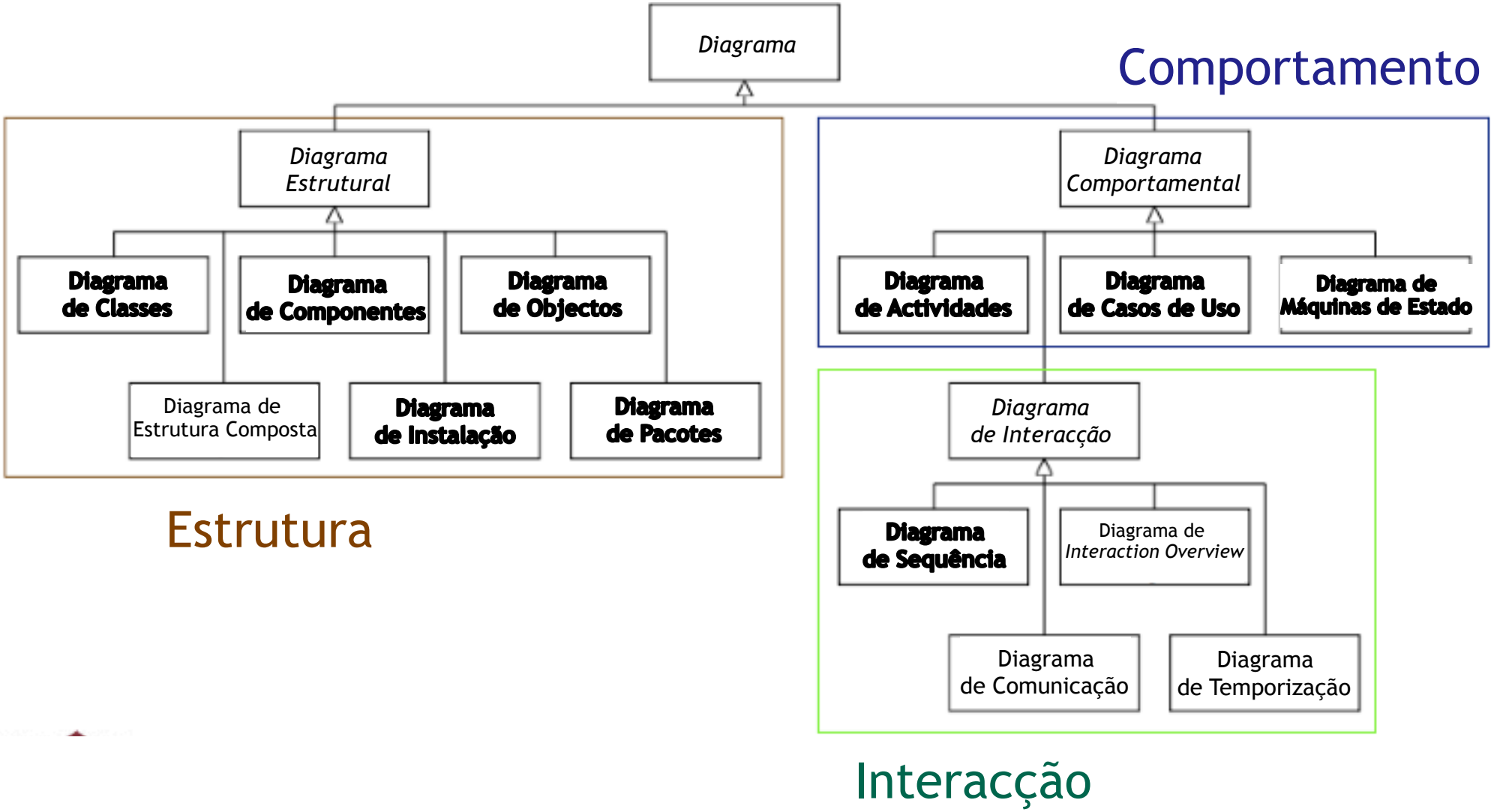
Desenvolvimento de Sistemas Software

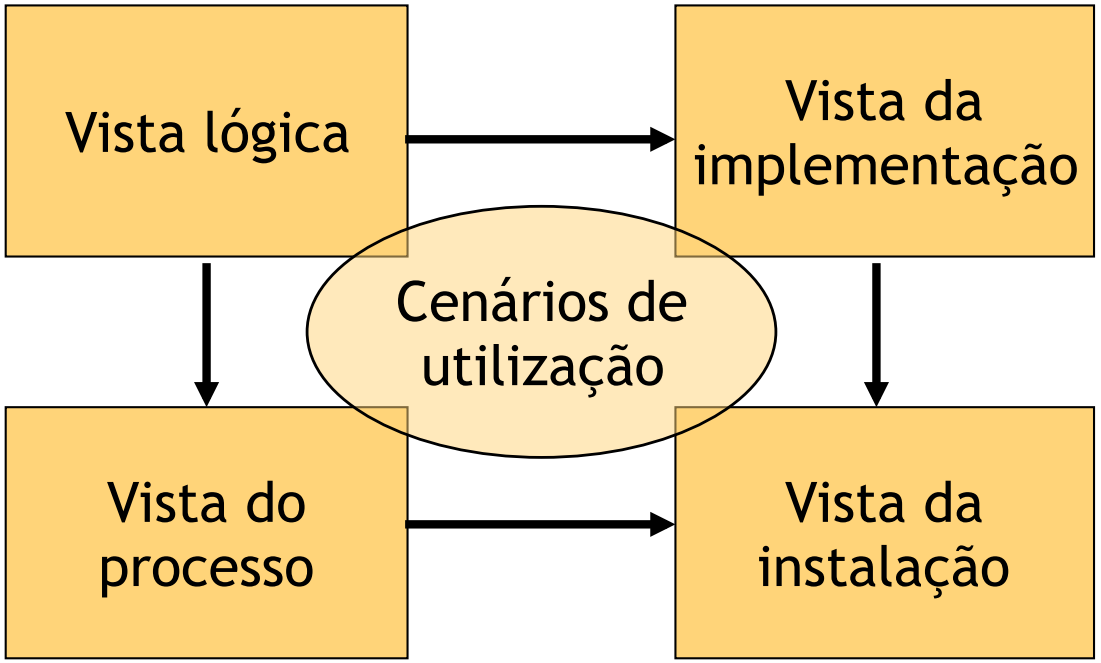
Aula Teórica 4: Modelação do Domínio

Desenvolvimento de Sistemas Software
António Nestor Ribeiro, José Creissac Campos, F. Mário Martins

Resumo das aulas anteriores...

