nemo

ontology & conceptual modeling research group

Banco de Dados – Modelagem Conceitual

Vítor E. Silva Souza

(vitorsouza@inf.ufes.br)

http://www.inf.ufes.br/~vitorsouza

Departamento de Informática

Centro Tecnológico

Universidade Federal do Espírito Santo

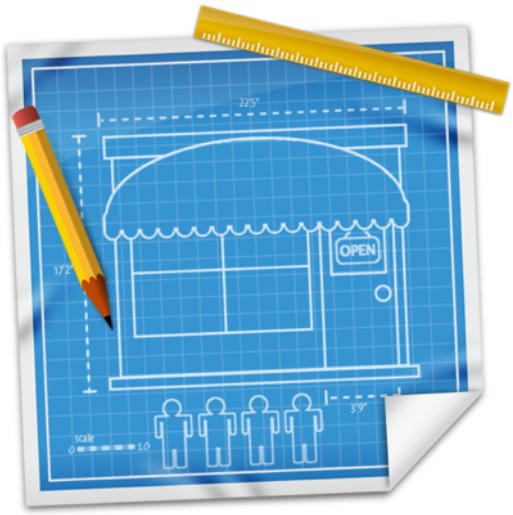
Licença para uso e distribuição

Créditos

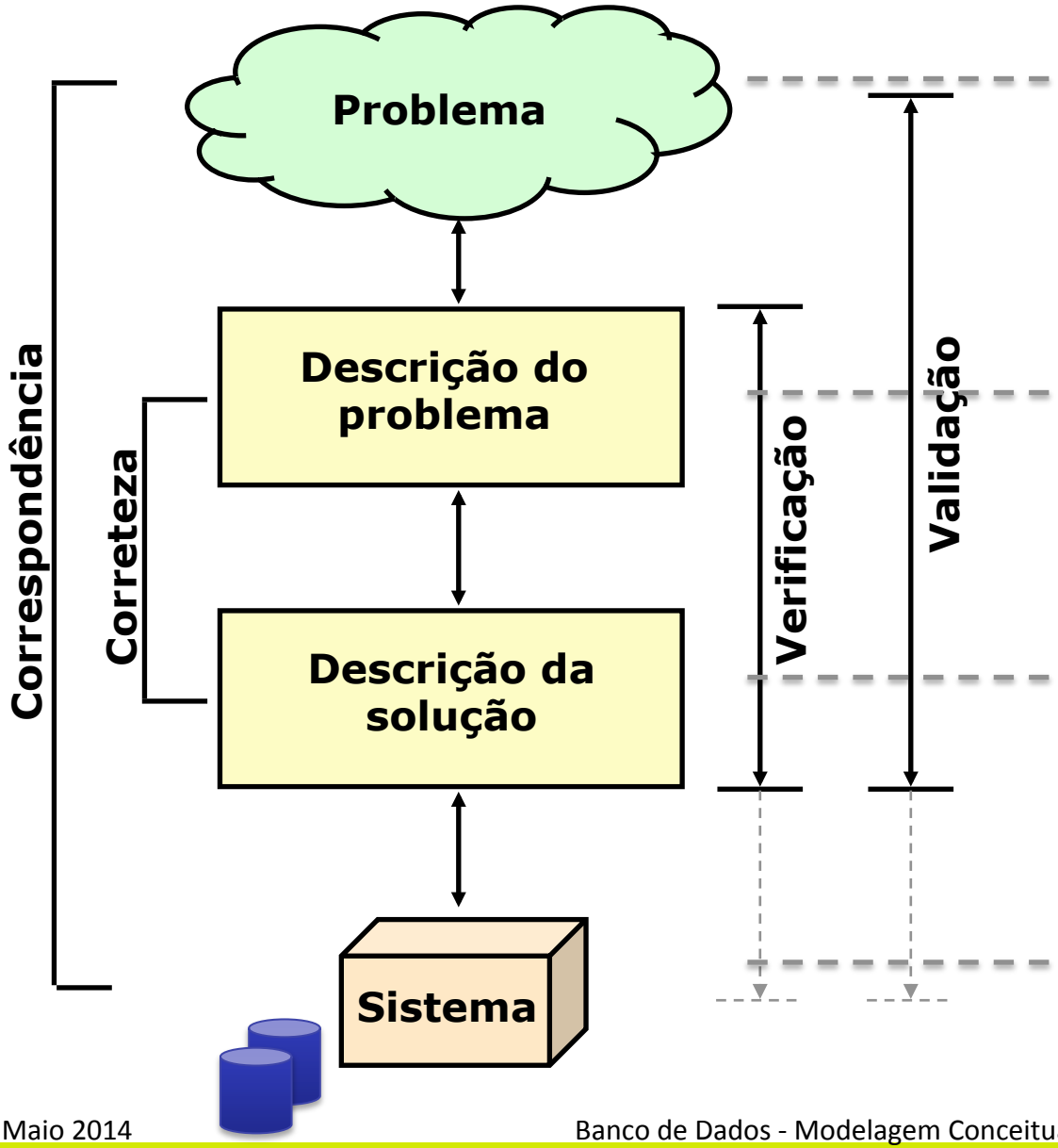
- Algumas informações foram retiradas e adaptadas dos slides do livro Projeto de Banco de Dados de Carlos A Heuser;

Modelos

- Maneira de projetar, comunicar, documentar, etc. soluções computacionais;
- Diversos níveis, por exemplo:
  - Ontologias (modelos genéricos, de domínio);
  - Requisitos (foco em um problema);
  - Projeto / arquitetura (foco em uma solução).
- Essenciais para o desenvolvimento de software;
- Assim como o desenvolvimento, também seguem os paradigmas (estruturado, OO, etc.).



Desenvolvimento de sistemas



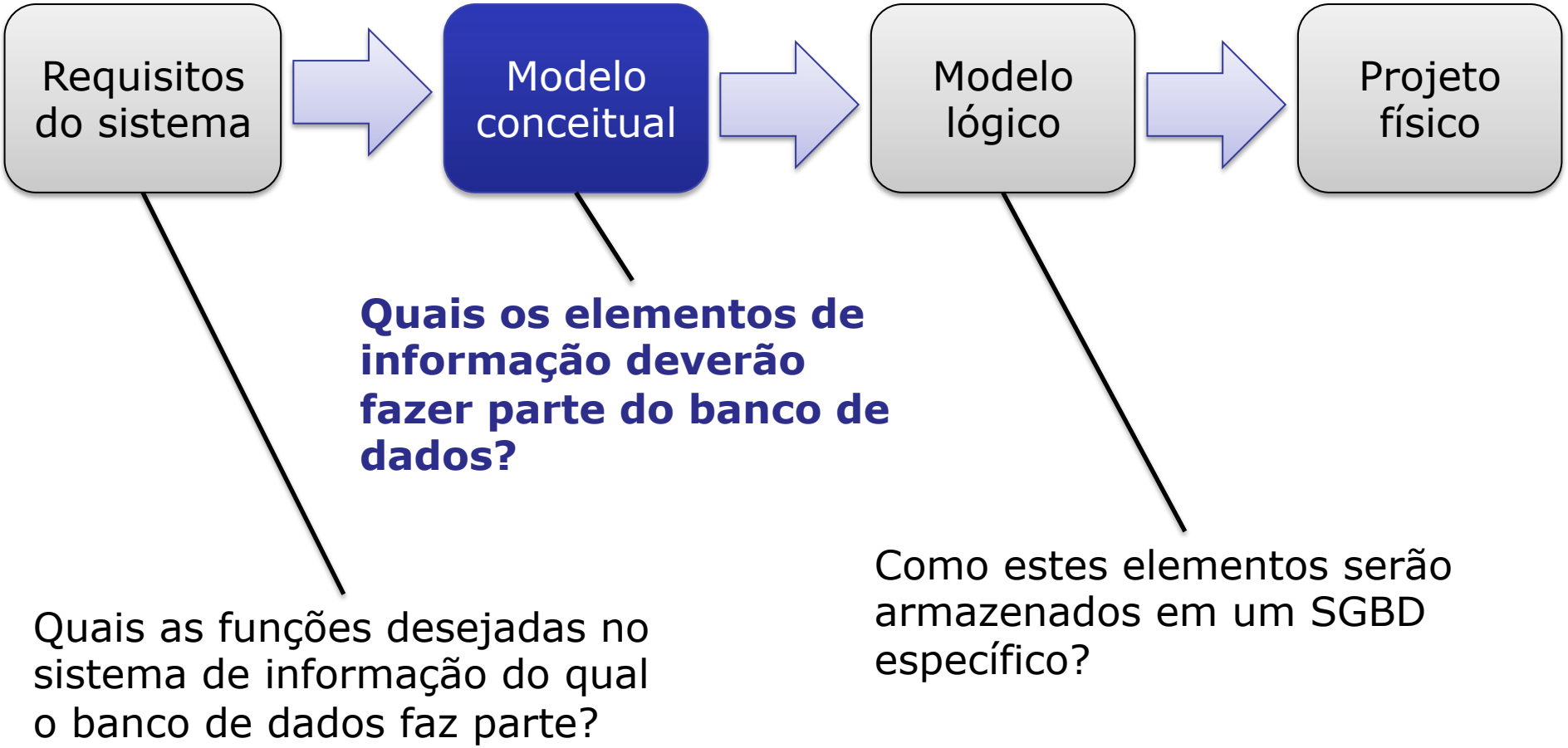
Uma biblioteca: livros, autores, usuários, funcionários, ...

Livros possuem título, autor, número de páginas, ...

Título é CHAR, número de páginas é INT, ...

Estruturas da ling. program. Tabelas do banco de dados

Projeto do banco de dados



Começar de um nível de abstração mais alto ajuda a comunicação com especialistas de domínio, a encontrar problemas mais cedo, etc.

Técnicas para modelagem conceitual

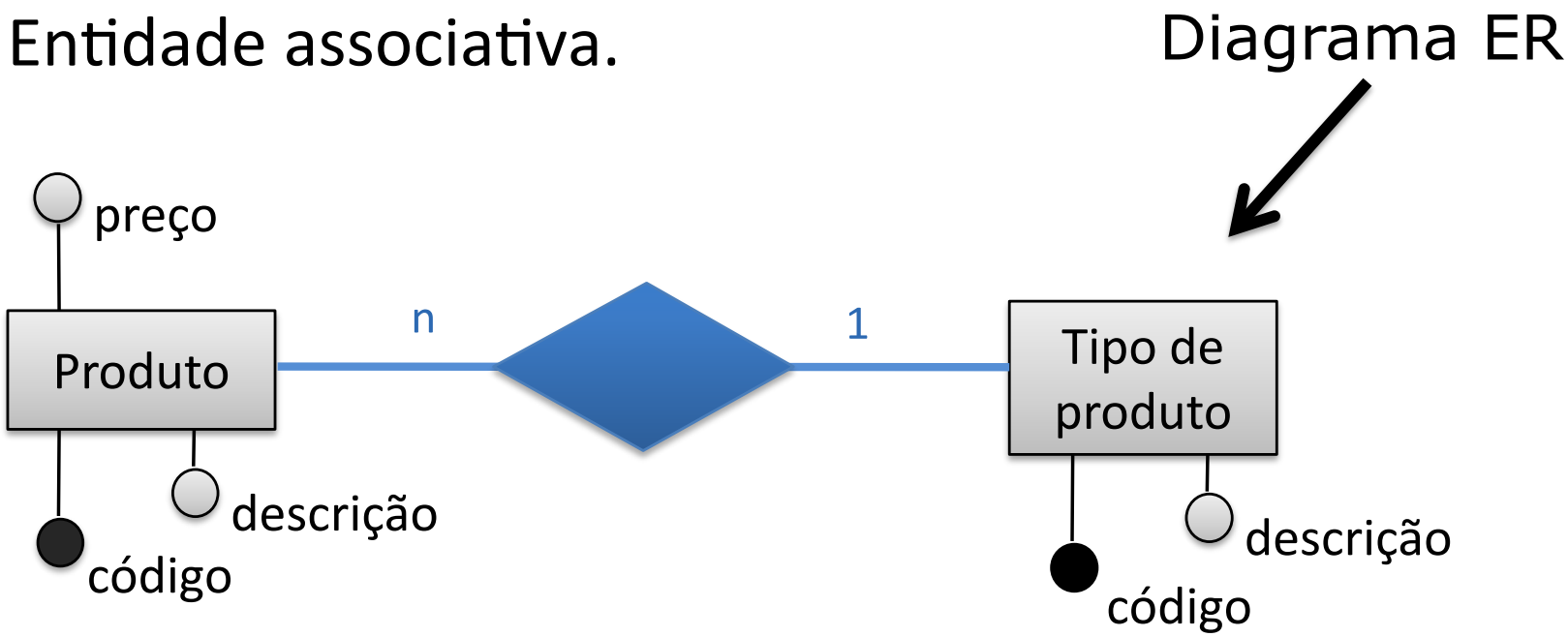
- Abordagem Entidade-Relacionamento (ER):
  - Criada em 1976, por Peter Chen;
  - Mais difundida e utilizada no paradigma estruturado;
  - Serviu de base para proposta subsequentes;
- A Linguagem de Modelagem Unificada (UML):
  - Junção de OMT (Rumbaugh), Booch e OOSE (Jacobson), criada na Rational Software em 1997;
  - Padrão ISO (2000) mantido pela OMG;
  - Não é exclusiva para modelagem conceitual;
  - Mais difundida e utilizada no paradigma orientado a objetos.

Diagramas da UML

- de Casos de Uso;
- de Classes;
- de Objetos;
- de Estrutura Composta;
- de Sequência;
- de Comunicação;
- de Estados;
- de Atividades;
- de Componentes;
- de Implantação;
- de Pacotes;
- de Interface Geral;
- de Tempo.

A abordagem Entidade-Relacionamento

- Conceitos centrais:
  - Entidade;
  - Relacionamento;
  - Atributo;
  - Generalização / especialização;
  - Entidade associativa.



Entidade: conceito

- Conjunto de objetos da realidade modelada sobre os quais deseja-se manter informações no banco de dados;
- Exemplos:
  - Em um sistema de informações industriais: produtos, tipos de produtos, vendas, compras, etc.;
  - Em um sistema de contas correntes: clientes, contas correntes, cheques, agências, etc.;
  - Em um sistema de marcação de reuniões: funcionários, salas, reuniões, agendamentos, etc.

Entidade: representação

- Podem representar objetos da realidade, sejam eles:
  - Concretos (ex.: pessoa, automóvel); ou
  - Abstratos (ex.: departamento, endereço);
- Representada no diagrama ER com um retângulo com o nome da entidade:

- A entidade (conceito) estabelece um conjunto de entidades ou classe de objetos;
- Os elementos desse conjunto são chamados de instâncias, ocorrências ou objetos.

