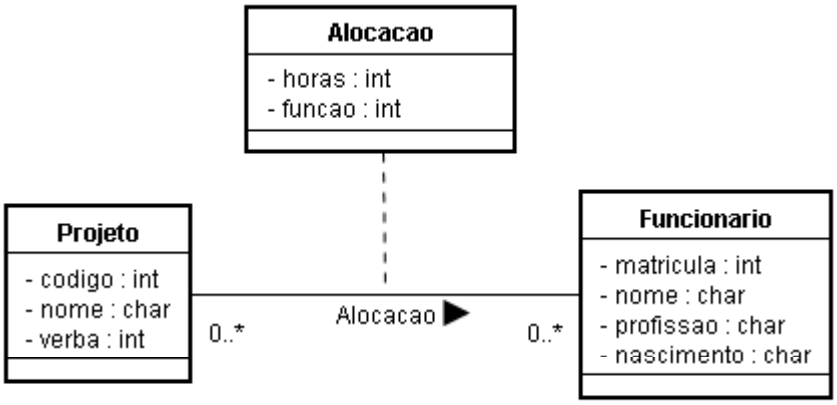
OBSERVAÇÕES:

- O nome da associação deve ser simples. É preferível não nomear do que usar nomes vagos ou óbvios.
- O mesmo vale para os papeis (O diagrama pode ficar poluído)

CLASSES ASSOCIATIVAS

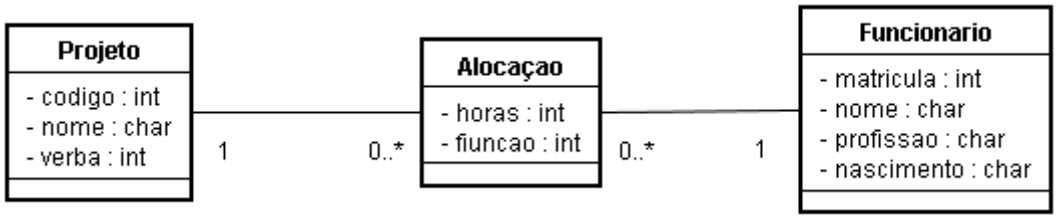
São classes que estão ligadas a associações. Ocorre quando duas ou mais classes estão associadas e é necessário manter informação sobre a associação. Em geral são encontradas classes associativas ligadas a associações de conectividade muitos para muitos.

Exemplo:

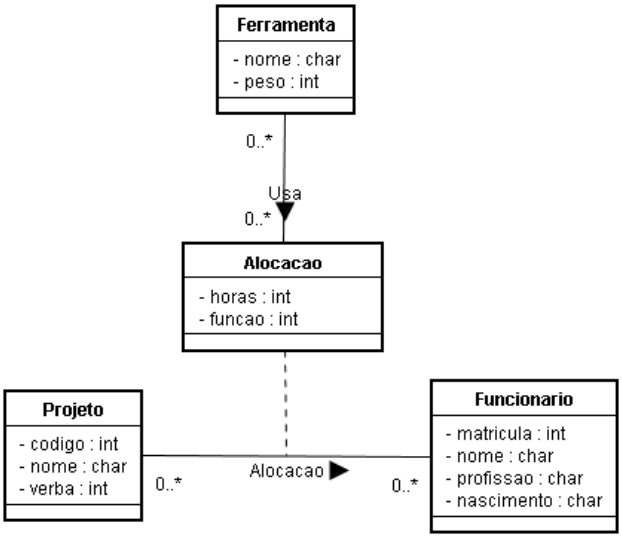


Um **Funcionário** trabalha em vários **Projetos** e um **Projeto** possui diversos **Funcionários** trabalhando. A classe associativa **Alocação** permite que se saiba para cada par [Projeto, Funcionário] quantas horas e em que função um funcionário ficou alocado naquele projeto.
Obs:

1) De uma forma geral, uma classe associativa pode ser substituída por uma classe ordinária, que deve estar associada no formato um para muitos com as classes originais.



2) Uma classe associativa pode participar de outros relacionamentos. Por exemplo, um funcionário alocado a um projeto pode usar nenhuma ferramenta ou muitas ferramentas.