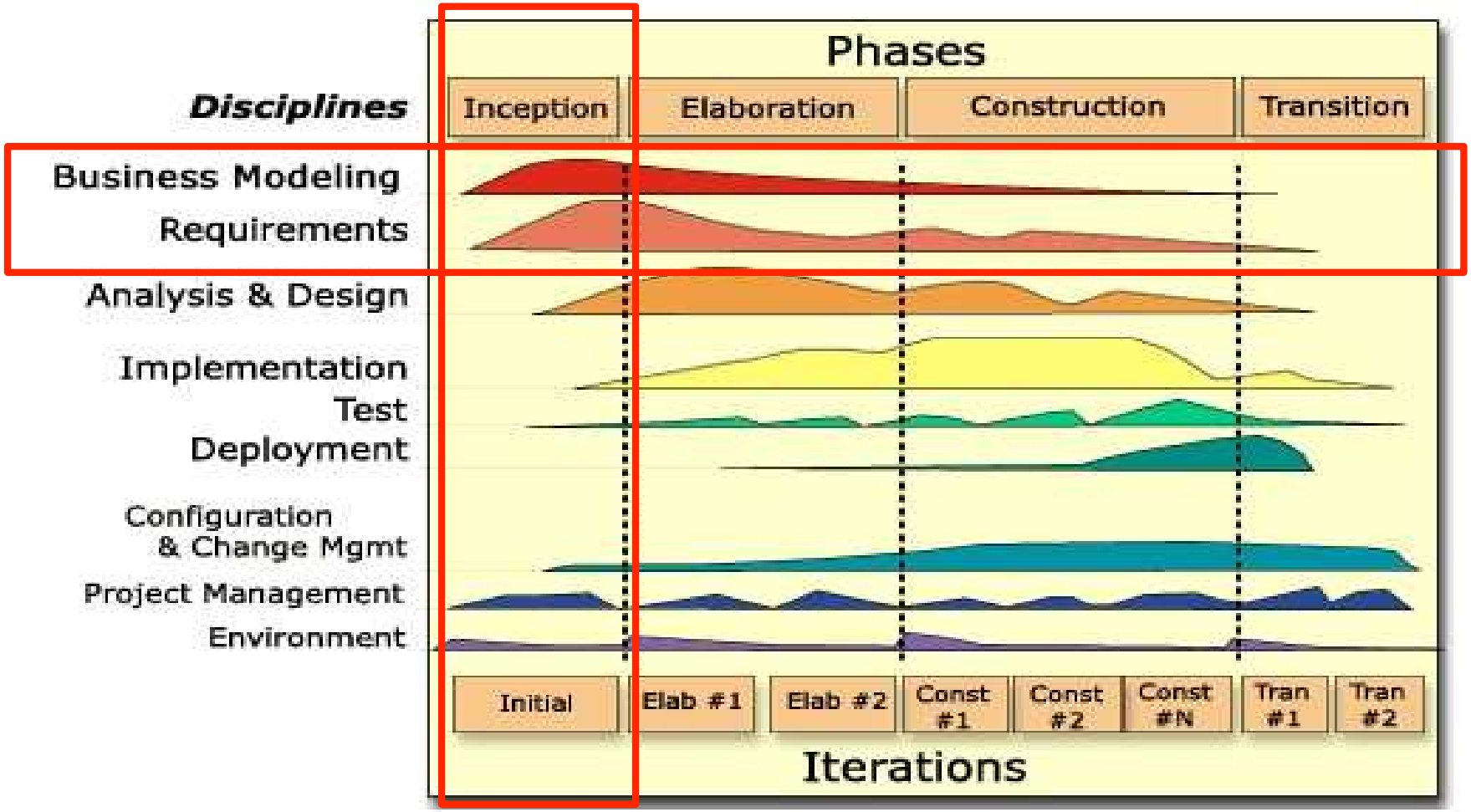


Resumo da aulas anteriores…

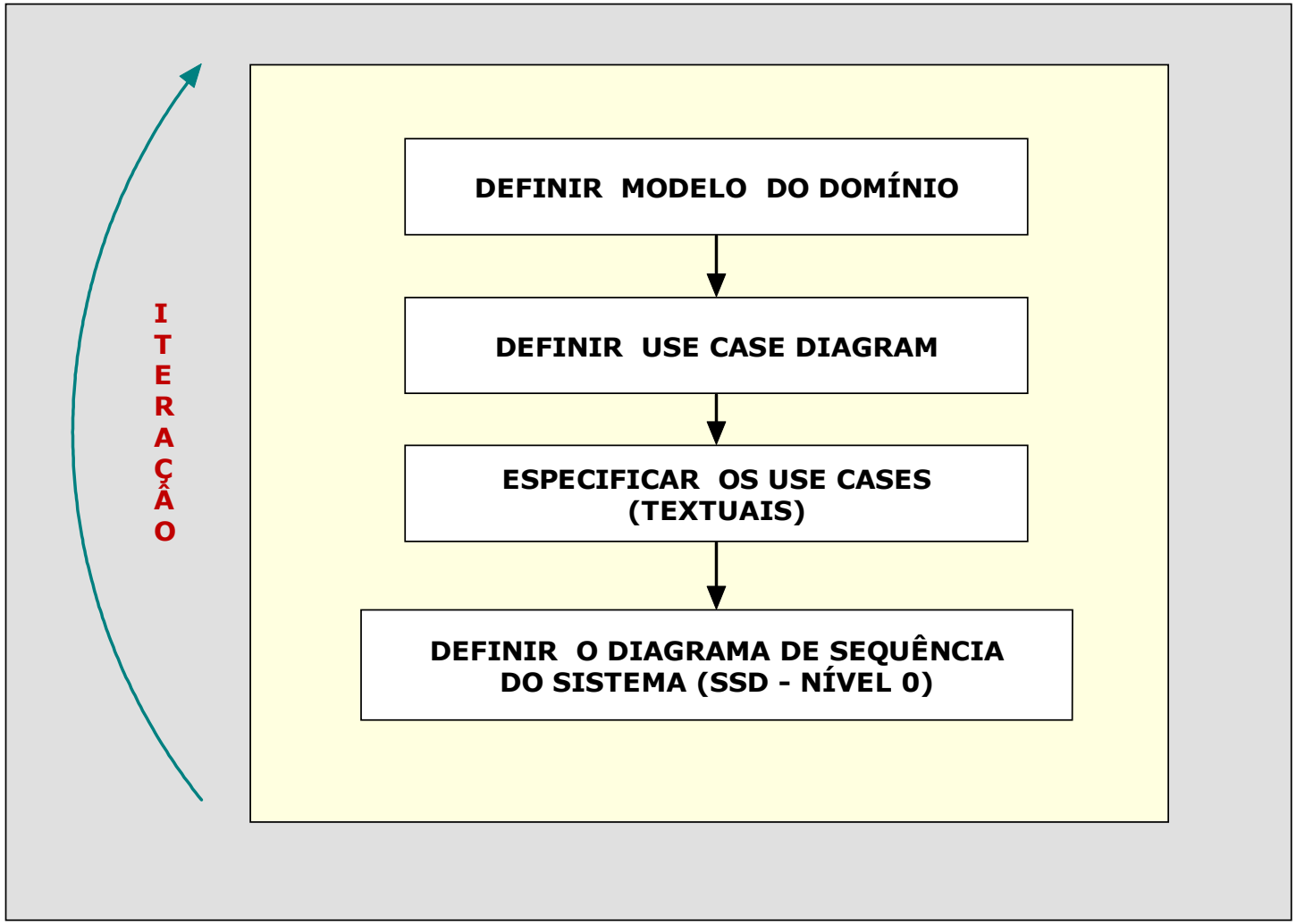


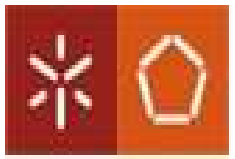
(Kruchten, 1995)