Entidade: propriedades

- A entidade sozinha pouco informa;
- Precisamos saber suas **propriedades**;
- Em um modelo ER, são representadas por:
  - Relacionamentos;
  - Atributos;
  - Generalizações / especializações.

Relacionamento: conceito e representação

- Conjunto de associações entre entidades sobre as quais deseja-se manter informações na base de dados;
- Cada instância é uma ligação específica entre determinadas instâncias de entidade;

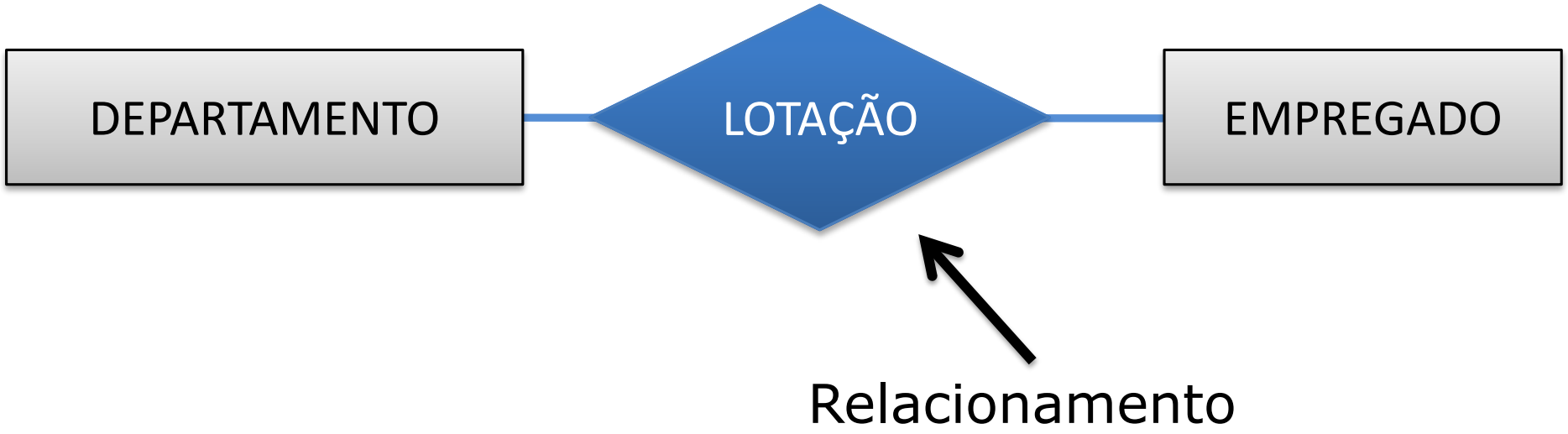
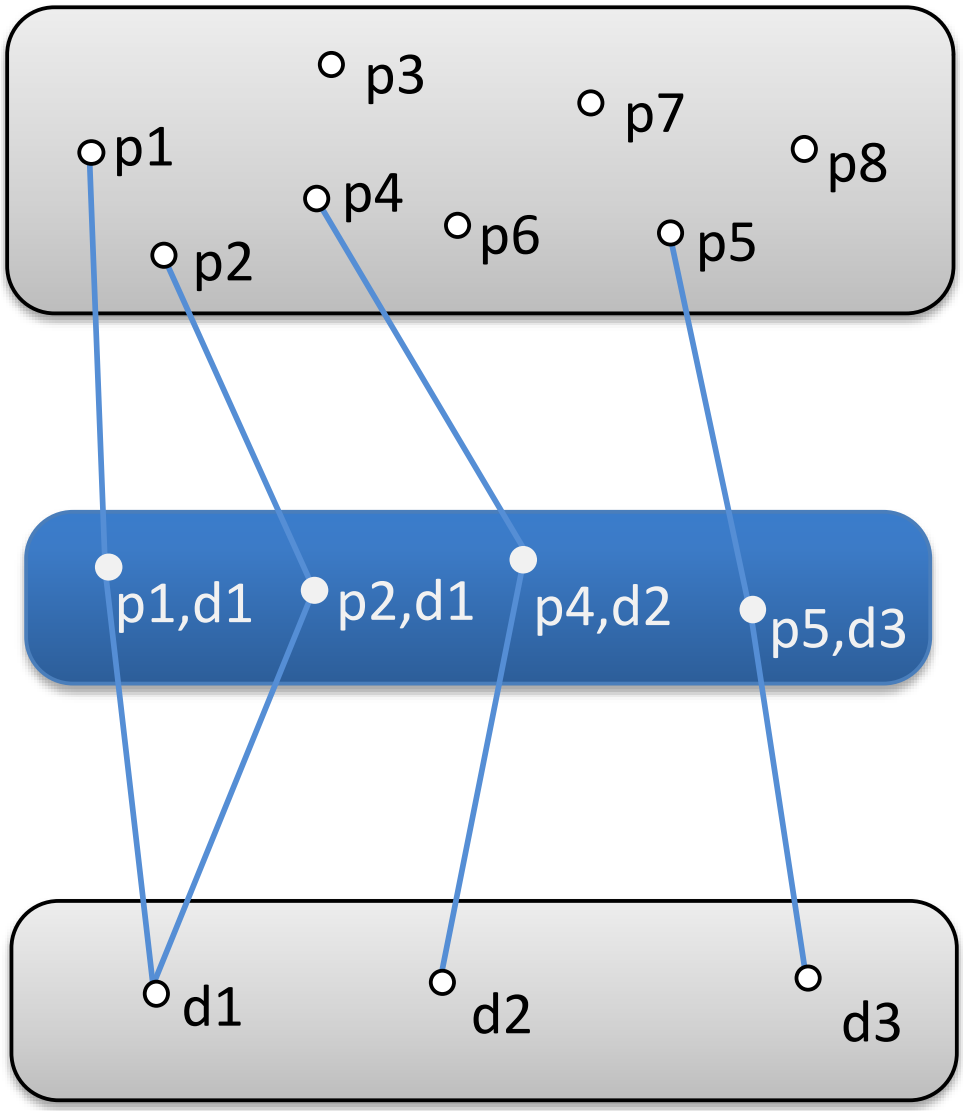