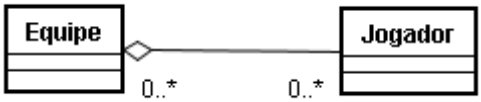
RELACIONAMENTOS ESPECIAIS

AGREGAÇÃO:

Uma agregação é utilizada para representar conexões entre objetos que guardam uma relação todo-parte entre si. A diferença entre agregação e associação é puramente semântica: em uma agregação um objeto está contido dentro de um outro ao contrário de uma associação. Entretanto, onde puder ser usada uma agregação, uma associação também poderá ser usada. Características:

- Agregações são assimétricas. Se o objeto A é parte de um objeto B então o objeto B não pode ser parte do objeto A.
- Objetos que fazem parte do todo são criados e destruídos independentemente do objeto todo.

Representação: Um diamante branco perto da classe que representa o todo.

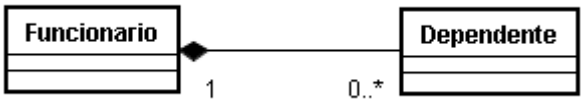


COMPOSIÇÃO:

É uma forma mais forte de agregação. Características:

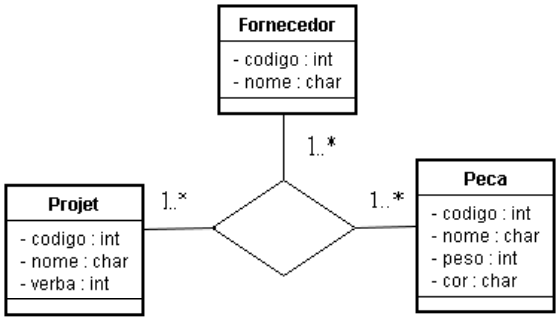
- Objetos parte pertencem a um único objeto todo.
- Objetos parte são sempre criados e destruídos pelo objeto todo.
- Se o objeto todo deixa de existir o mesmo acontece com suas partes.

Representação: Um diamante escuro do lado da classe que representa o todo.

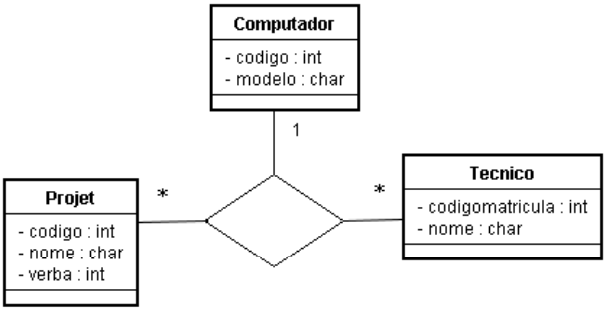


ASSOCIAÇÃO TERNÁRIA:
Ocorre quando é necessário associar objetos de três classes distintas.

Exemplos:
Fornecedores fornecem peças para projetos.



Técnicos utilizam computadores no projeto



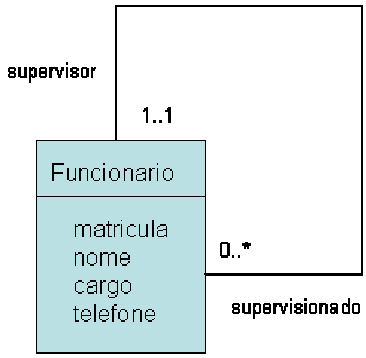
OBS: Assim como ocorre nos relacionamentos binários, os relacionamentos ternários também podem estar associados a uma classe associativa para guardar as informações de cada instância da associação.

ASSOCIAÇÃO REFLEXIVA (AUTO-ASSOCIAÇÃO)

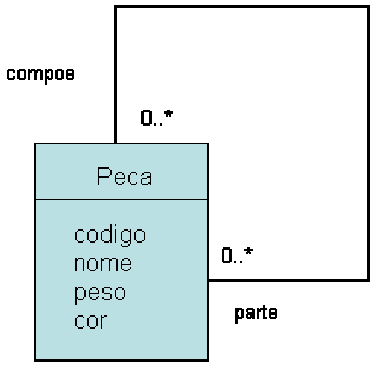
É uma associação entre objetos de uma mesma classe. Cada objeto tem um papel distinto nesta associação. Neste tipo de associação a utilização de papeis é bastante importante para evitar ambigüidades.

Exemplos:

Um funcionário supervisiona zero ou mais funcionários, e um funcionário é supervisionado por um único supervisor.