Resumo das aulas anteriores…



Requisitos funcionais





Modelos do Domínio

O que é um Domínio ?

O termo **Domínio** denota, em Engenharia Informática, mas não só, um conjunto de sistemas ou áreas funcionais, dos vários sectores organizacionais de actividade, ou da sociedade em geral, onde a existência de uma terminologia (**sintaxe ou termo**) deve estar inequivocamente associada a certos conceitos, o que em muito simplifica a compreensão e, em consequência, a correcta comunicação entre quem tem que "definir contractos de informatização" e quem tem que "realizar tais contractos" sno âmbito (domínio) de tais sistemas, áreas, etc.

Alguns Domínios em que a Engª Informática tem produzido "produtos":

Quase todos ou todos. Quem indica excepções ?

Exemplos de Domínios típicos do âmbito da Engª Informática:

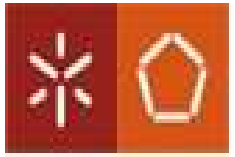
Bancário; Aviónica; Construção Civil; Administração Pública; Telecomunicações; Medicina e Clínica Médica; Biotecnologia; Segurança, enfim, todas as outras Engenharias e, actualmente, todas as outras áreas, cf. Arqueologia, Museus, Documentação, Administrativo, Gestão, etc.

Modelos do Domínio

- **Como modelar um Domínio numa perspectiva OO ?**

1) Organizando e definindo todo o vocabulário "fundamental" do domínio do problema, em especial o que sobressai da análise de requisitos com os interlocutores numa tabela semântica, ou seja, um dicionário terminológico aceite e assinado por ambas as partes;

2) Organizando e relacionando termos que estão definidos num glossário ou num dicionário de dados firmado, se a complexidade do problema o justificar, definindo as classes que representam as entidades do estado interno persistente e partilhado do sistema, como sendo as entidades (e eventos) do negócio, e suas respectivas ligações semânticas a outras entidades, bem como as classes que modelam a estrutura de documentos (que são dados estruturados) trocados entre o sistema e o seu ambiente;



Modelos do Domínio

- Objectos no modelo de domínio podem ser objectos físicos ou conceitos abstractos que representam actores, sistemas, objectos físicos (de dados, sistemas, organizações);
- O modelo de domínio é estático, ou seja, não representa tempo, não representa operações, eventos nem fluxos de dados;
- O modelo de domínio não inclui nem representa software;
- As entidades do modelo do domínio (representados como classes) são apenas candidatos a serem classes de programação;
- Os objectos no modelo do domínio podem ter atributos mas estes devem ser simples (números, strings, etc. mas não objectos);
- O modelo do domínio deve representar os relacionamentos fundamentais entre os objectos;
- Modelo do domínio => Objectos, atributos, associações (relacionamentos), multiplicidade, direcção de navegação e papel de um objecto num relacionamento

No **Mapa de Voos** do dia do AEROCEST, cada voo é identificado por um código de voo, tem uma entidade responsável, um conjunto de passageiros afectos a tal voo, caso seja um voo comercial poderá ter ou não uma eventual lista de espera de passageiros substitutos, um conjunto de cargas a embarcar (definida numa **lista de carga de produtos**), um destino, e um tempo de partida (hora/minuto). Uma **aeronave** específica capaz de realizar tal voo e uma **tripulação**, ser-lhe-ão posteriormente associadas também.

Um voo comercial é o mais usual e mais bem conhecido. Um voo militar deverá ter a si associada a seguinte informação adicional: tempo de voo, ramo das forças armadas e código de missão para comunicação (exº DELTA77).

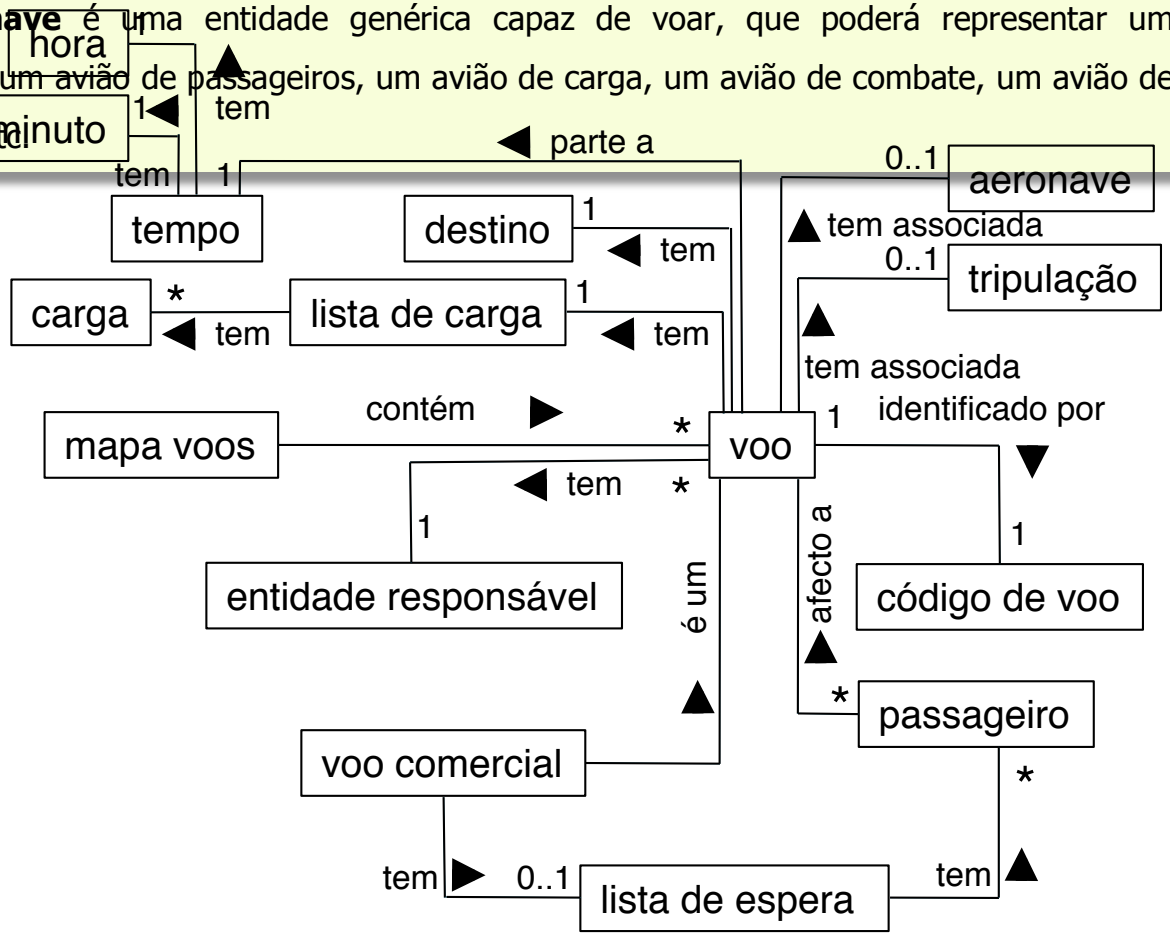
Uma **Aeronave** é uma entidade genérica capaz de voar, que poderá representar um helicóptero, um avião de passageiros, um avião de carga, um avião de combate, um avião de incêndios, etc.