Diagrama de ocorrências

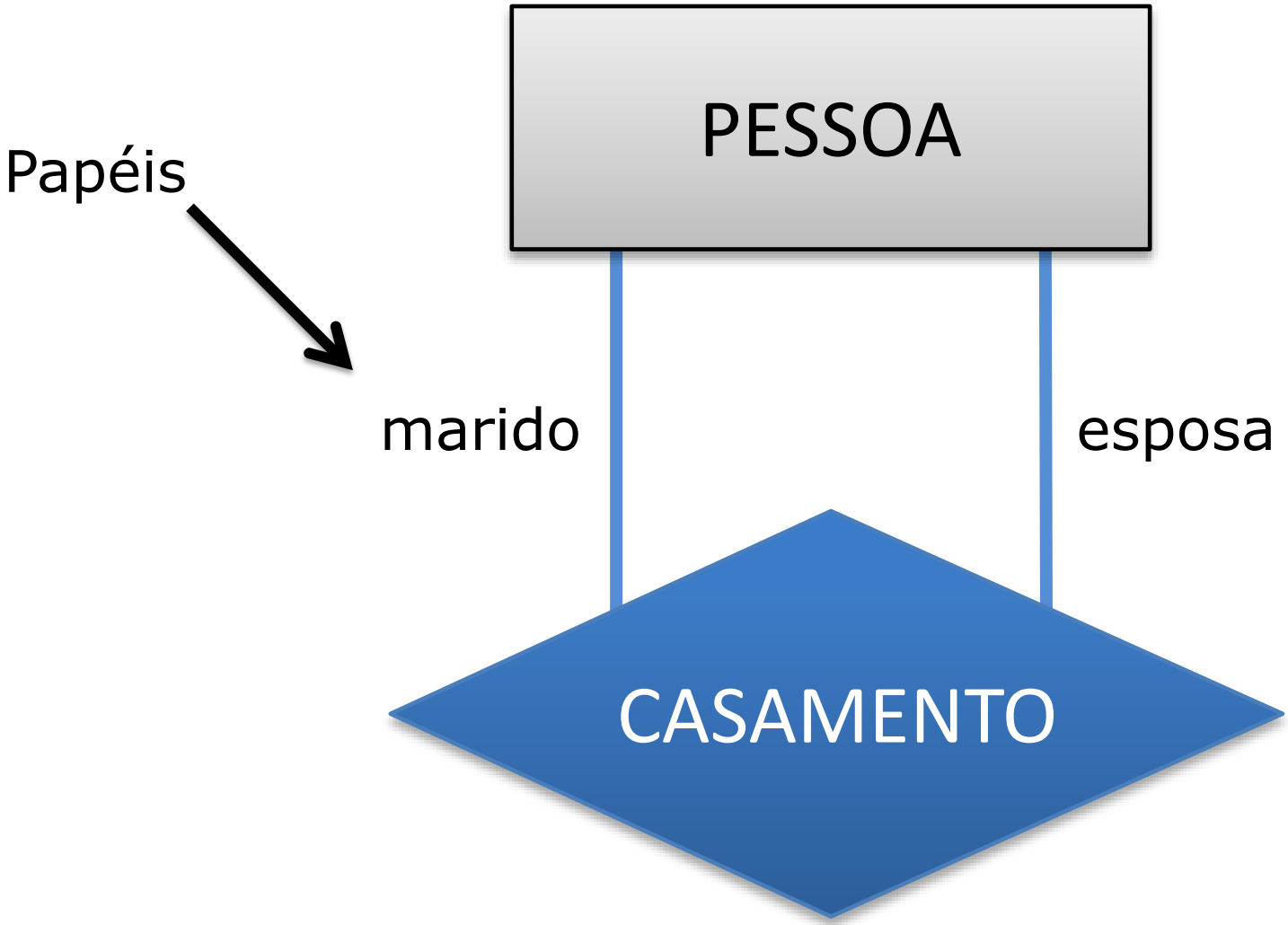


entidade
EMPREGADO

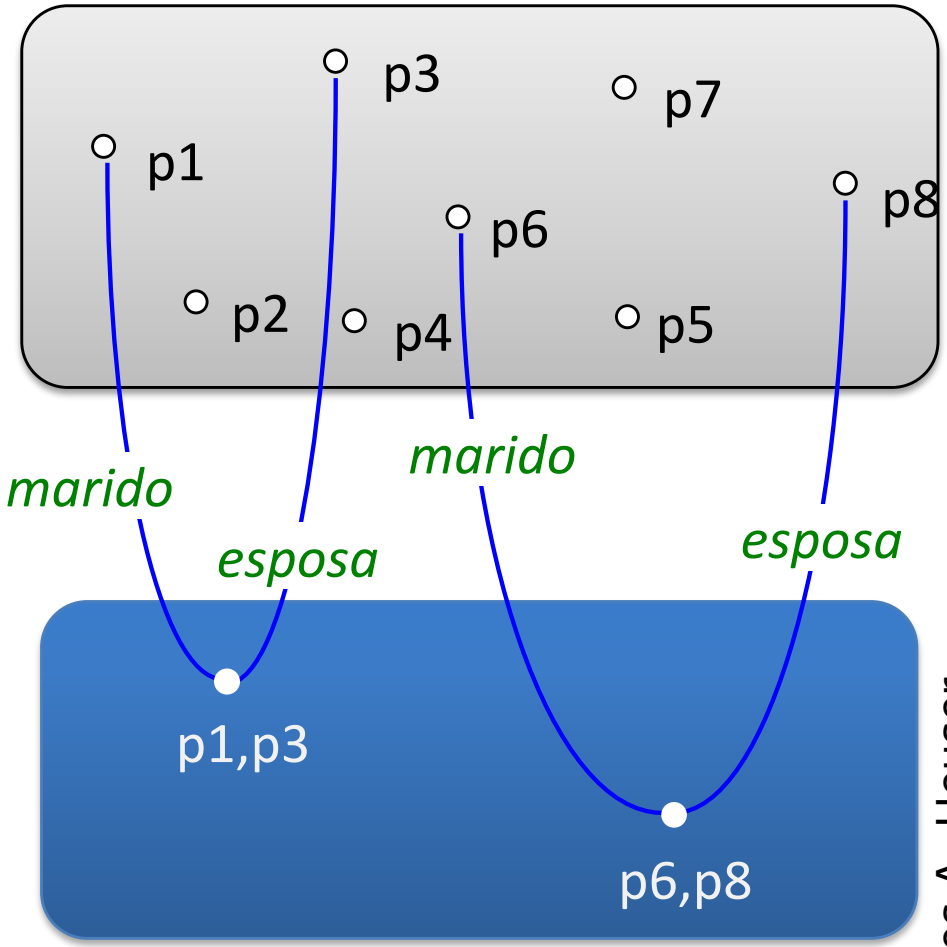
relacionamento
LOTAÇÃO

entidade
DEPARTAMENTO

Auto-relacionamento



Auto-relacionamento: diag. de ocorrências

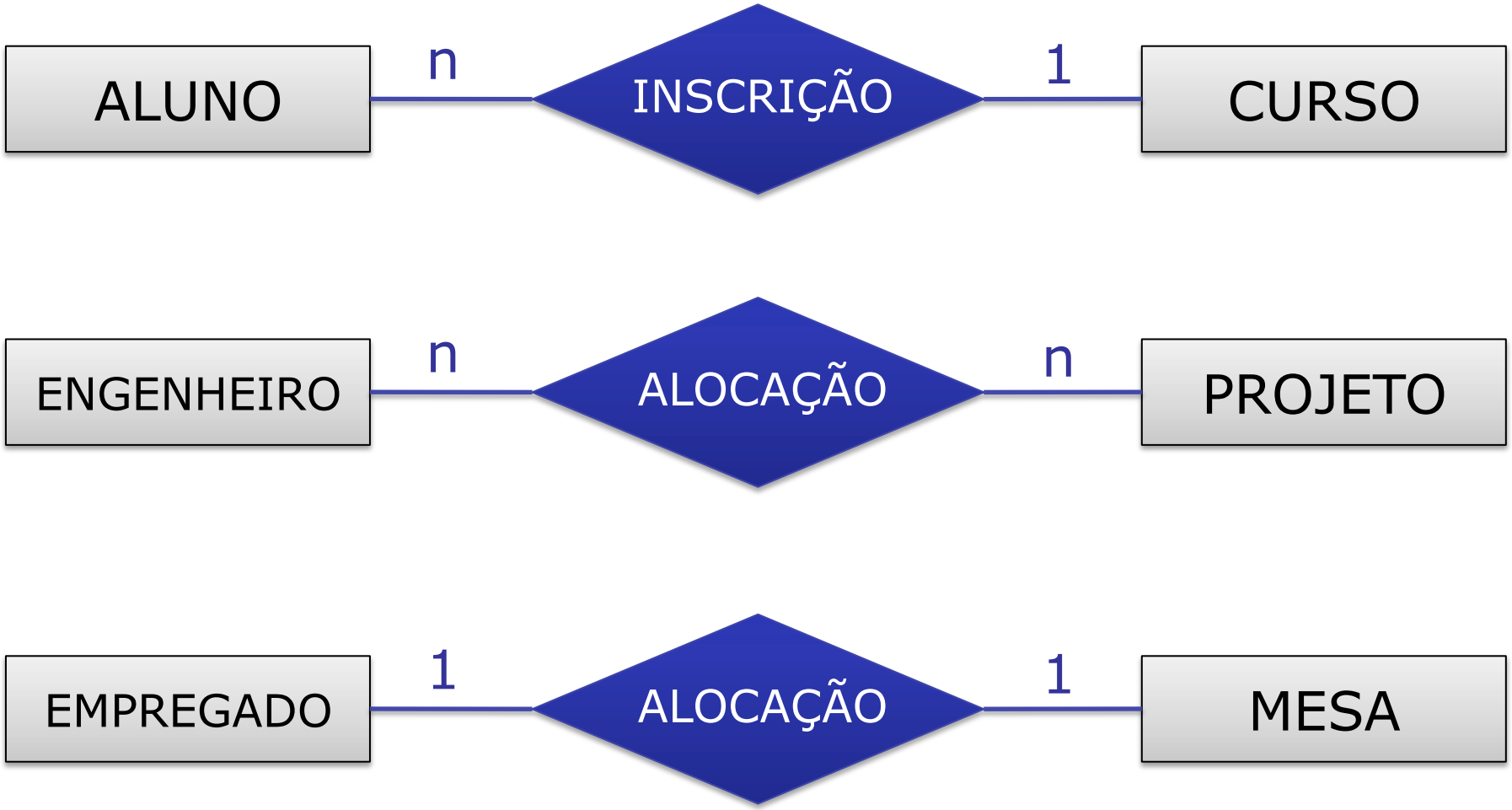


Cardinalidade

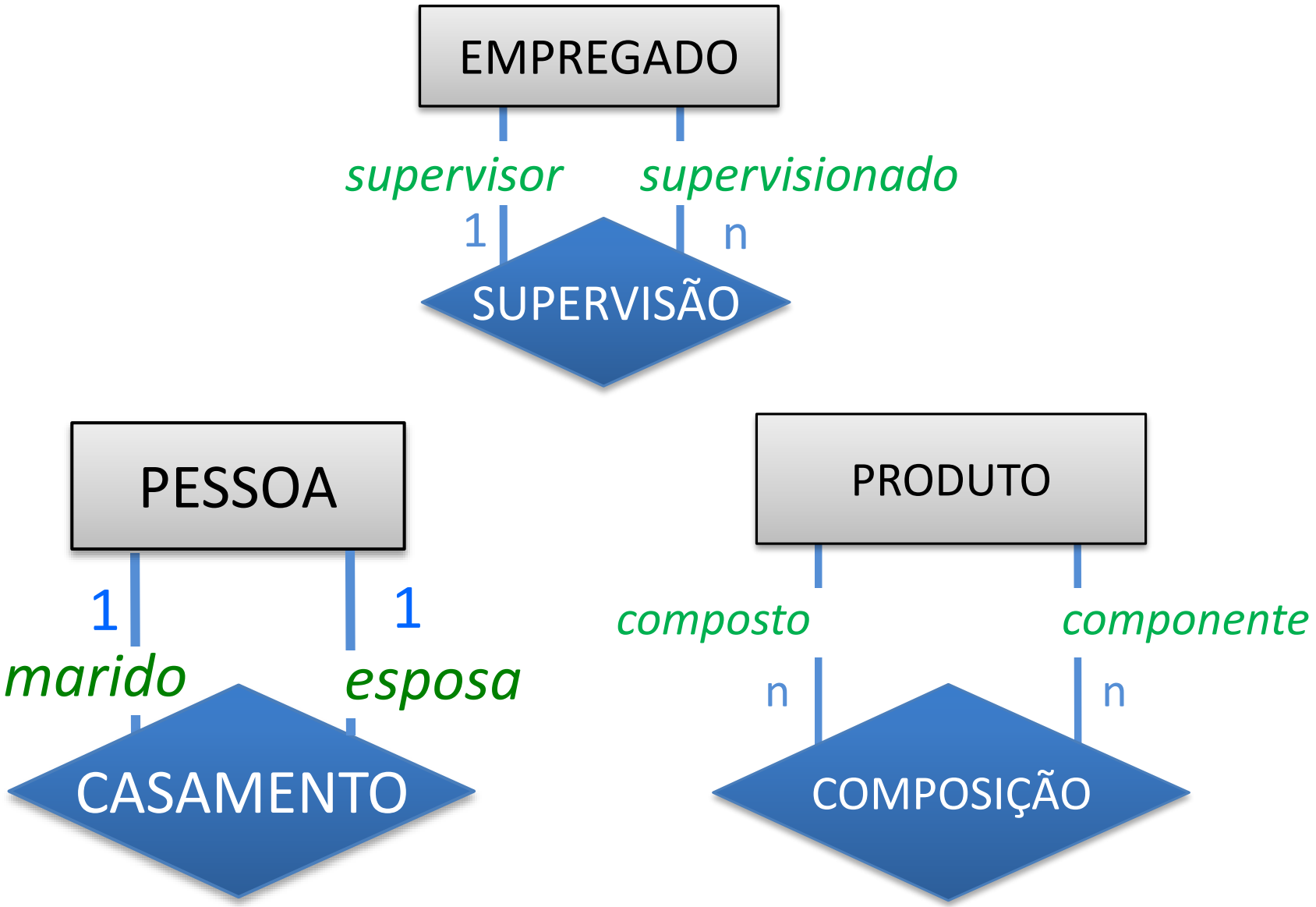
- Número de ocorrências de uma entidade que podem estar associadas através de um relacionamento;
- Em bancos de dados simples, não se distingue entre valores > 1, representando-os simplesmente como “n”:
  - 1-1 (um para um);
  - 1-N (um para muitos);
  - N-N (muitos para muitos).