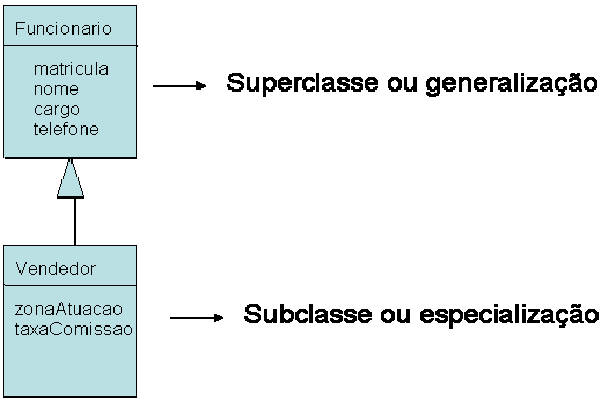


Uma peça é composta por nenhuma peça ou por muitas peças; e uma peça pode fazer parte da composição de nenhuma ou de muitas peças.



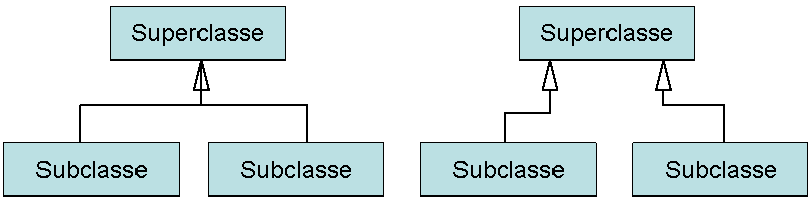
ASSOCIAÇÃO – GENERALIZAÇÃO / ESPECIALIZAÇÃO:
Uma associação entre classes, onde uma classe herda as propriedades e comportamentos de uma ou mais classes.

Exemplo: Todo funcionário posui matricula, nome, cargo e telefone. Aqueles que são vendedores possuem também a zona de atuação e a taxa de comissão.

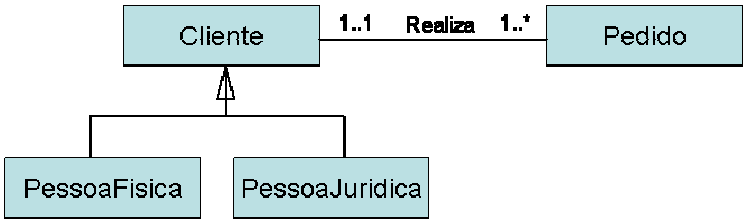


Obs1: Vendedor possui todas as características (atributos e operações) de funcionário

Representações alternativas:

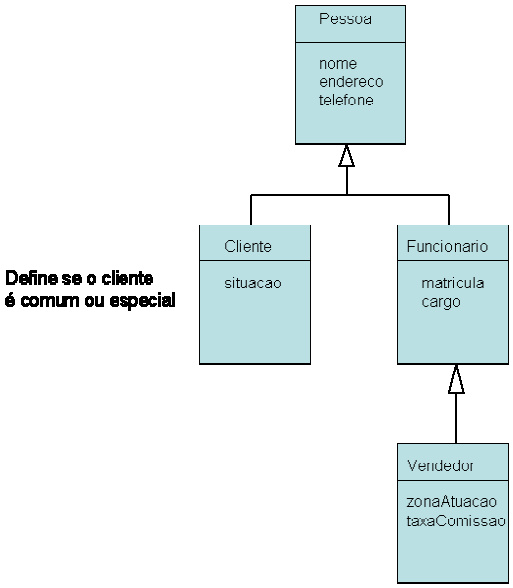


Obs2: Associações que estão definidas na superclasse também são herdadas pelas subclasses.



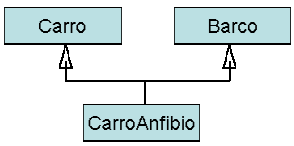
Não há necessidade de se criar associações entre Pedido e as subclasses.

Obs3: Uma classe em uma hierarquia herda tanto de uma superclasse imediata como de superclasses não imediatas.



Transitividade (Vendedor possui 7 atributos)

Obs4: Herança múltipla. Uma classe pode ter mais de uma superclasse.



Obs 5: Classes Abstratas: não geram instâncias diretas. Usadas para organizar uma hierarquia de generalização.