mapa voos
voo
código de voo
entidade responsável
passageiro
voo comercial
lista de espera
carga
lista de carga
destino
tempo de partida
hora
minuto
aeronave
tripulação
voo militar
tempo de voo
ramo das forças armadas
código da missão
helicóptero
avião de passageiros
avião de carga
avião de combate
avião de incêndios

No **Mapa de Voos** do dia do AEROGEST, cada voo é identificado por um código de voo, tem uma entidade responsável, um conjunto de passageiros afectos a tal voo, caso seja um voo comercial poderá ter ou não uma eventual lista de espera de passageiros substitutos, um conjunto de cargas a embarcar (definida numa **lista de carga de produtos**), um destino, e um tempo de partida (hora/minuto). Uma **aeronave** específica capaz de realizar tal voo e uma **tripulação**, ser-lhe-ão posteriormente associadas também.

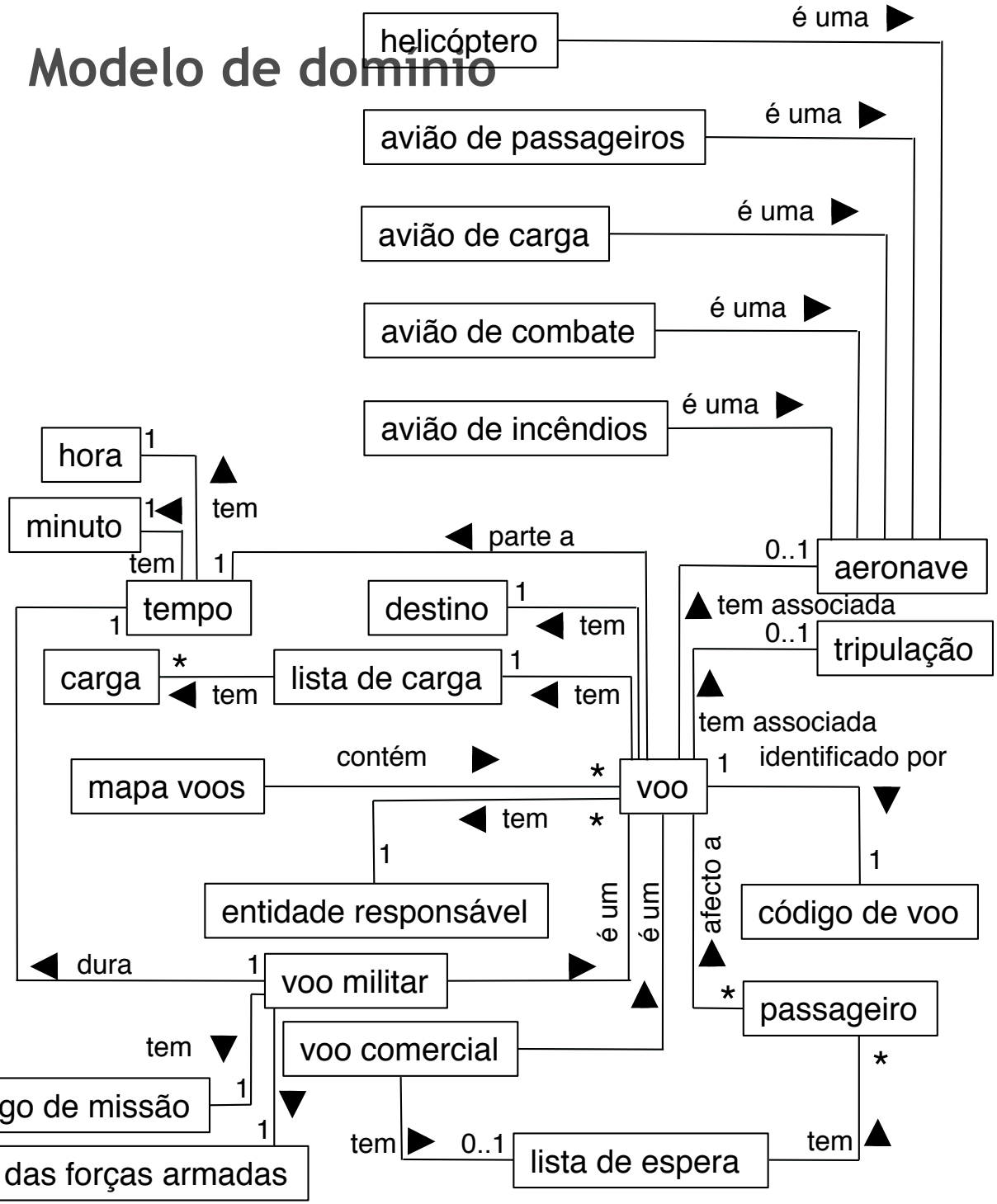
Um voo comercial é o mais usual e mais bem conhecido. Um voo militar deverá ter a si associada a seguinte informação adicional: tempo de voo, ramo das forças armadas e código de missão para comunicação (exº DELTA77).