Relacionamentos binários

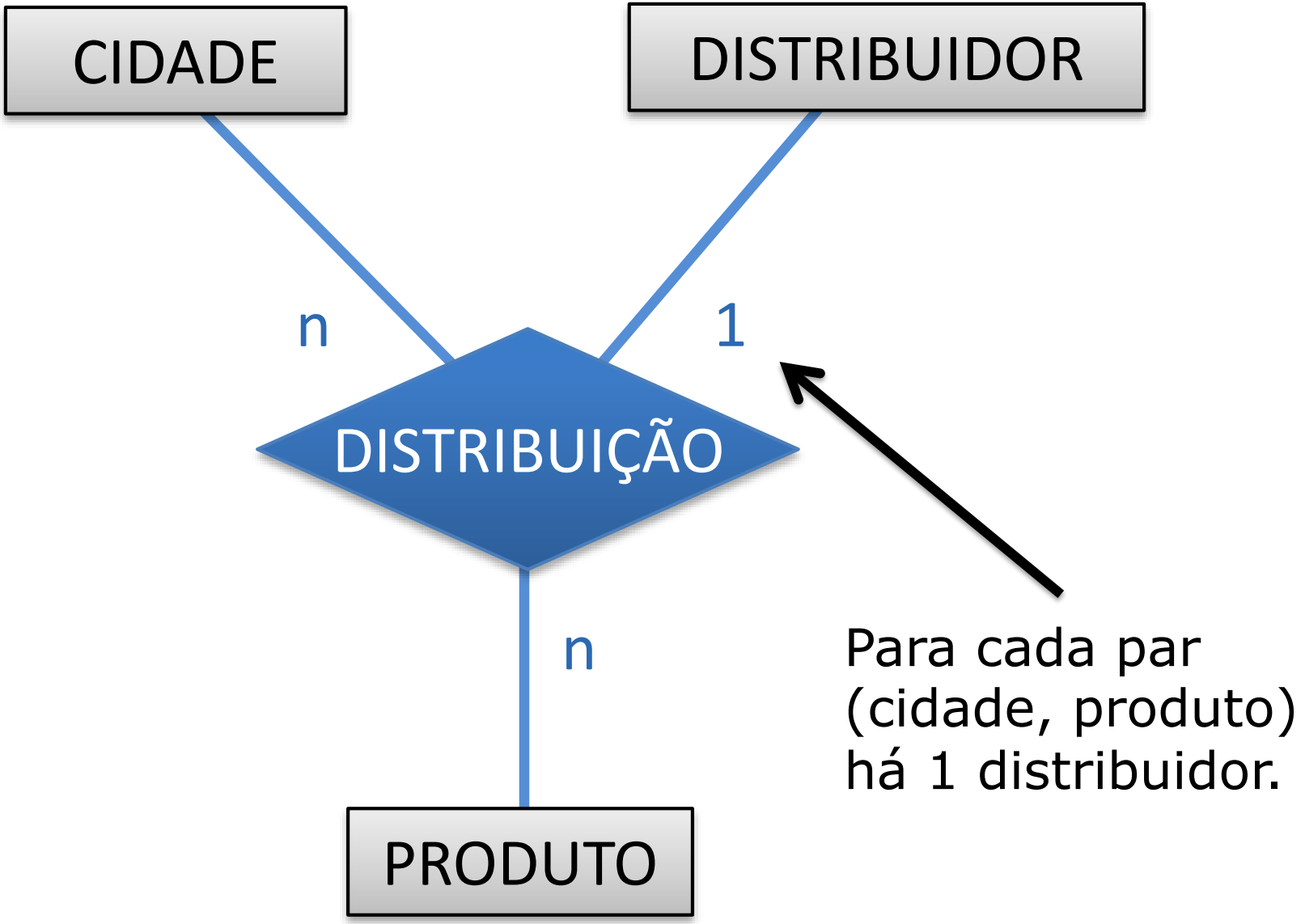
Cardinalidade: exemplos



Cardinalidade: exemplos



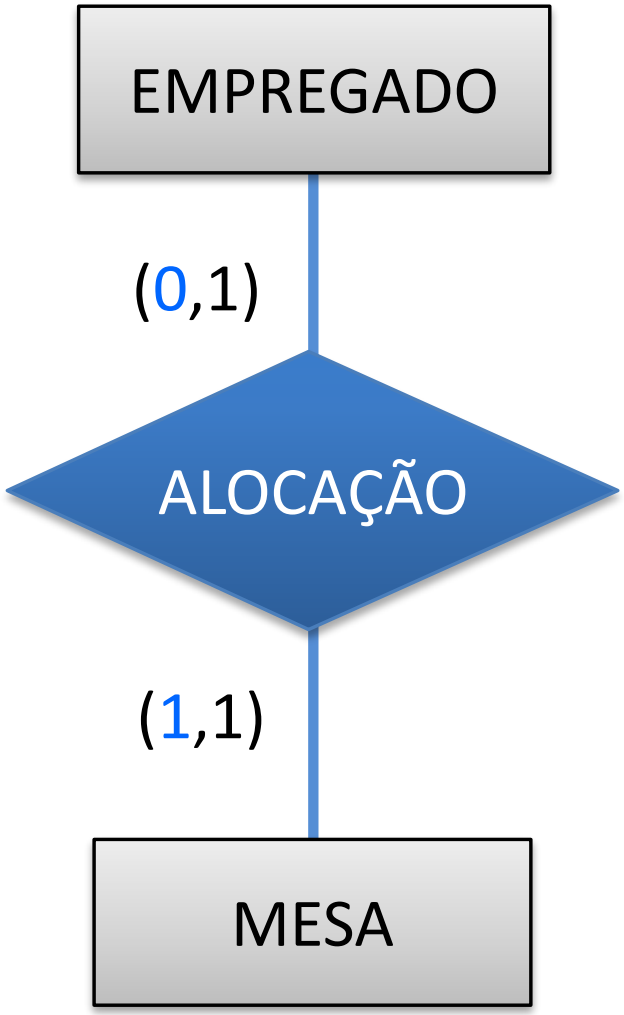
Relacionamento ternário

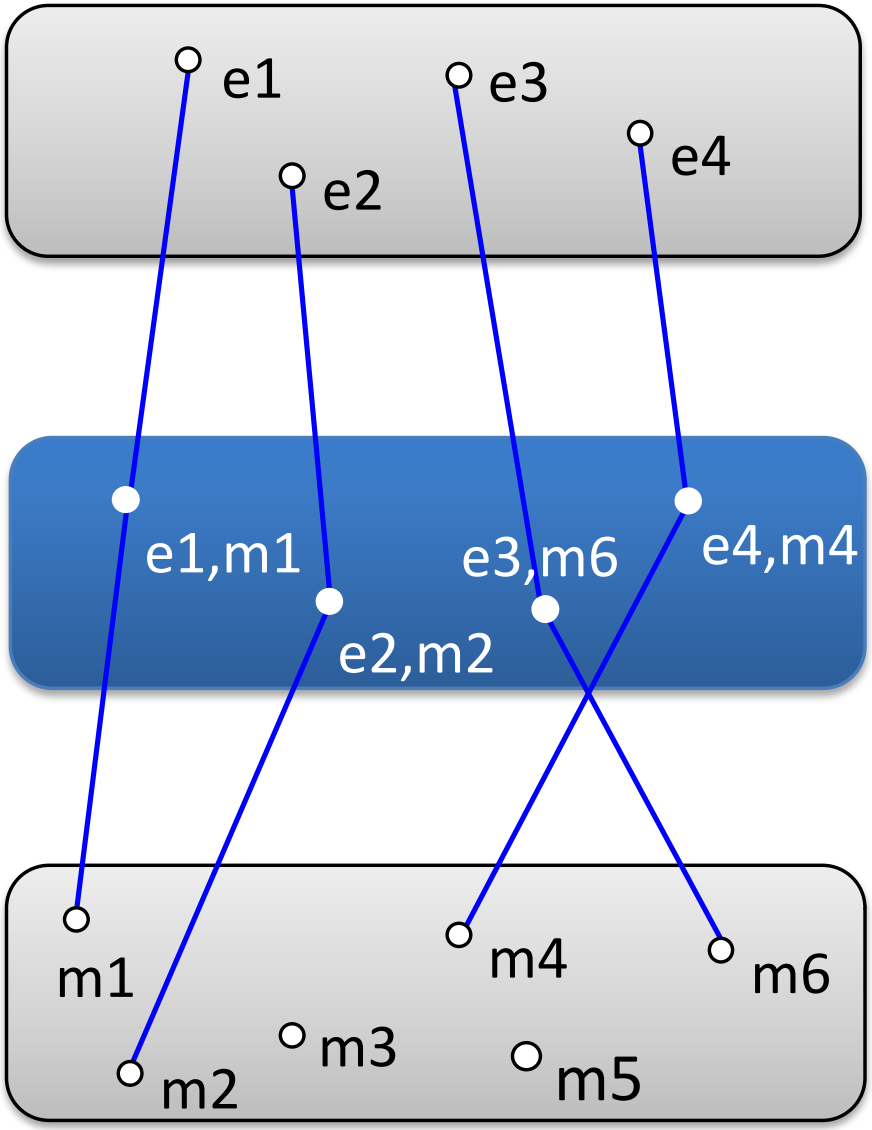


Cardinalidade mínima

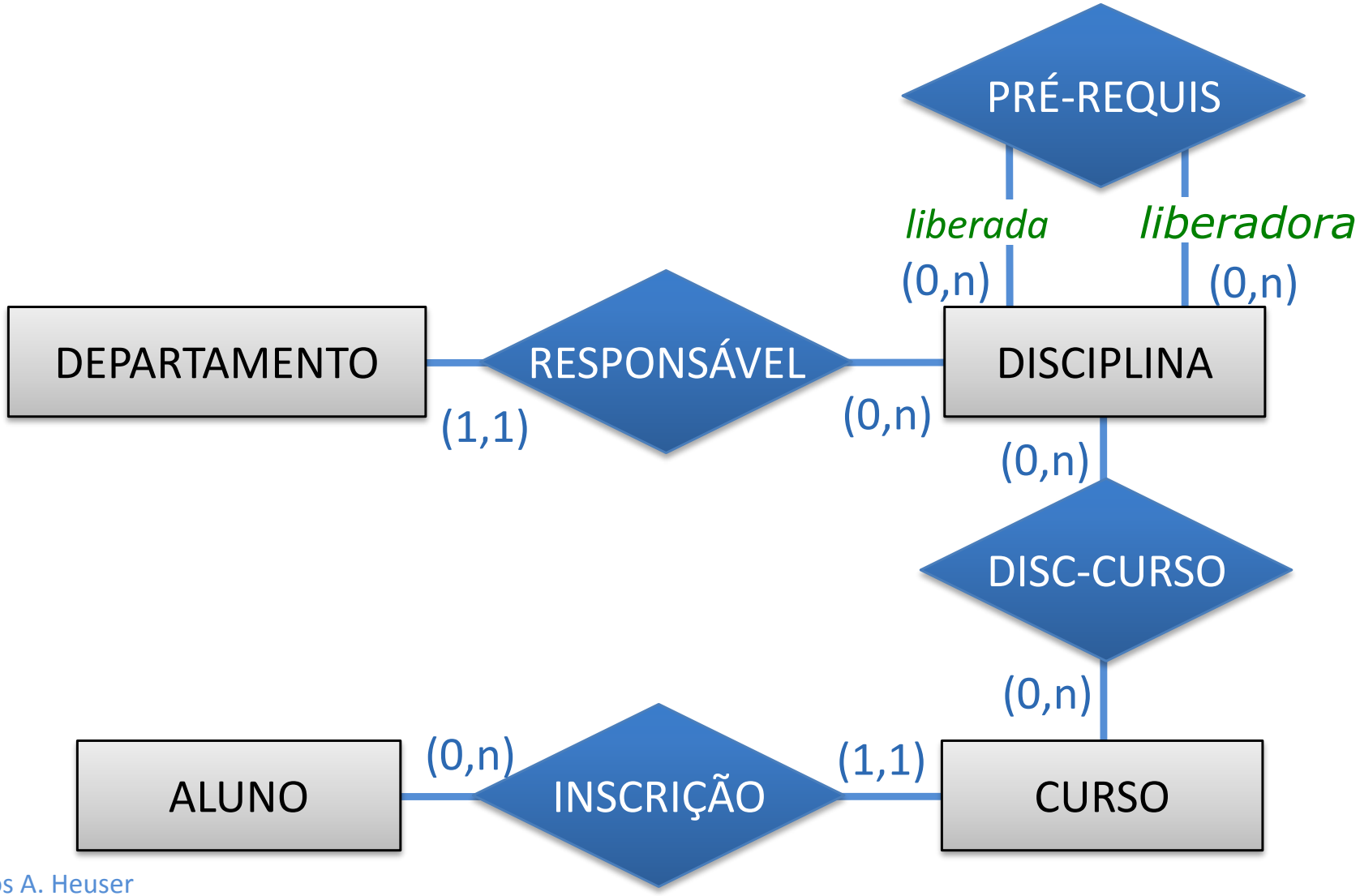
- Número mínimo de ocorrências de entidade que são associadas a uma ocorrência de uma entidade através de um relacionamento;
- Em bancos de dados simples, são consideradas apenas 2 cardinalidades mínimas:
  - 0 (zero): associação opcional;
  - 1 (um): associação obrigatória.

Cardinalidade mínima



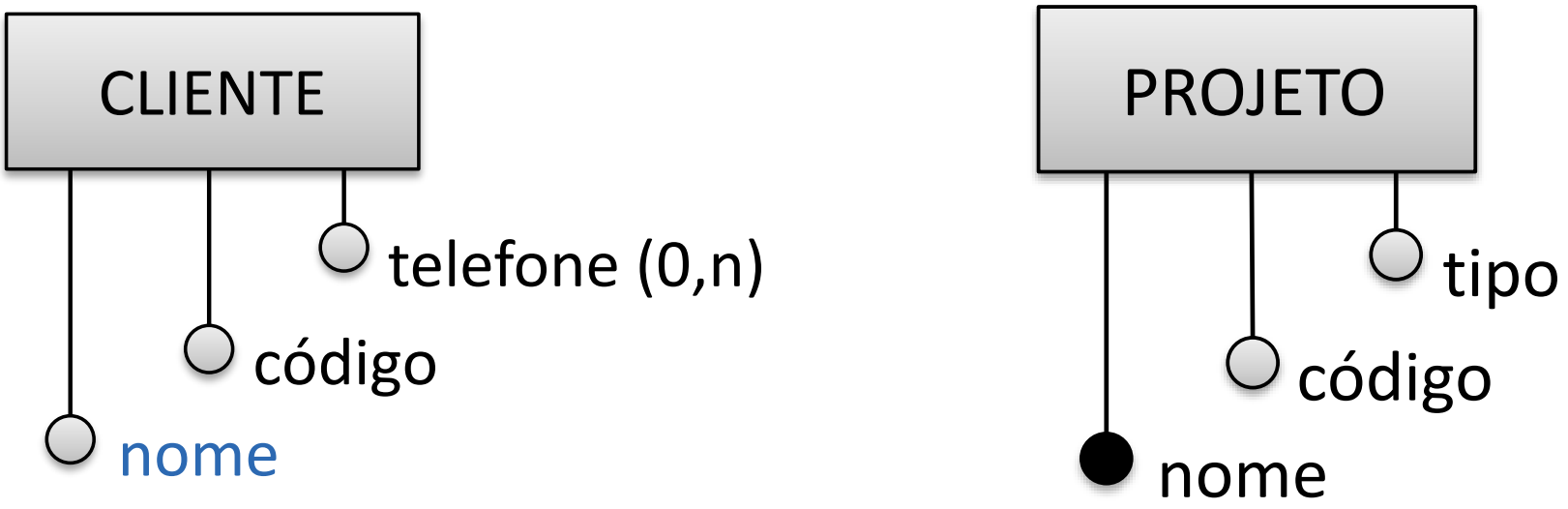


Exemplo

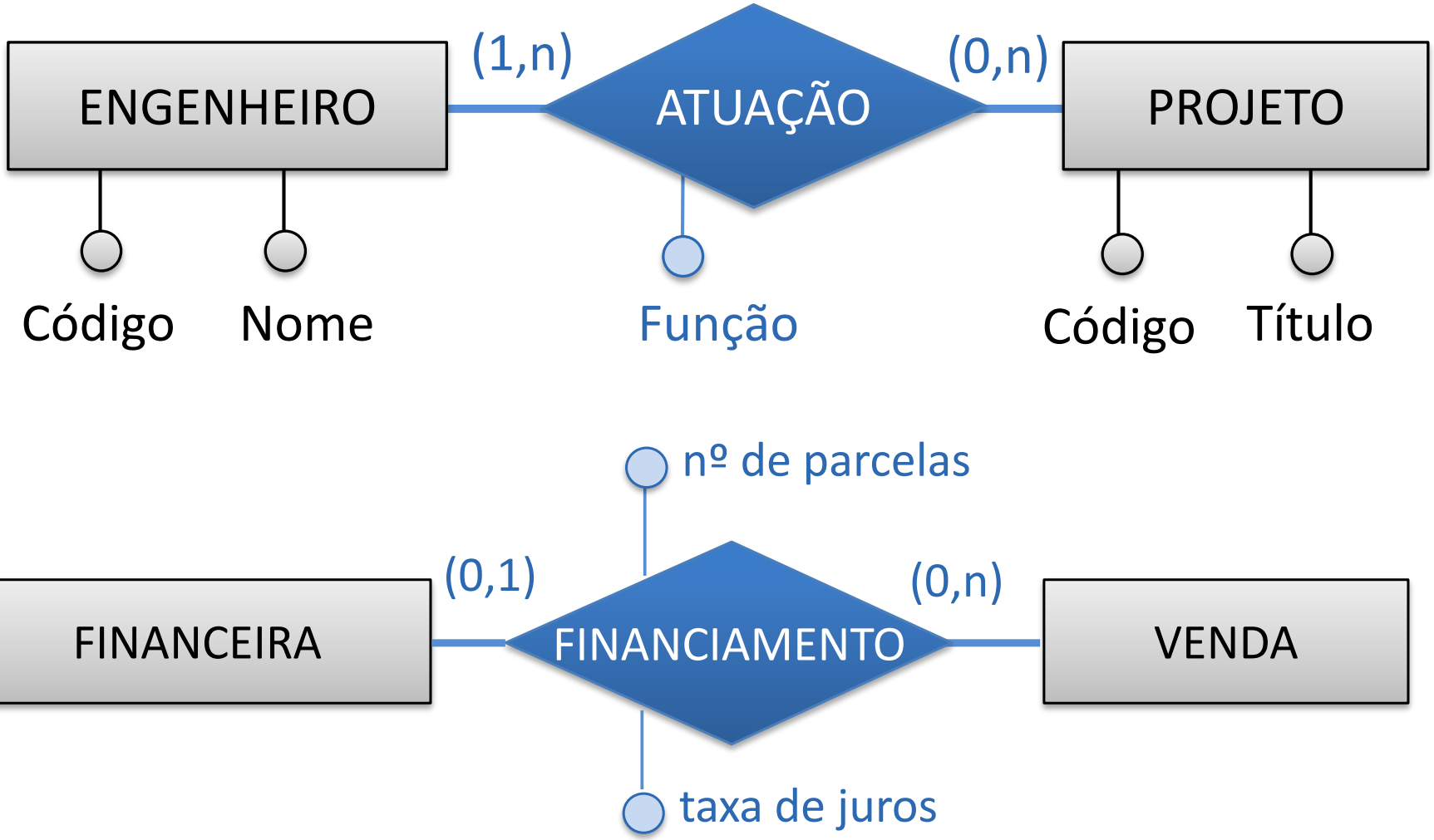


Atributo

- Dado ou informação que é associado a cada ocorrência (instância) de uma entidade ou de um relacionamento;
- Cardinalidades:
  - Mínima: 0 (opcional) ou 1 (obrigatório);
  - Máxima: 1 (mono-valorado) ou n (multivalorado).

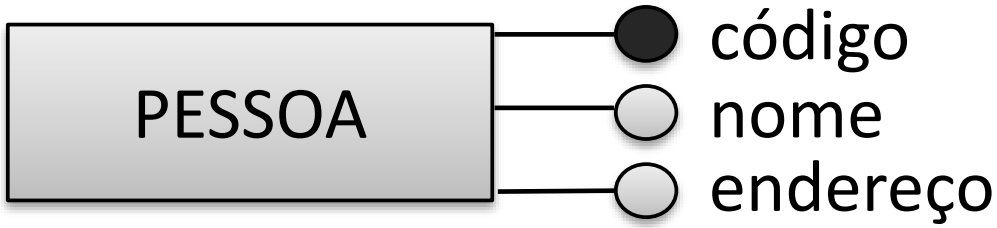


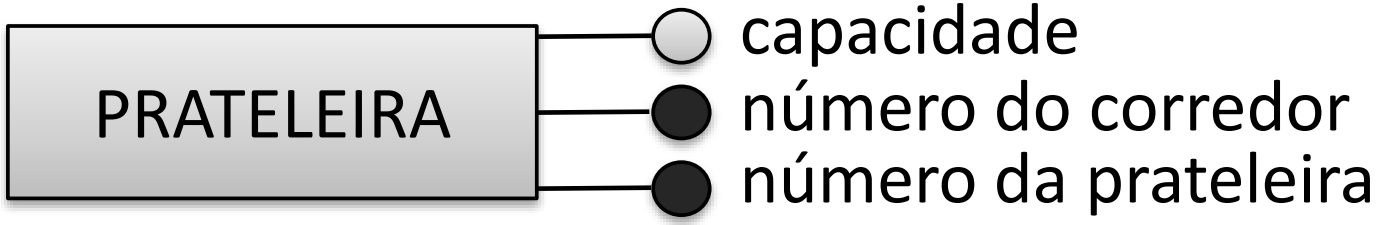
Atributo em relacionamento



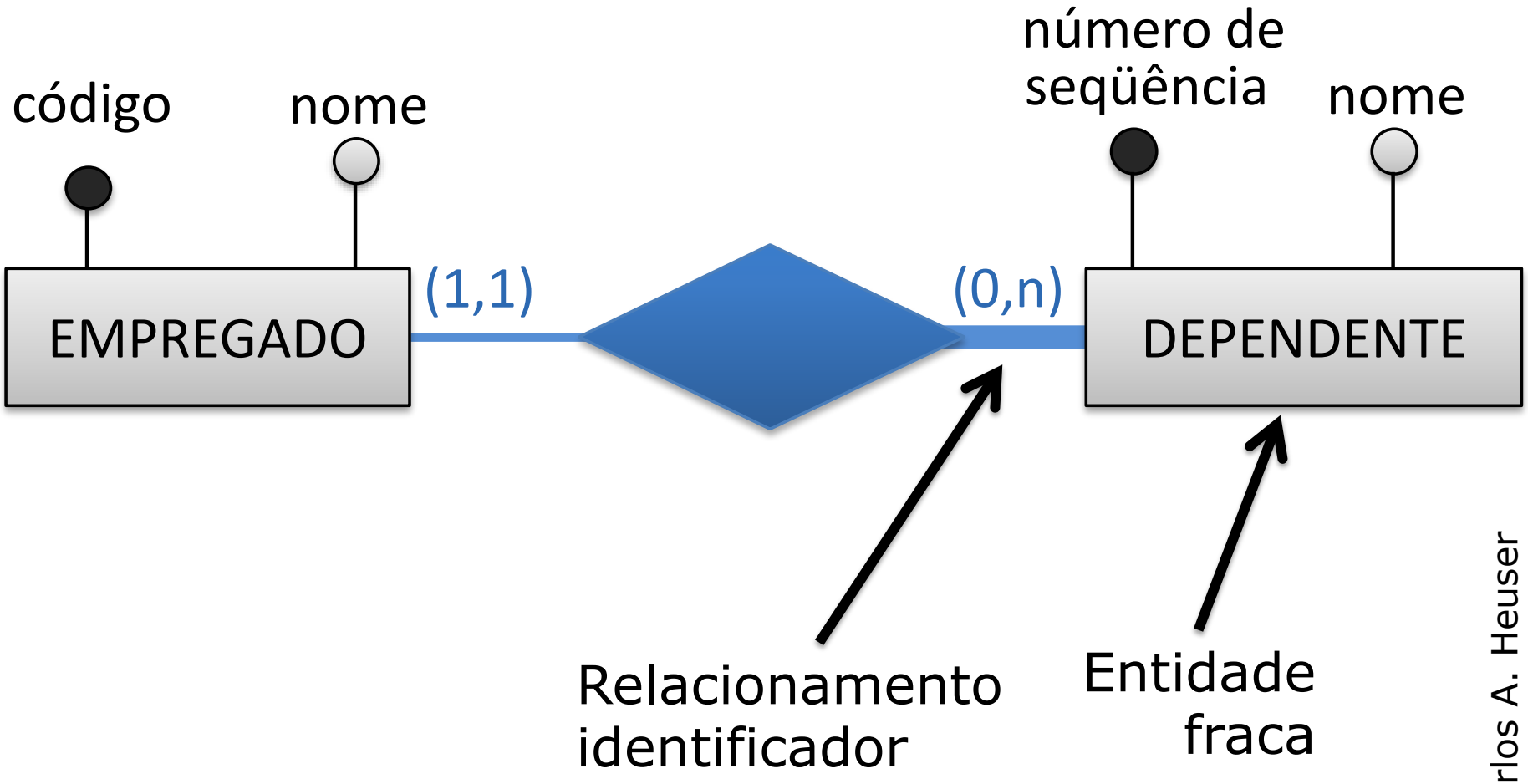
Atributo identificador da entidade

- Conjunto de propriedades (atributos, relacionamentos) de uma entidade cujos valores servem para distinguir uma ocorrência (instância) da entidade das demais;
- Cada entidade deve ter um identificador;

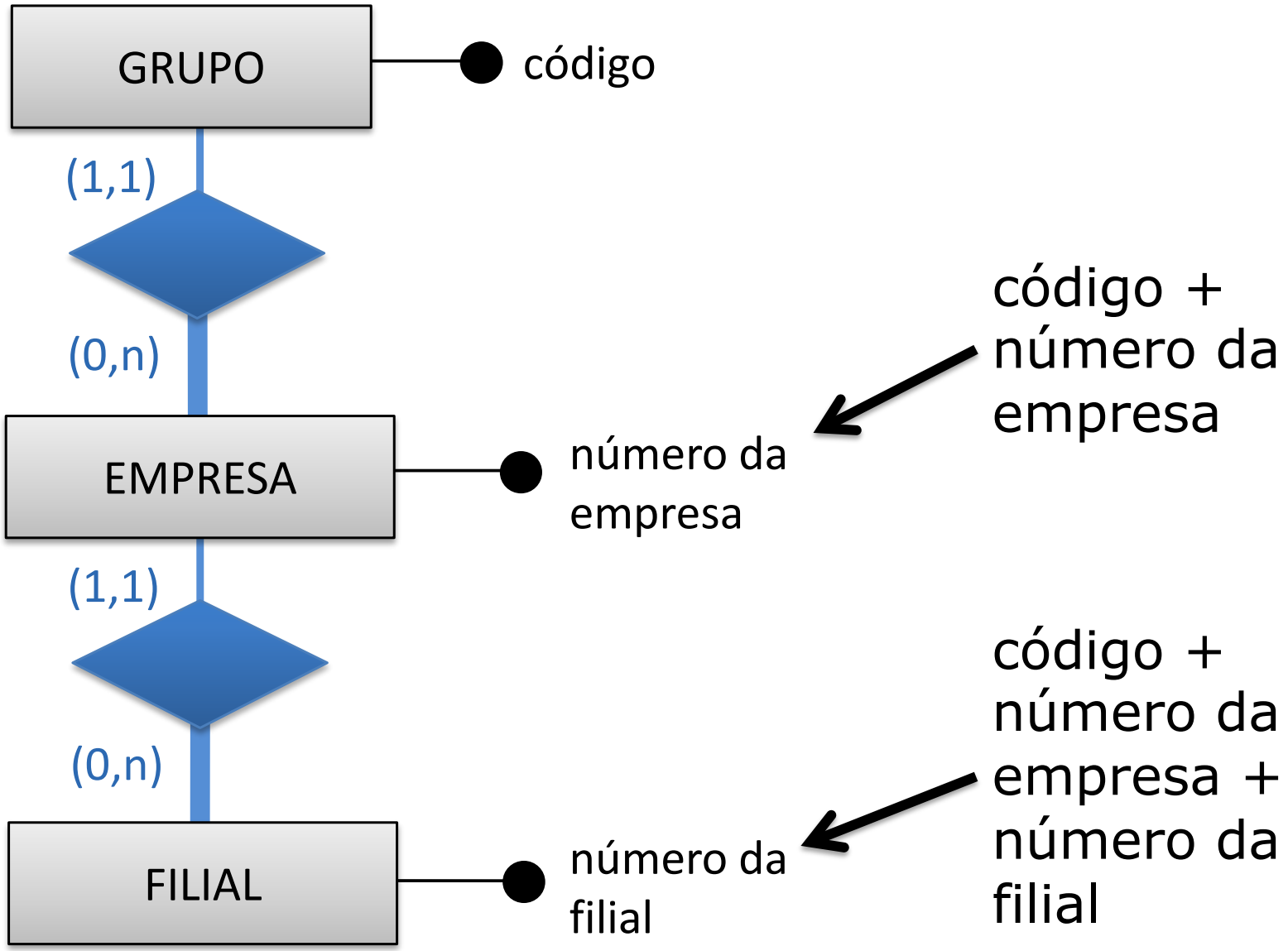




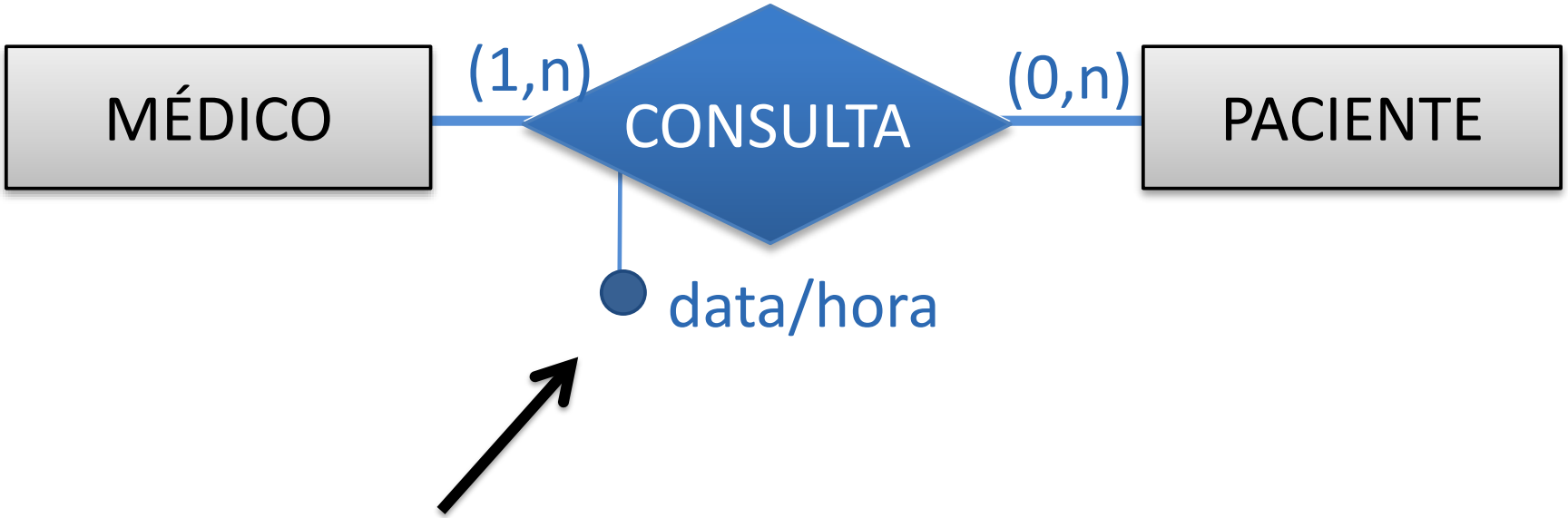
Relacion. identificador / entidade fraca



Recursão do relacionamento identificador



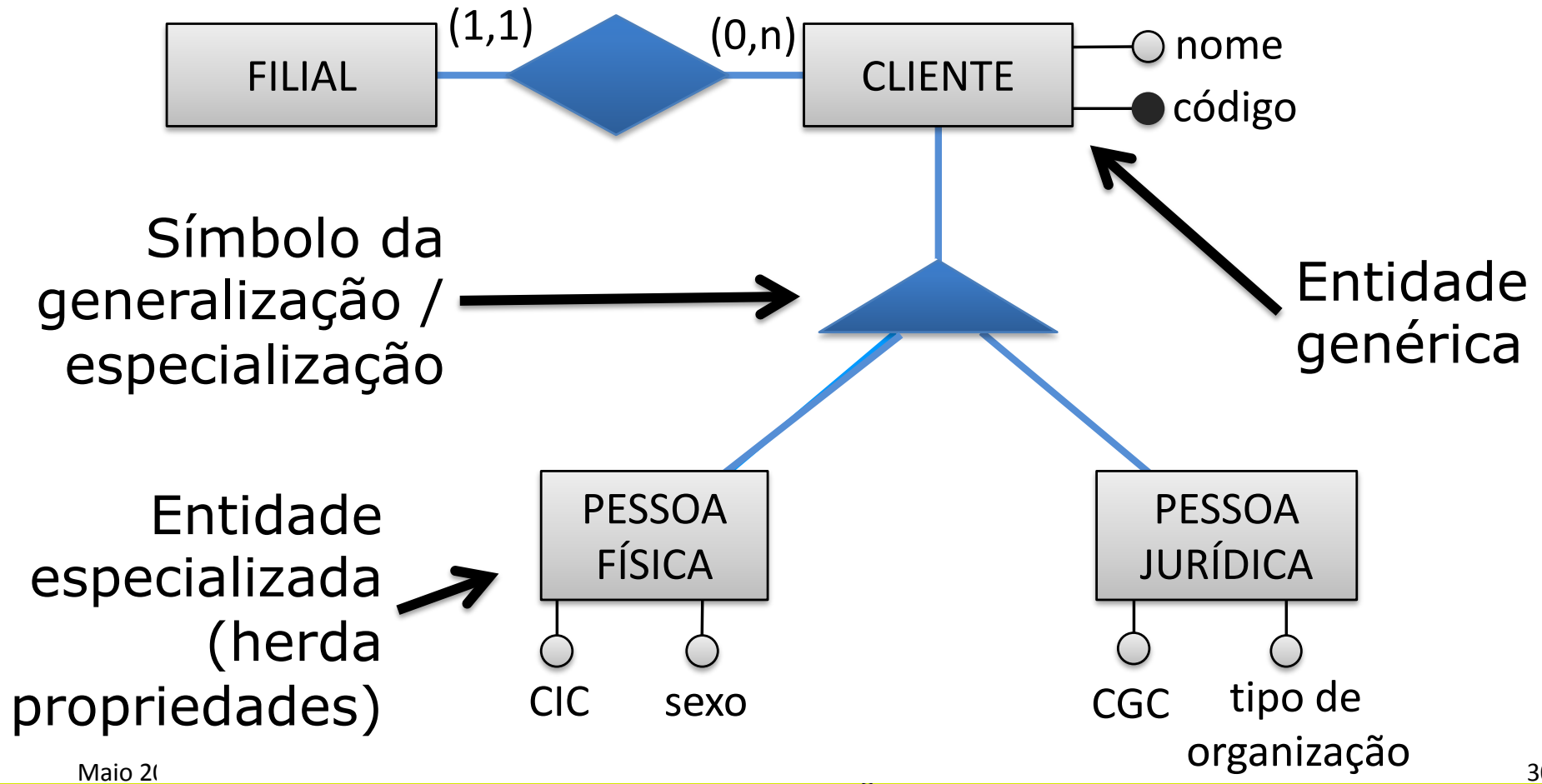
Relacionamento com atributo identificador



Duas instâncias de consulta (par médico-paciente) se distinguem pela data/hora da consulta.

Generalização / especialização

- Permite atribuir propriedades particulares a um subconjunto das ocorrências (especializadas) de uma entidade genérica:



Estruturado ou orientado a objetos?

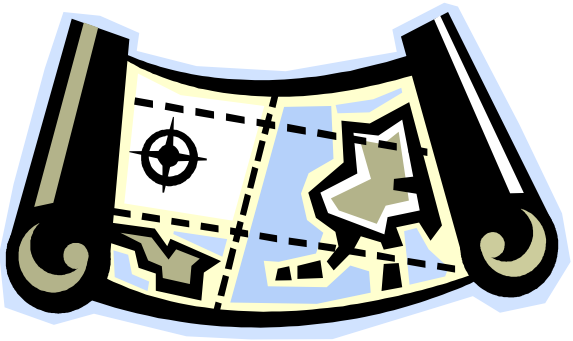
- Estruturado:
  - Modelo entrada – processamento – saída;
  - Dados separados das funções;
  - Visto na disciplina de PBC.
- Orientado a Objetos (OO):
  - O mundo é composto por objetos;
  - Objetos combinam dados e funções;
  - Conceitos do problema são modelados como objetos que são associados e interagem entre si.

Estruturado ou orientado a objetos?

- No estruturado, o gap semântico é maior, o que frequentemente gera sistemas difíceis de manter:
  - As funções tem que conhecer a estrutura dos dados;
  - Mudanças na estrutura dos dados acarreta alteração em todas as funções relacionadas.
- O paradigma orientado a objetos vem substituindo-o:
  - Melhoria da interação analistas x especialistas;
  - Apoio à reutilização, extensão, legibilidade;
  - O mundo é composto por objetos...
- Importância da consistência entre os modelos.