Uma **Aeronave** é uma entidade genérica capaz de voar, que poderá representar um helicóptero, um avião de passageiros, um avião de carga, um avião de combate, um avião de incêndios, etc.



mapa voos
voo
código de voo
entidade responsável
passageiro
voo comercial
lista de espera
carga
lista de carga
destino
tempo de partida
hora
minuto
aeronave
tripulação
voo militar
tempo de voo
ramo das forças armadas
código da missão
helicóptero
avião de passageiros
avião de carga
avião de combate
avião de incêndios

Modelo de domínio



mapa voos
voo
código de voo
entidade responsável
passageiro
voo comercial
lista de espera
carga
lista de carga
destino
tempo de partida
hora
minuto
aeronave
tripulação
voo militar
tempo de voo
ramo das forças armadas
código da missão
helicóptero
avião de passageiros
avião de carga
avião de combate
avião de incêndios

Algumas notas sobre entidades

- Entidades no modelo de domínio correspondem a “substantivos” na descrição

- Algumas regras para ponderar rejeição de entidades (a partir dos substantivos)
  - É sinónimo de outra entidade?
    - tempo de partida / tempo de voo?
  - Está fora do âmbito da análise?
    - AEROGEST…
  - Refere-se a relações entre outras entidades?
    - “voo comercial poderá ter tem um conjunto de passageiros”
  - É fruto do estilo de escrita?
    - “Um voo militar deverá ter a si associada a seguinte informação adicional:”

Algumas notas sobre relações

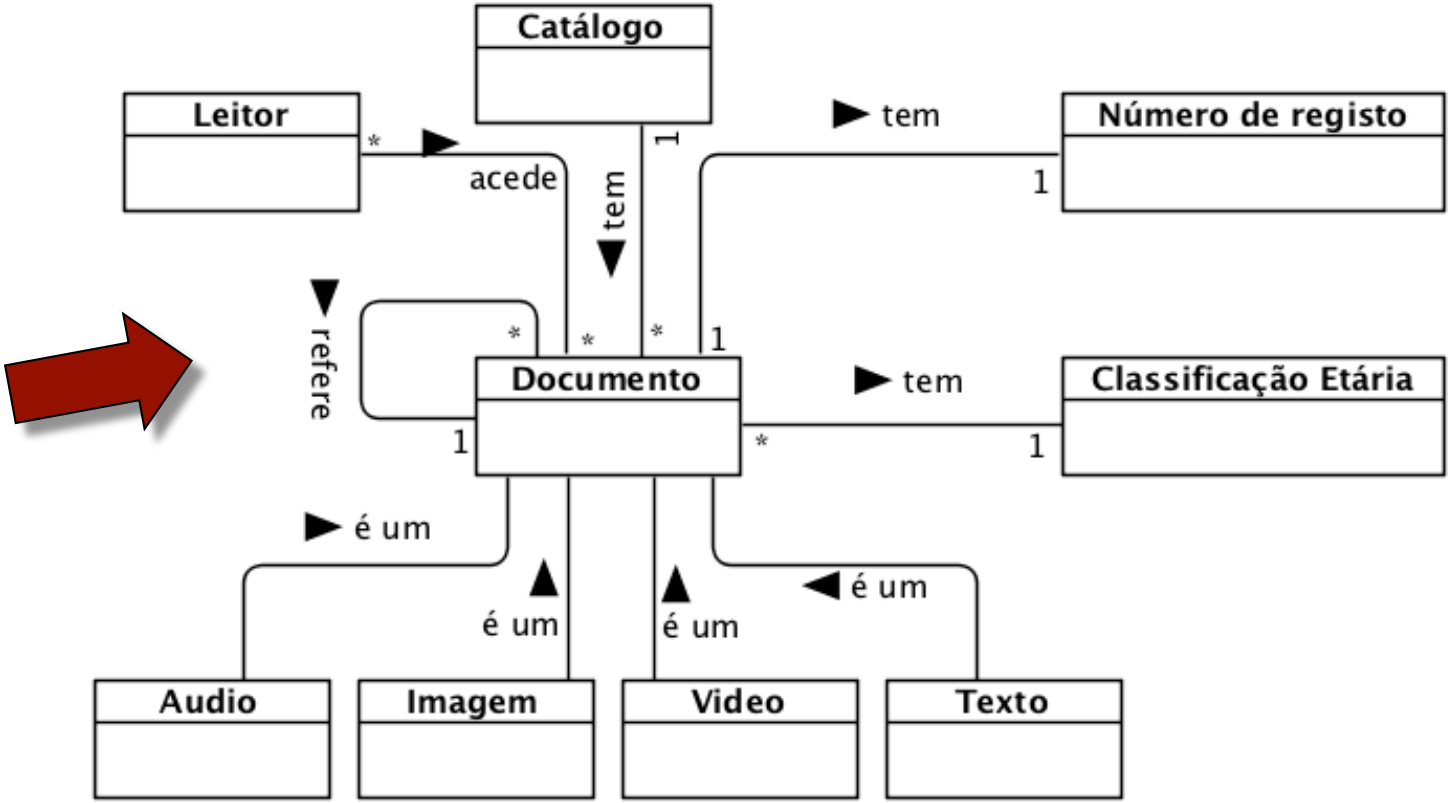
- Relações no modelo de domínio correspondem a “verbos” na descrição

- Relação “é um(a)”
  - Explicita relação de tipagem
  - Representação posterior em OO
    - Herança – classe / sub-classe
    - Realização – classe / interface
    - Atributo na classe!

Modelo de domínio

- Modelos de domínio são representados em **diagramas de classe** da UML
- Utilizam apenas um sub-conjunto da notação

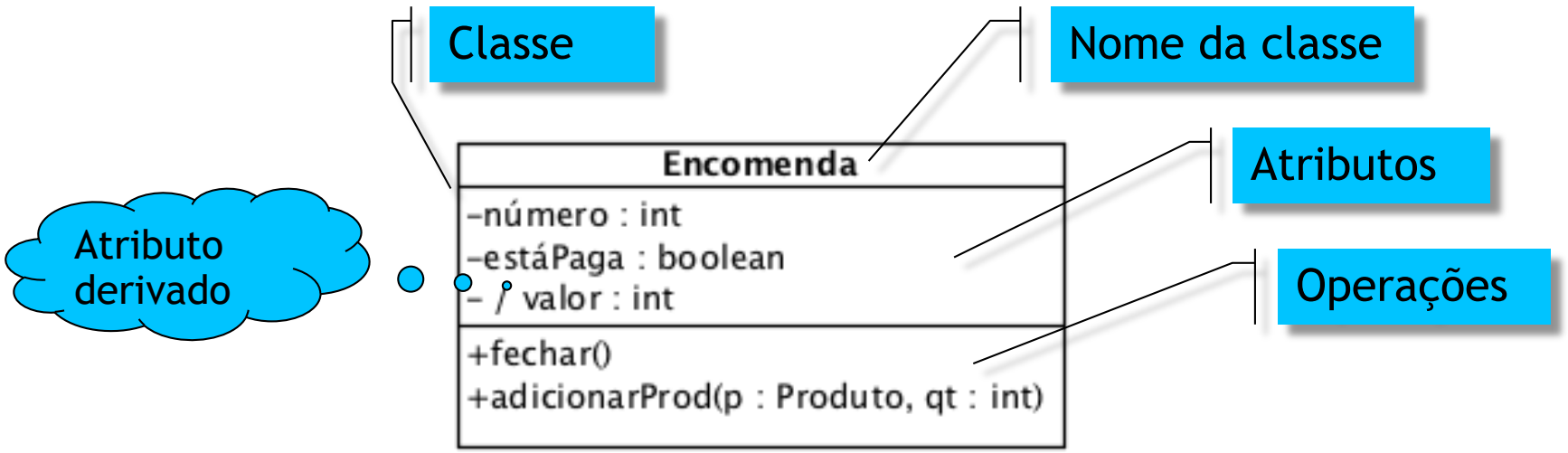
Diagrama de Classe UML



Diagramas de Classe I - conceitos base

- A noção de classe é fundamental no paradigma OO
  - tipicamente uma classe representa uma **abstração** de uma entidade do mundo real.
- Cada classe descreve um conjunto de objectos com a mesma estrutura e comportamento:
  - Estrutura:
    - atributos
    - relações
  - Comportamento:
    - operações
- A organização do código em classes tem dois objectivos fundamentais:
  - facilitar a reutilização — através da reutilização de classes previamente desenvolvidas em novos sistemas;
  - facilitar a manutenção — o sistema deverá ser desenvolvido de forma a que a alteração de uma classe tenha o menor impacto possível no resto do sistema.

Representação de classes em UML



- Compartimentos pré-definidos
  - Nome da classe — começa com maiúsculas / substantivo
  - Atributos (de instância) — representam propriedades das instâncias desta classe / começam com minúsculas / substantivos
  - Operações (de instância) — representam serviços que podem ser pedidos a instâncias da classe / começam com minúsculas / verbos
- Compartimentos podem ser omitidos — isso não significa que não exista lá informação!

Visibilidade de atributos e operações

O nível de visibilidade (acesso) que se pretende para cada atributo/operação é representado com as seguintes anotações:

- privado — só acessível ao objecto a que pertence (cf. encapsulamento)

\# protegido — acessível a instâncias das sub-classes (atenção: em Java fica também acessível a instâncias de classes do mesmo *package*!)

pacote/*package* — acessível a instâncias de classes do mesmo *package* (nível de acesso por omissão)

\+ público — acessível a todos os objectos no sistema (que *conhecam* o objecto a que o atributo/operação pertence!)

Relações entre classes

- Três tipos de relações possíveis entre as classes:
  - Generalização/Especialização

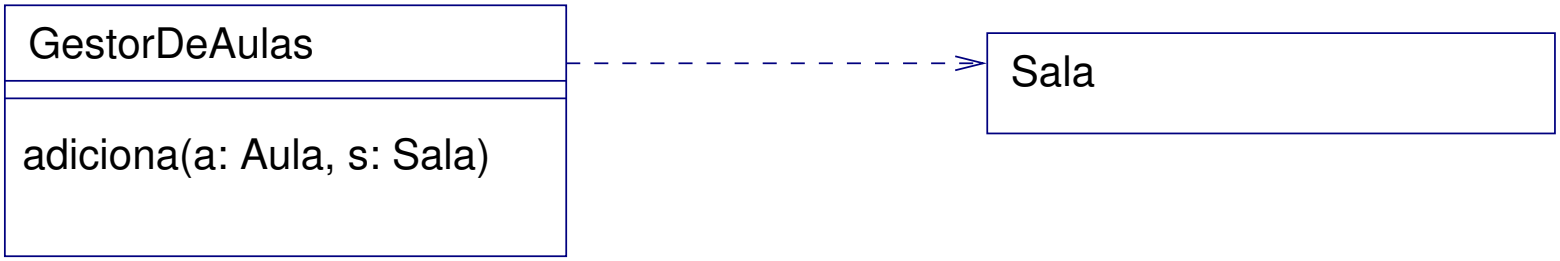
    relação entre classe mais geral e classe mais específica
  - Dependência

    indica que uma classe depende de outra
  - Associação

    indica que existe algum tipo de ligação entre objectos das duas classes

Dependência

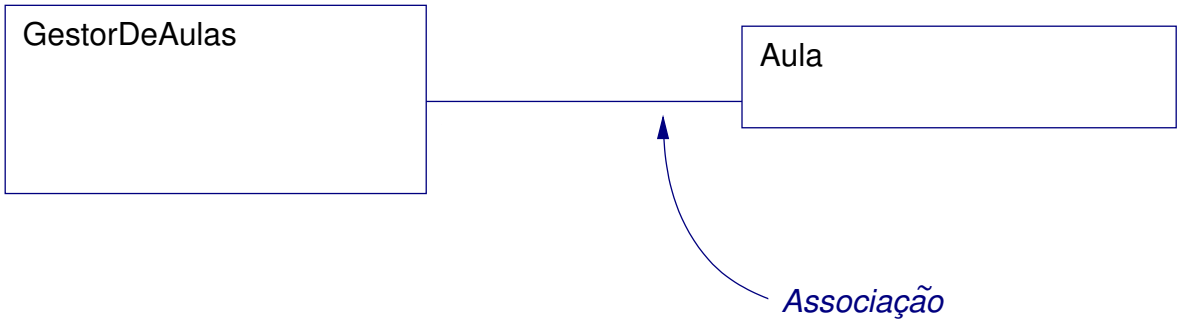
- Notação:



- Indica que a definição de uma classe está dependente da definição de outra.
- Utiliza-se normalmente para mostrar que instâncias de origem utilizam, de alguma forma, instâncias de destino (por exemplo: um parâmetro de um método)
- Uma alteração no destino (quem é usado) pode alterar a origem (quem usa)