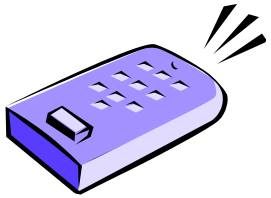
Conceitos OO: abstração

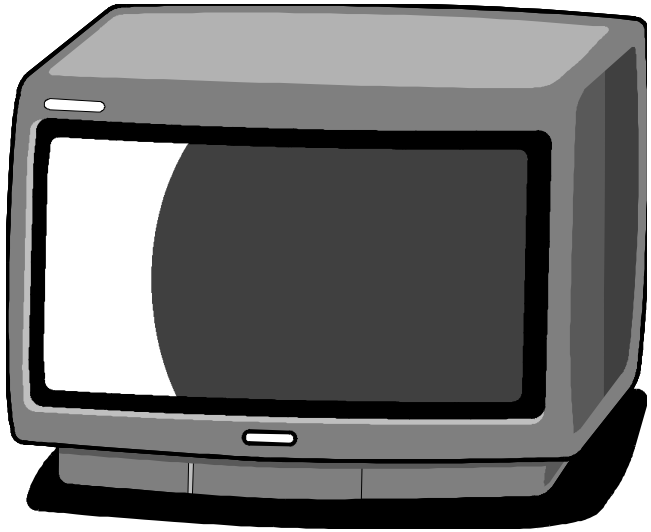
- "Modelos mentais": visão simplificada do mundo construída por cada um em cada situação;
- Abstrair consiste em ignorar aspectos irrelevantes e concentrar nos principais.



Conceitos OO: encapsulamento

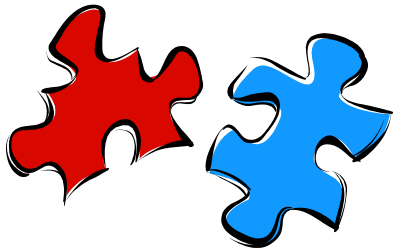
- Separar os aspectos externos (o que faz) dos aspectos internos (como faz):
  - Aspectos externos = interface, contrato;
  - Aspectos internos = implementação.

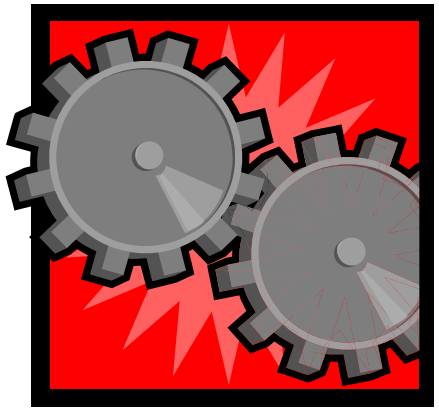




Conceitos OO: modularidade

- Decomposição do sistema em módulos:
  - Coesos (baixo acoplamento);
  - Autônomos;
  - De interface simples e coerente.
- Fundamental para o reuso e extensão.



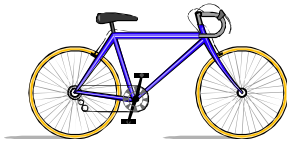


Conceitos OO: hierarquia

- É uma forma de arrumar as abstrações e simplificar o entendimento do problema;
- Também promovem o reuso;
- Sinergia para administrar a complexidade:
  - Abstração auxilia a identificar os conceitos relevantes do mundo real;
  - Encapsulamento oculta a visão interna das abstrações identificadas;
  - Modularidade nos dá um meio de agrupar logicamente abstrações relacionadas;
  - Por fim, abstrações formam hierarquias.

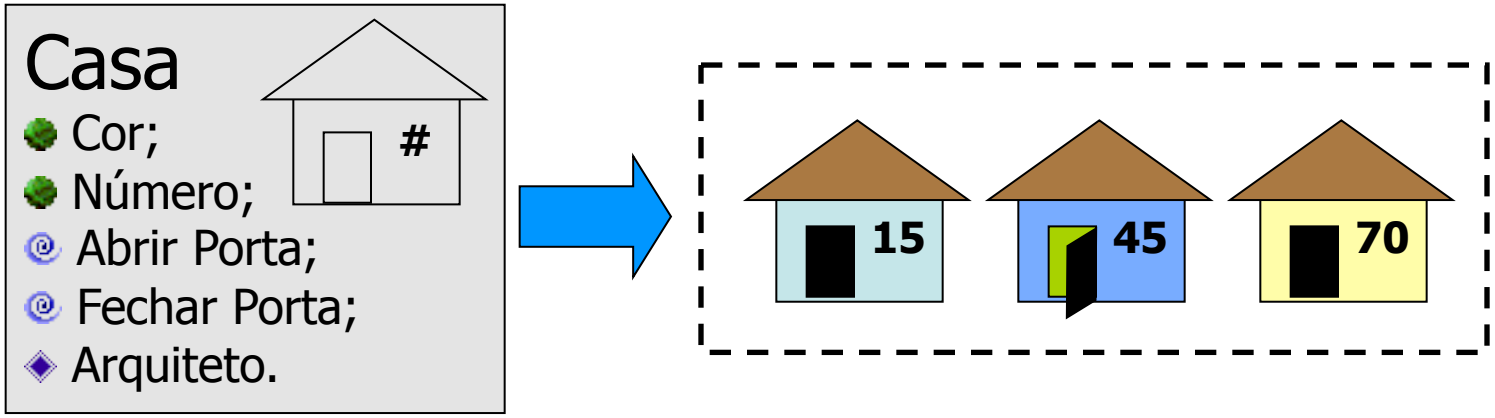
Conceitos OO: objetos

- "Um objeto é uma entidade que incorpora uma abstração relevante no contexto de uma aplicação";
- Podem ser coisas abstratas (ex.: uma reserva de passagem aérea) ou concretas (ex.: um documento).



Conceitos OO: classes

- Uma classe descreve um conjunto de objetos com as mesmas propriedades, o mesmo comportamento, os mesmos relacionamentos com outros objetos e a mesma semântica;
- Similar ao conceito de tipo.

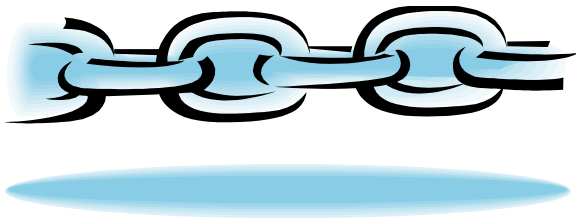


Conceitos OO: classes e instâncias

- Objeto = Instância de classe;
- Paradigma OO norteia o desenvolvimento por meio de classificação de objetos:
  - Modelamos classes, e não objetos;
  - Objetos são entidades reais – executam algum papel no sistema;
  - Classes são abstrações – capturam a estrutura e comportamento comum a um conjunto de objetos.

Mecanismos de estruturação OO

- Objetos relacionam-se uns com os outros;
- É preciso modelar esta complexidade e estruturar as classes;
- Mecanismos propostos:
  - Associação;
  - Composição;
  - Herança.

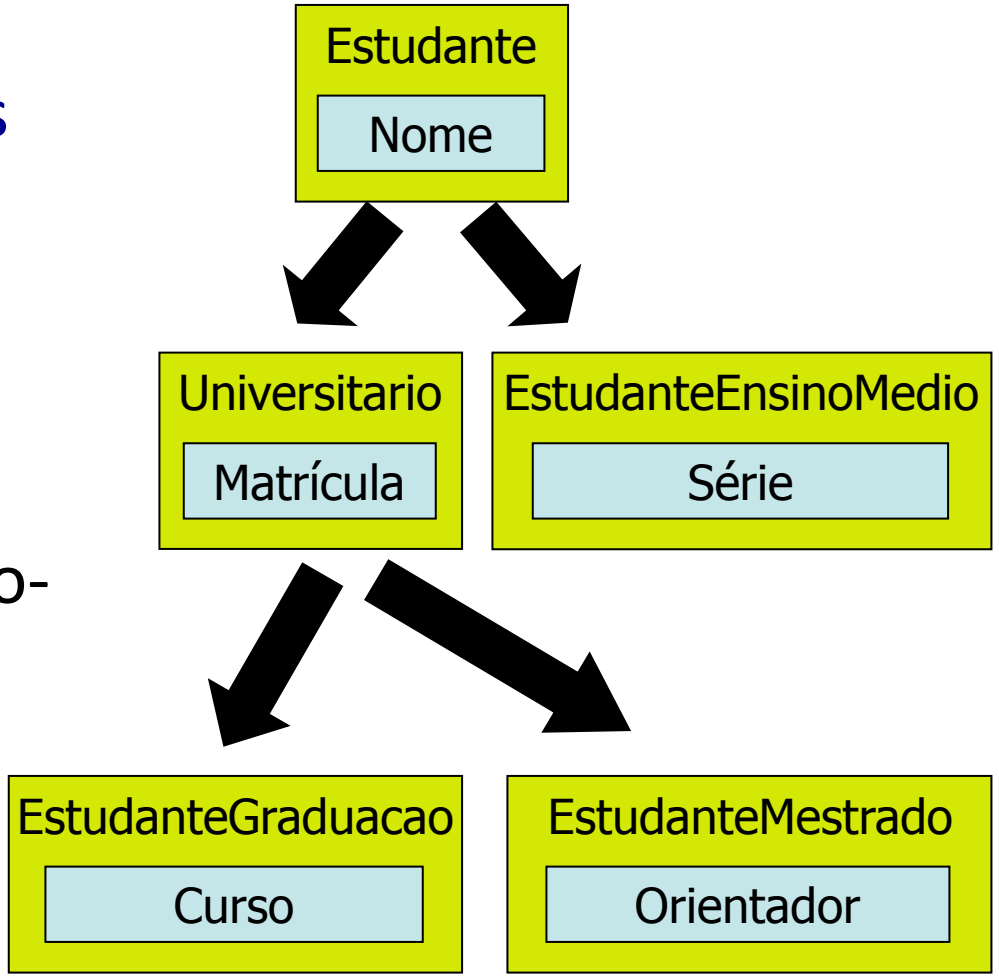


Conceitos OO: ligações e associações

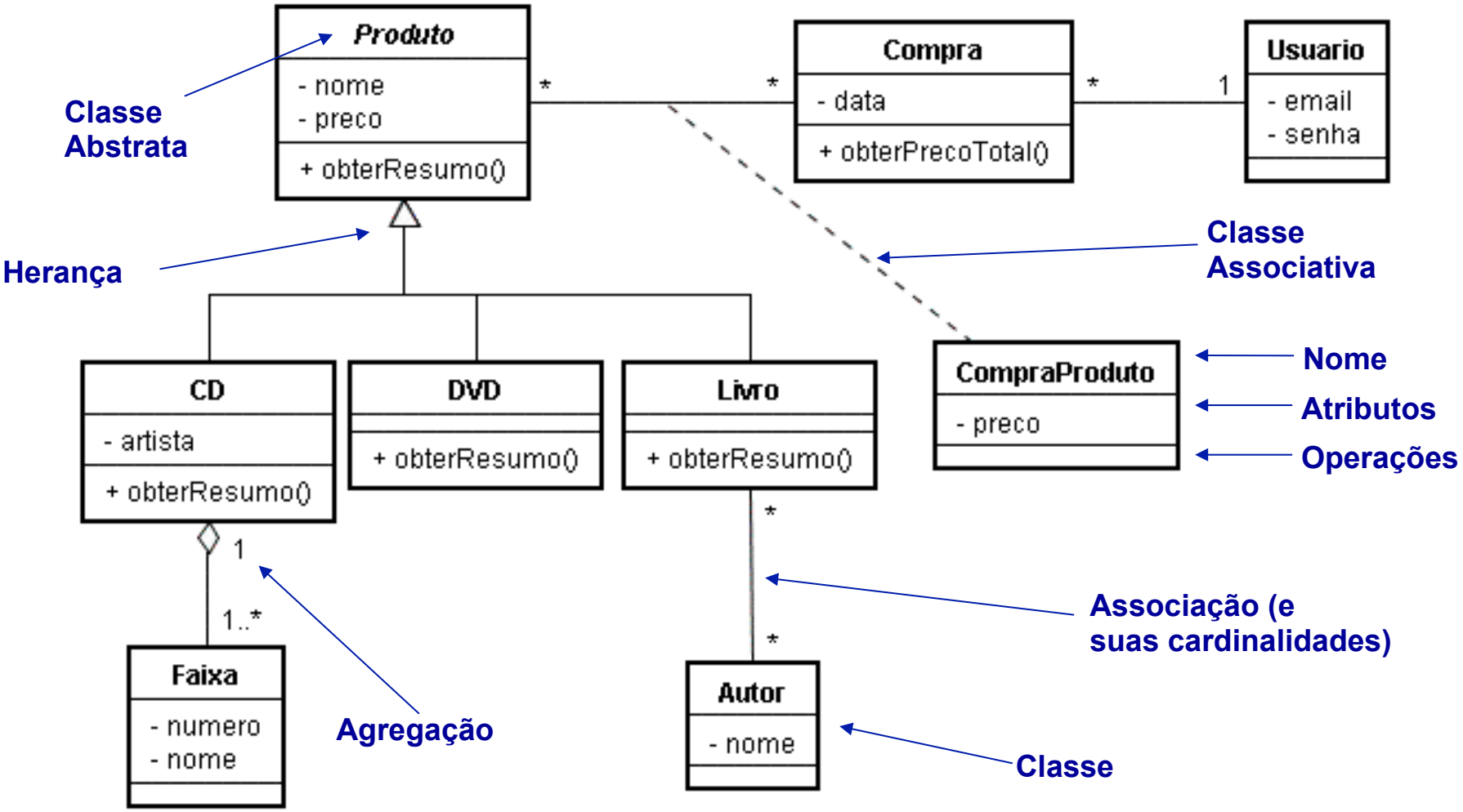
- Ligação: conexão entre objetos;
- Associação: conexão entre classes que representa a existência de ligações;
- Uma associação descreve um conjunto de potenciais ligações da mesma maneira que uma classe descreve um conjunto de potenciais objetos [Rumbaugh].

Conceitos OO: herança

- Generalização: quando classes têm semelhanças podemos definir uma classe mais geral;
- Especialização: muitas vezes um conceito pode ser refinado, adicionando-se novas características.



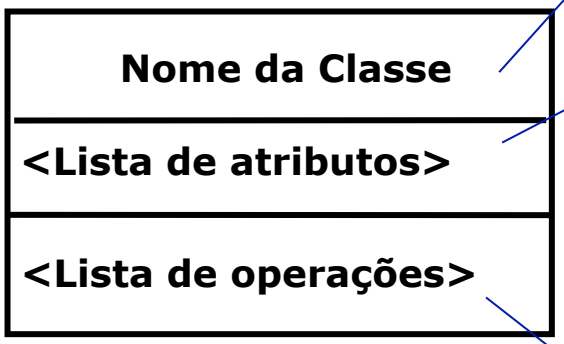
O diagrama de classes da UML



Representa as classes relevantes (abstração!) para o domínio, problema ou solução.

Representação de classes

Representação em UML



Se estiver em itálico, a classe é abstrata.