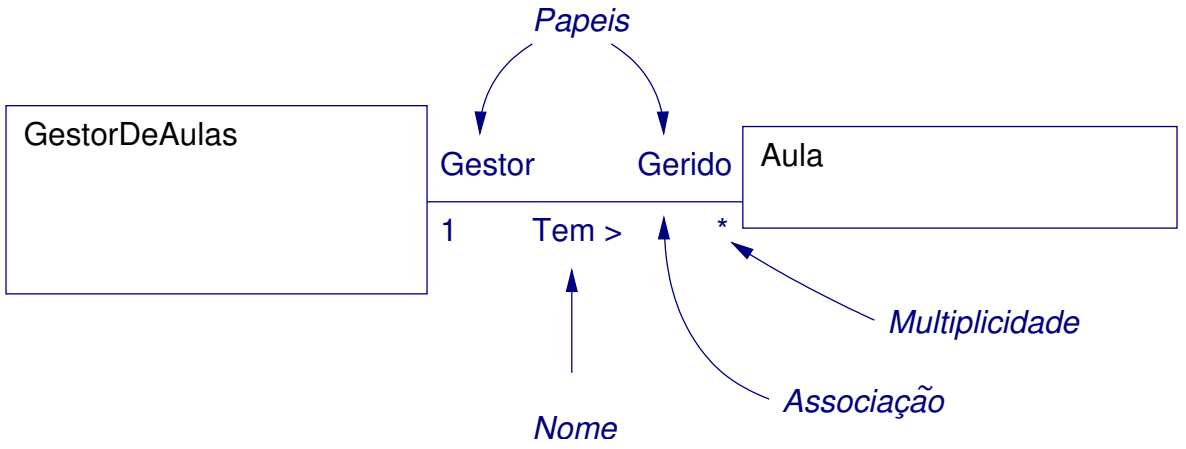
Associação

- Notação:



- Indica que objectos de uma estão ligados a objectos de outra — define uma relação entre os objectos
- Noção de nevagabilidade (cf. diagramas E-R)
- Por omissão representam navegação bidireccional — mas pode indicar-se o sentido da navegabilidade utilizando setas nos extremos da associação.

- Três decorações possíveis:
  - nome — descreve a natureza da relação (pode ter direcção)
  - papeis — indica o papel que cada classe desempenha na relação definida pela associação (usualmente utilizado como alternativa ao nome)
  - multiplicidade — quantos objectos participam na relação:
    - $*$ — zero ou mais objectos
    - $1..*$ — um ou mais objectos
    - n — n objectos
    - 1 — um objecto / objecto obrigatório
    - $0..1$ — zero ou um objectos / objecto opcional

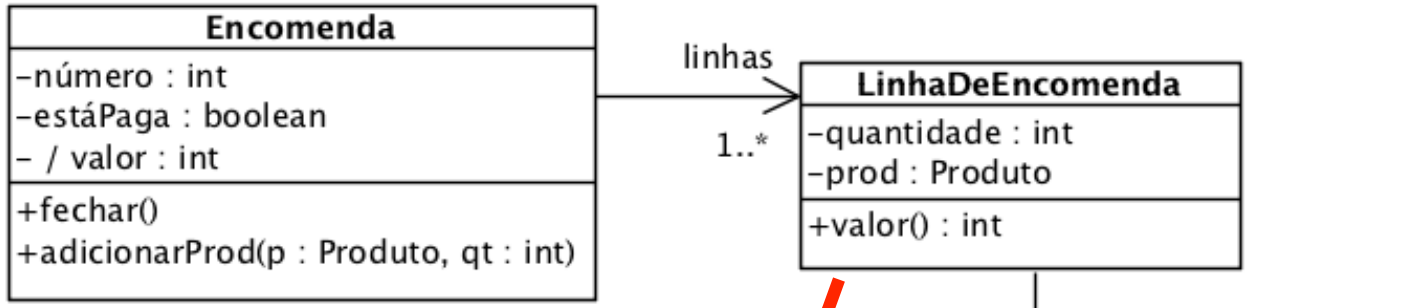


Associações vs. Atributos

- Atributos (de instância) representam propriedades das instânias das classes
  - são codificados como variáveis de instância
- Associações também representam propriedades das instâncias das classes
  - também são codificados como variáveis de instância
- Atributos devem ter tipos simples
  - utilizar associações para tipos estruturados

Erro de Principiante!

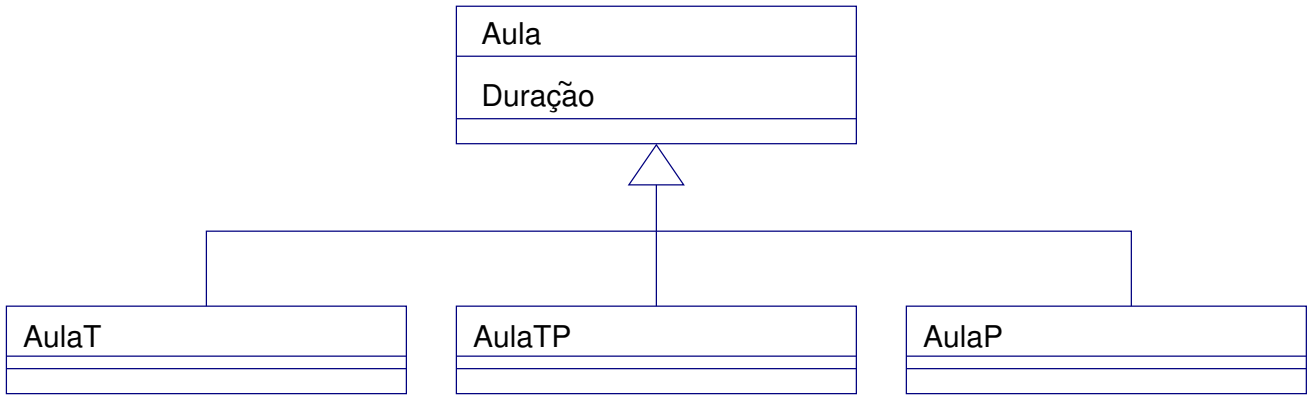
Repetir as associações nos atributos.