Sintaxe: <escopo> <nome> : <tipo> = <valor default>
Escopo:
- privado
+ público
# protegido

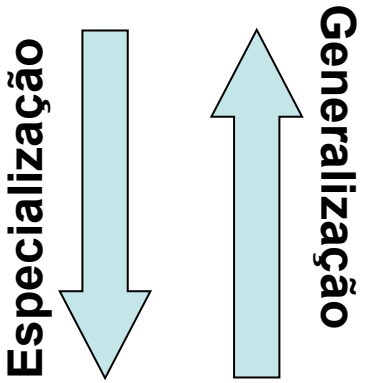
Sintaxe: <escopo> <nome> (<parâmetros>) : <tipo>

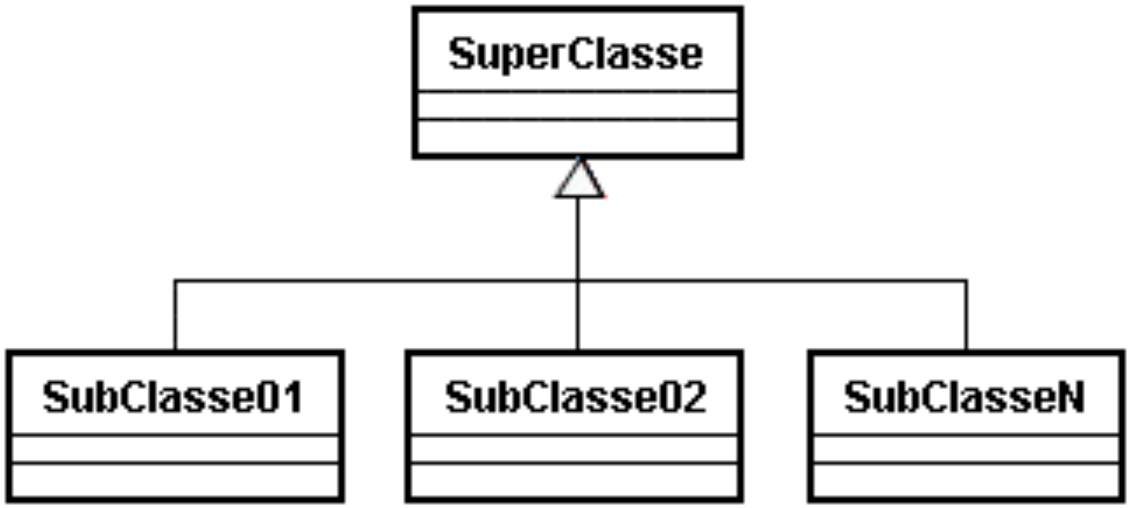
<parâmetros> = lista de pares "<nome> : <tipo>", separada por vírgula.

Dependendo do nível de abstração, alguns detalhes podem ser omitidos (ex.: tipo e escopo na fase de análise).

Herança (inheritance)

- Devem modelar relações “é-um-tipo-de”;
- Subclasses devem suportar toda a funcionalidade das superclasses e possivelmente mais;
- Funcionalidade comum a diversas classes deve estar o mais alto possível na hierarquia;
- Classes abstratas não podem herdar de classes concretas.



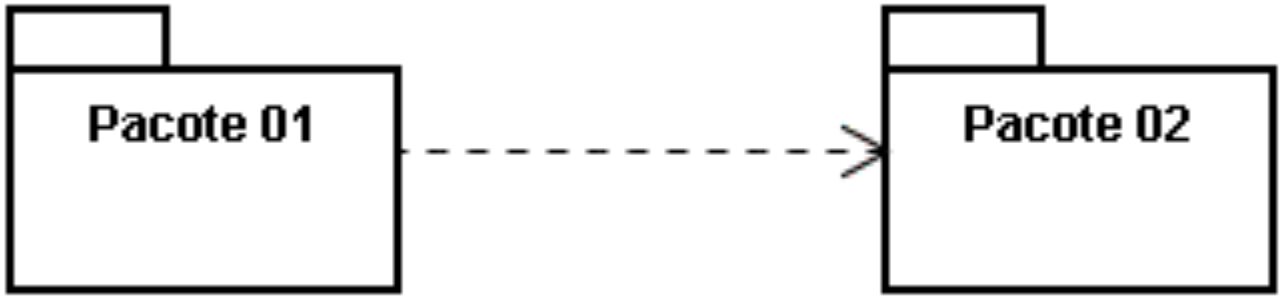


Separação em subsistemas / módulos

- Projetos grandes podem conter centenas de classes e estruturas diversas;
- Divisão das classes em pacotes:
  - Coleção de classes que colaboram entre si;
  - Conjunto coeso de responsabilidades;
  - "Caixa preta".
- Vantagens:
  - Facilita o entendimento para leitores;
  - Auxilia na organização de grupos de trabalho;
  - Organiza a documentação;
  - Em suma, facilita a manutenção.

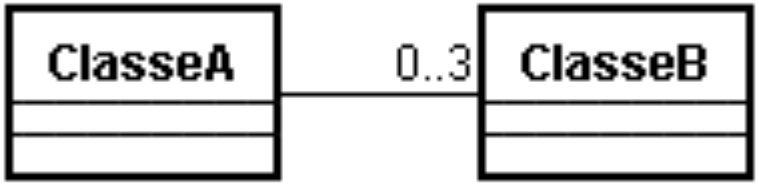
Pacotes (packages)

- Podem ser usados para organizar diversos tipos de elementos de modelos, inclusive diagramas inteiros;
- Muito utilizados para organizar classes em módulos, da mesma forma que será feito em Java/C++;
- É possível representar relação de dependência entre pacotes:

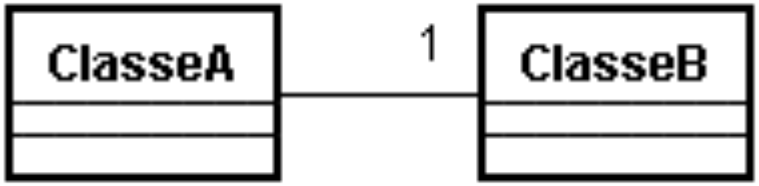


Associações (associations)

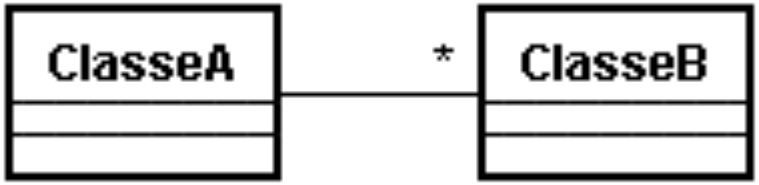
- Relacionamento entre classes é representado por associações, agregações e composições;
- Associações podem indicar cardinalidade (cardinality):



Objetos da ClasseA podem se relacionar com no mínimo zero e no máximo três objetos da ClasseB.



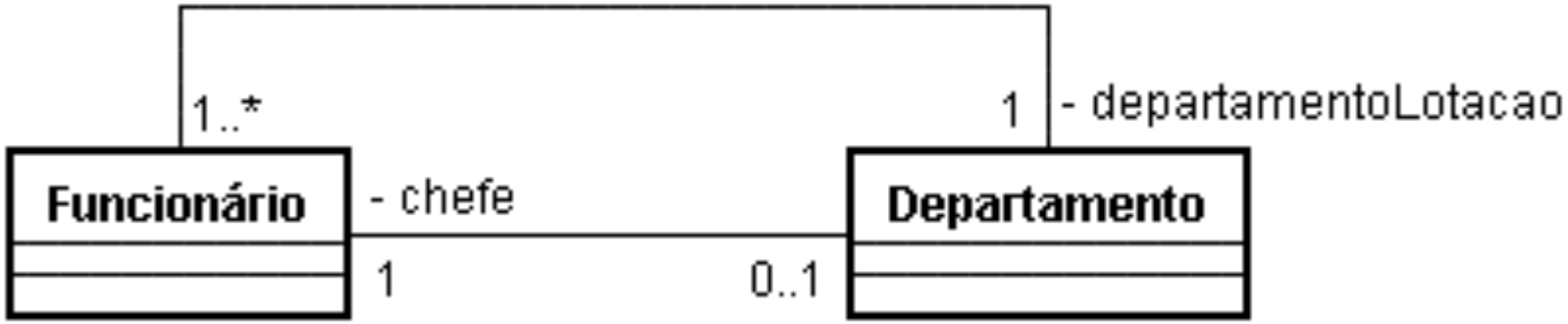
Um e somente um.



Nenhum, um, ou vários.

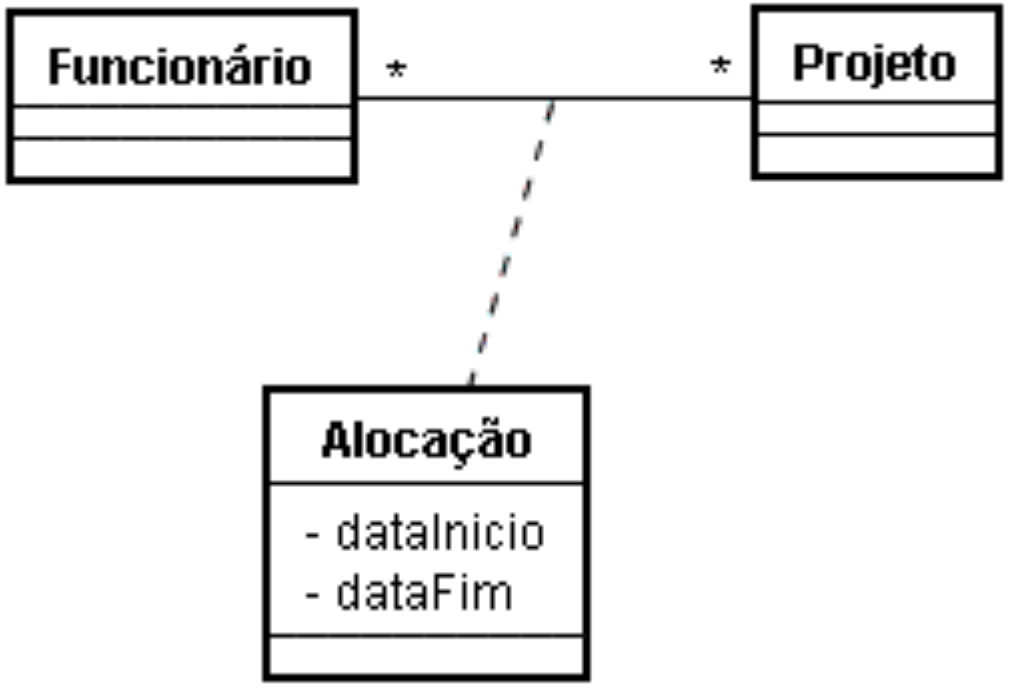
Papéis (roles)

- Indicam o papel que a classe desempenha na associação (são usados substantivos);
- É opcional, usado quando melhora o entendimento do modelo;
- Sintaxe: <escopo> <nome>.



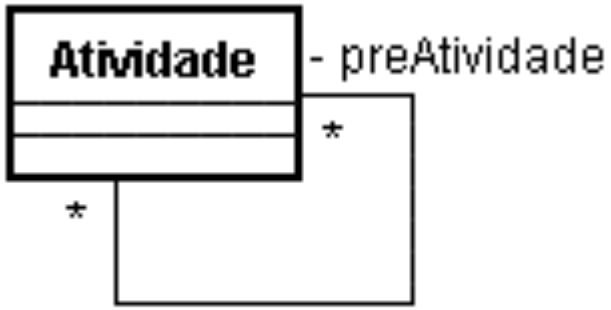
Classes associativas (association class)

- Utilizadas quando a associação possui atributos;
- Comuns em relações n-para-n.



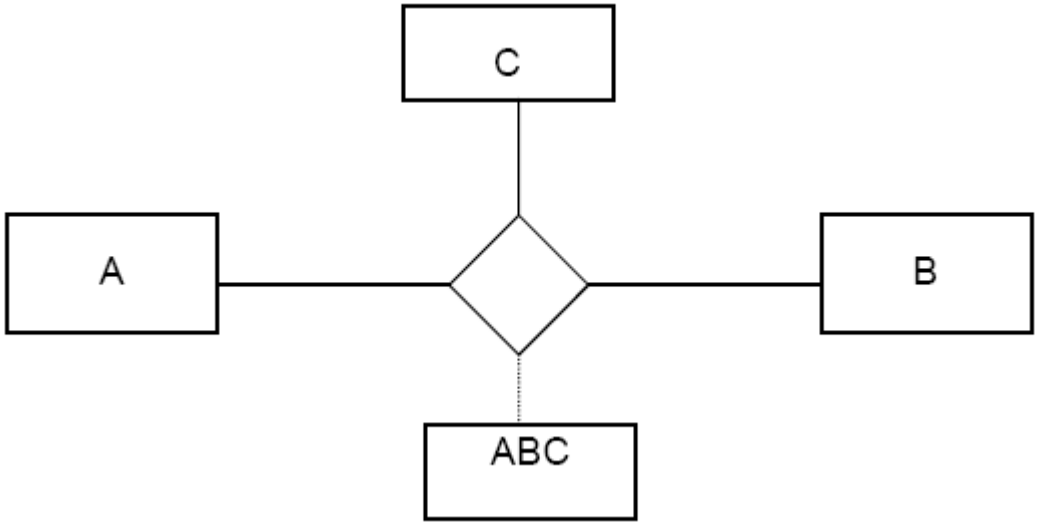
Relacionamentos recursivos

- Perfeitamente legais;
- Geralmente pedem definição de papéis.



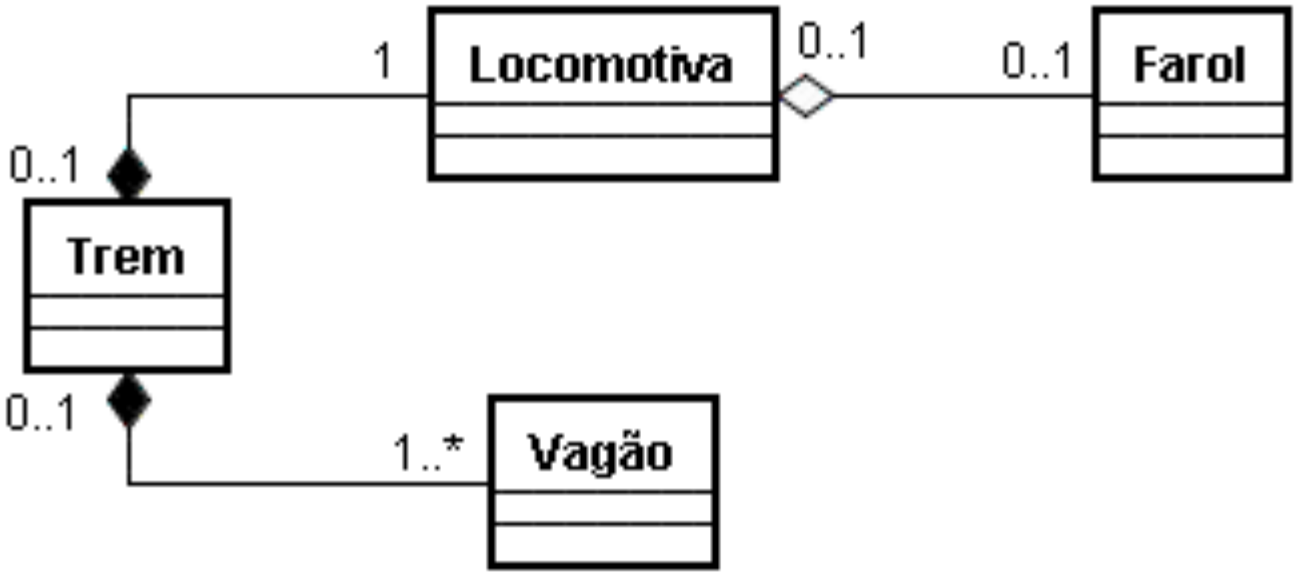
Associações n-árias

- Associações entre três ou mais classes;
- Extremamente raras, muitas vezes as ferramentas CASE nem dão suporte;
- Podem ser substituídas por uma nova classe e N associações.



Agregação e composição

- Representam associações todo-parte;
- Adicionam um losango à sintaxe, na extremidade da classe que representa o todo:

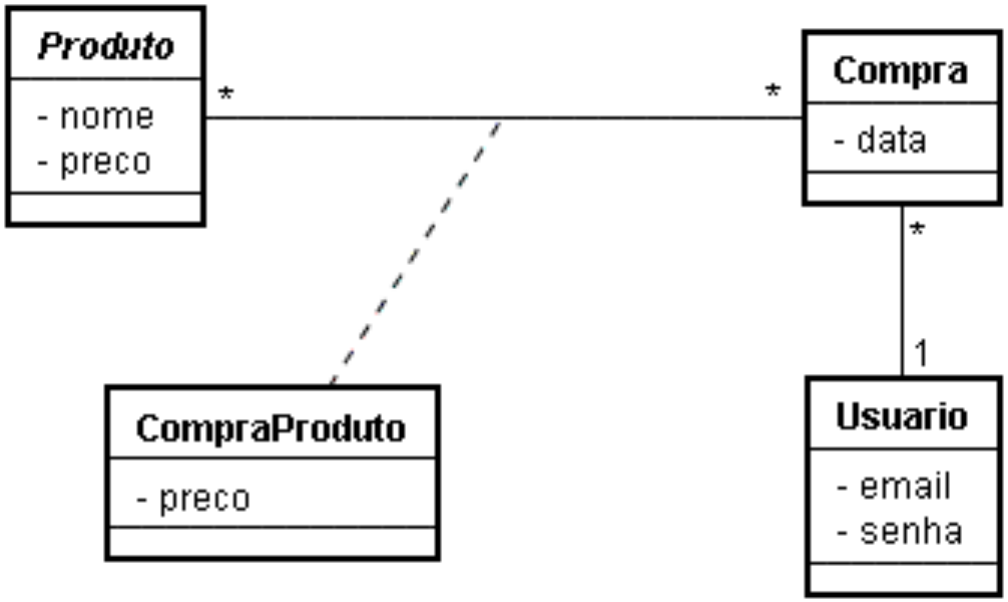


Atributos (attributes)

- Atributos são informações de estado (propriedades) para o qual cada objeto em uma classe tem seu valor;
- Muito similares às associações:
  - Como atributos têm um tipo, podemos considerar que são associações com um tipo;
  - Para tipos primitivos definimos atributos, do contrário modelamos uma associação;
  - Em última instância, associações e atributos são implementados da mesma forma;
  - Atributos e associações definem uma classe.

Especificação de atributos

- Escolha um nome com significado;
- Siga um padrão de nomenclatura;
- Inclua-o na modelagem de classes:

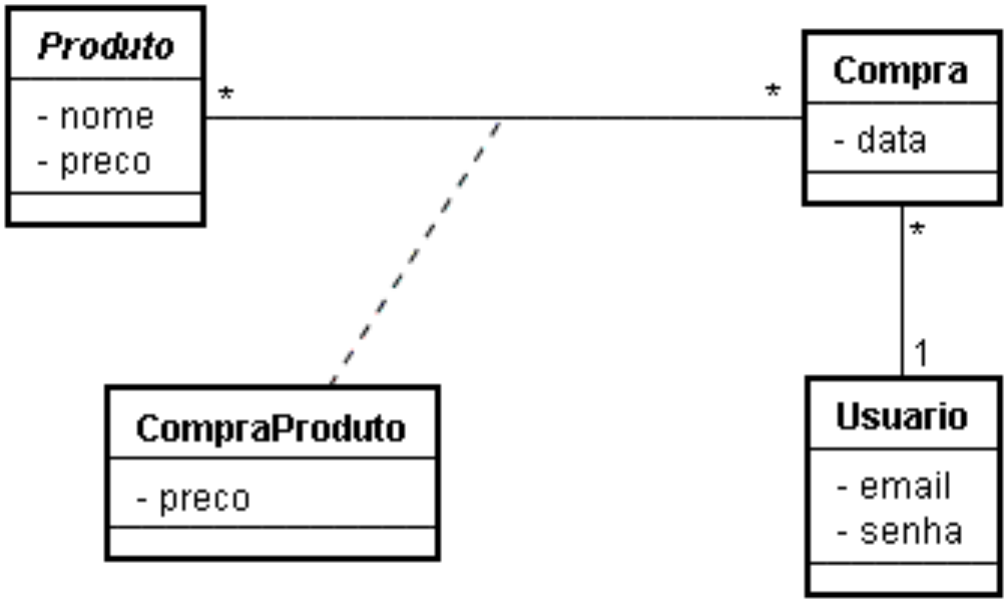


Atributos e hierarquias de classe

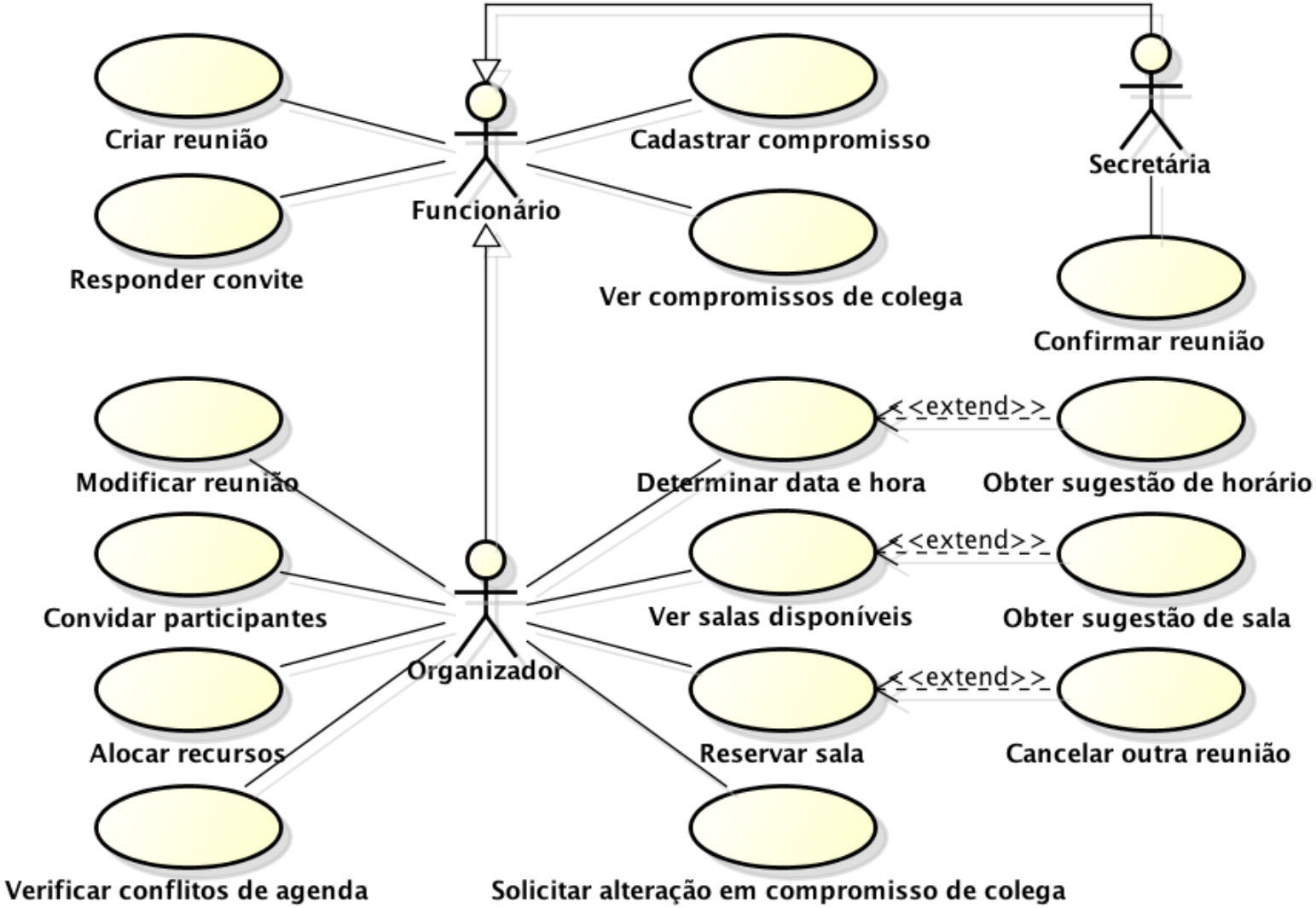
- Atenção à hierarquias de classes:
  - Atributos genéricos ficam mais acima na hierarquia;
  - Por outro lado, se ele não se aplica a algumas subclasses, deve ser trazido "para baixo", somente para as classes apropriadas.
- Revisão da hierarquia:
  - Descoberta de atributos nos leva a um melhor entendimento, o que possivelmente implicará revisão de hierarquias.

O modelo conceitual

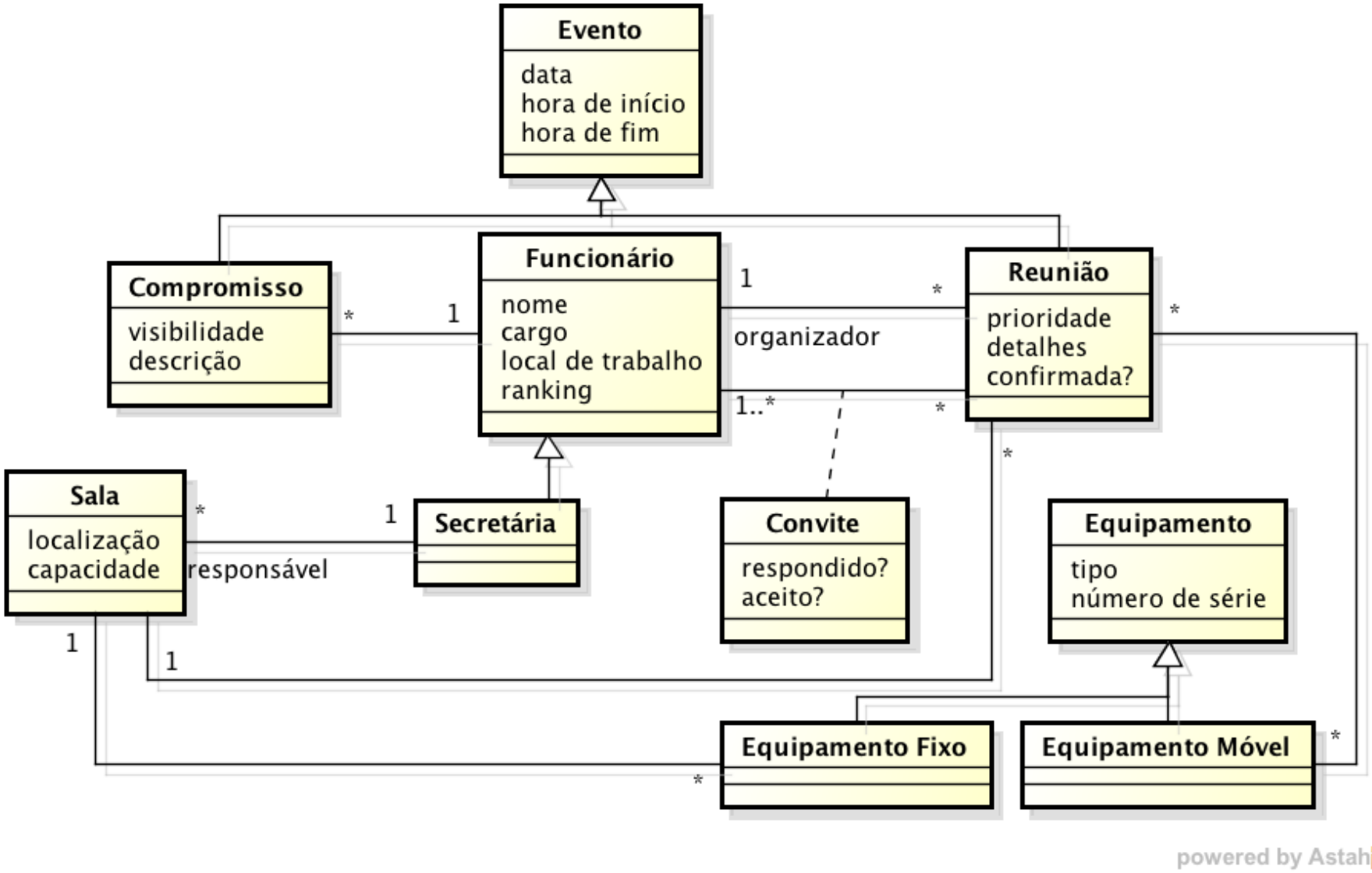
- Representa os elementos do mundo real;
- Desconsidera preocupações tecnológicas (ex.: tipos dos dados):

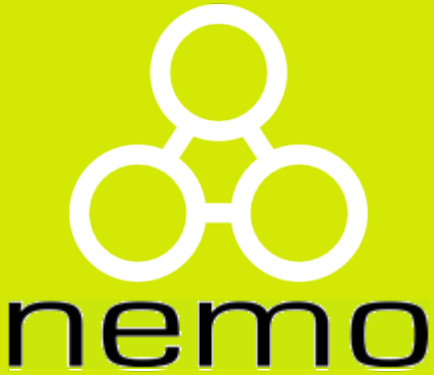


Dos casos de uso ao modelo conceitual



Dos casos de uso ao modelo conceitual





http://nemo.inf.ufes.br/