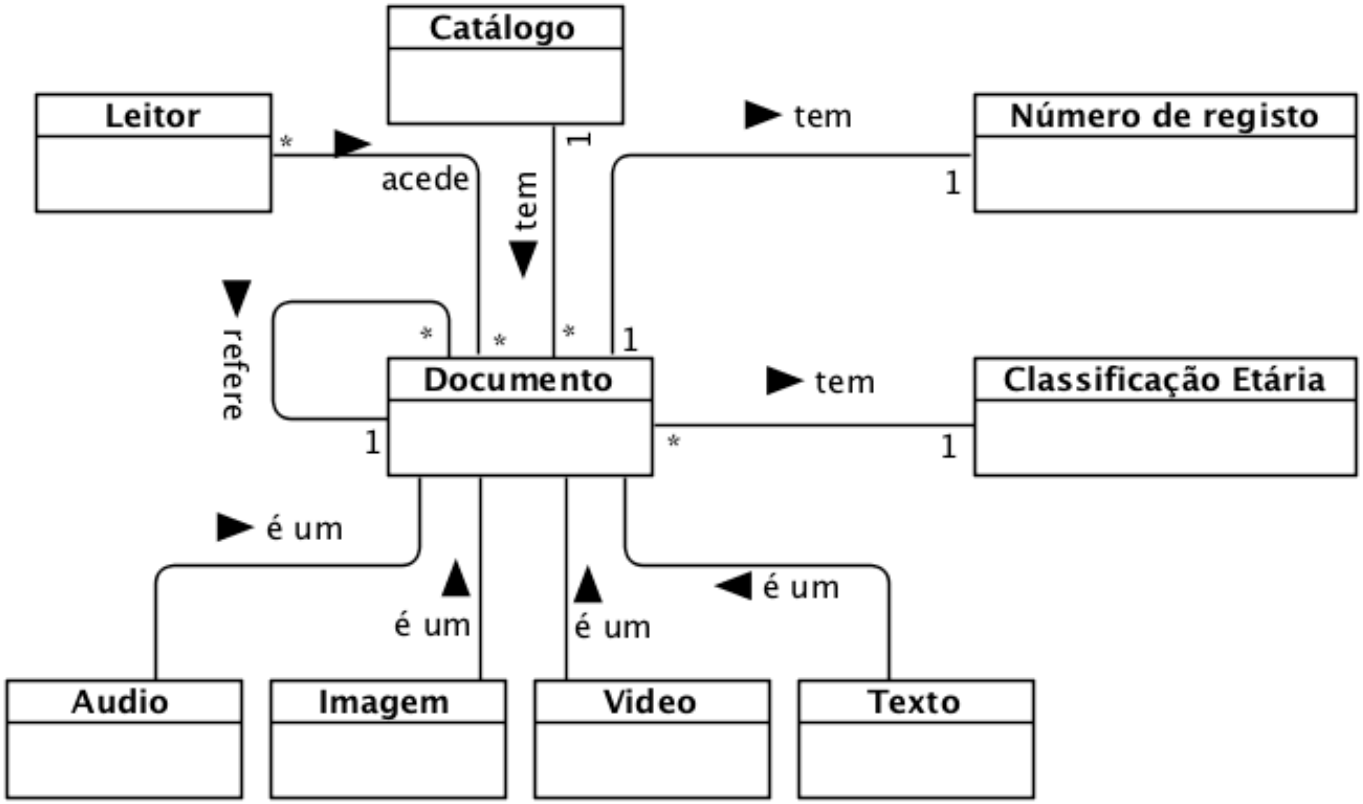

```
class LinhaDeEncomenda {
        private int quantidade;
        private Produto prod;
        private Produto produtoEcomendado;

        public play() {...}
}
```

Generalização/Especialização

- Indica a relação entre uma classe mais geral (super-classe) e uma classe mais específica (sub-classe).
- Noção de *is-a* — tipagem / substitubilidade
- Polimorfismo — duas sub-classes podem fornecer métodos diferentes para implementar uma operação da super classe.
- *Overriding* — sub-classe pode alterar o método associado a uma operação declarada pela super-classe
- Herança simples vs. herança multipla
- Notação:



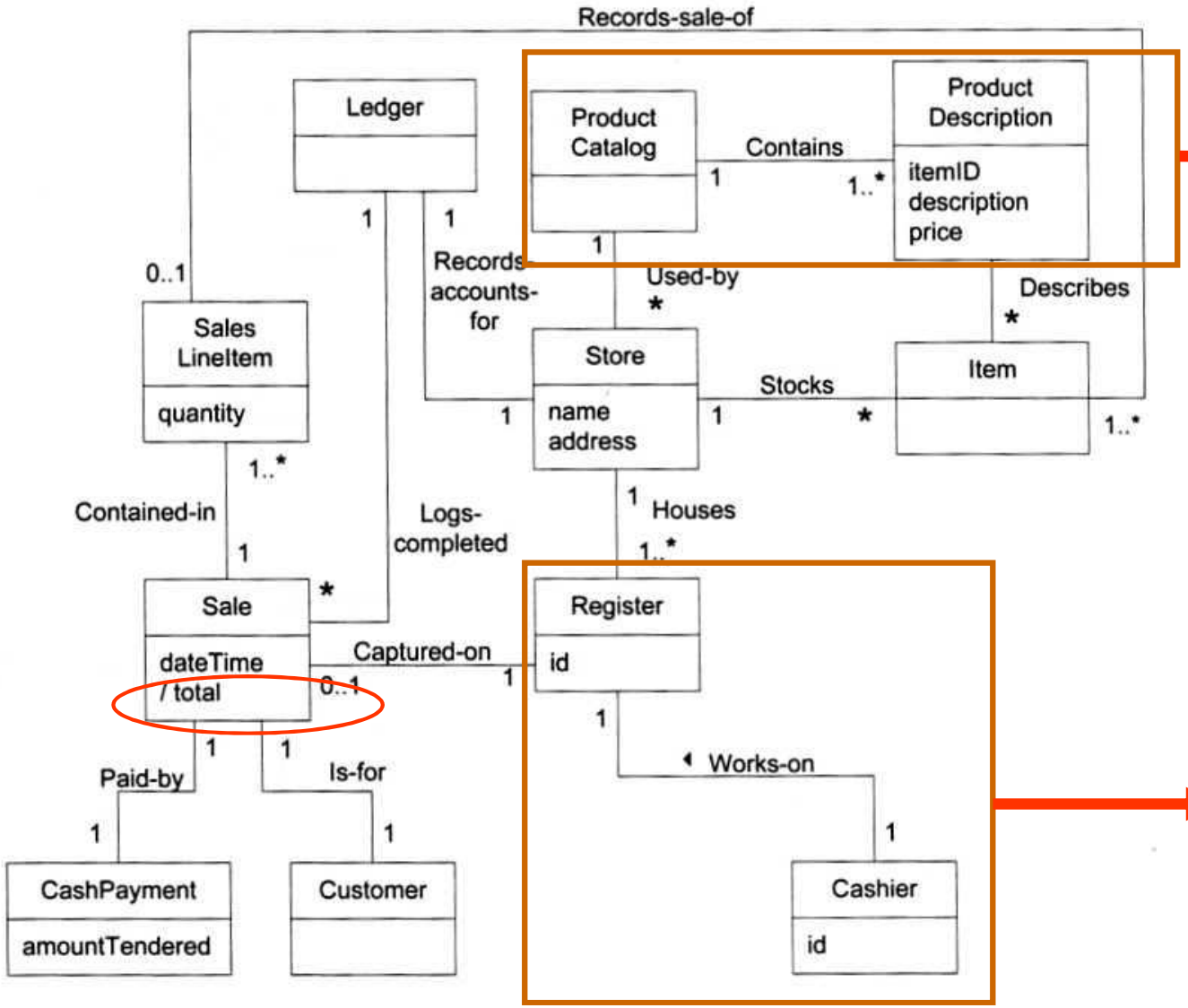
Exemplos



Um catálogo (con)tem documentos que são acedidos por leitores. Um documento pode ser audio, imagem, vídeo, ou texto e tem sempre em número de registo e uma classificação etária. Cada documento refere sempre um outro documento(!).

Universidade do Minho
Departamento de Informática

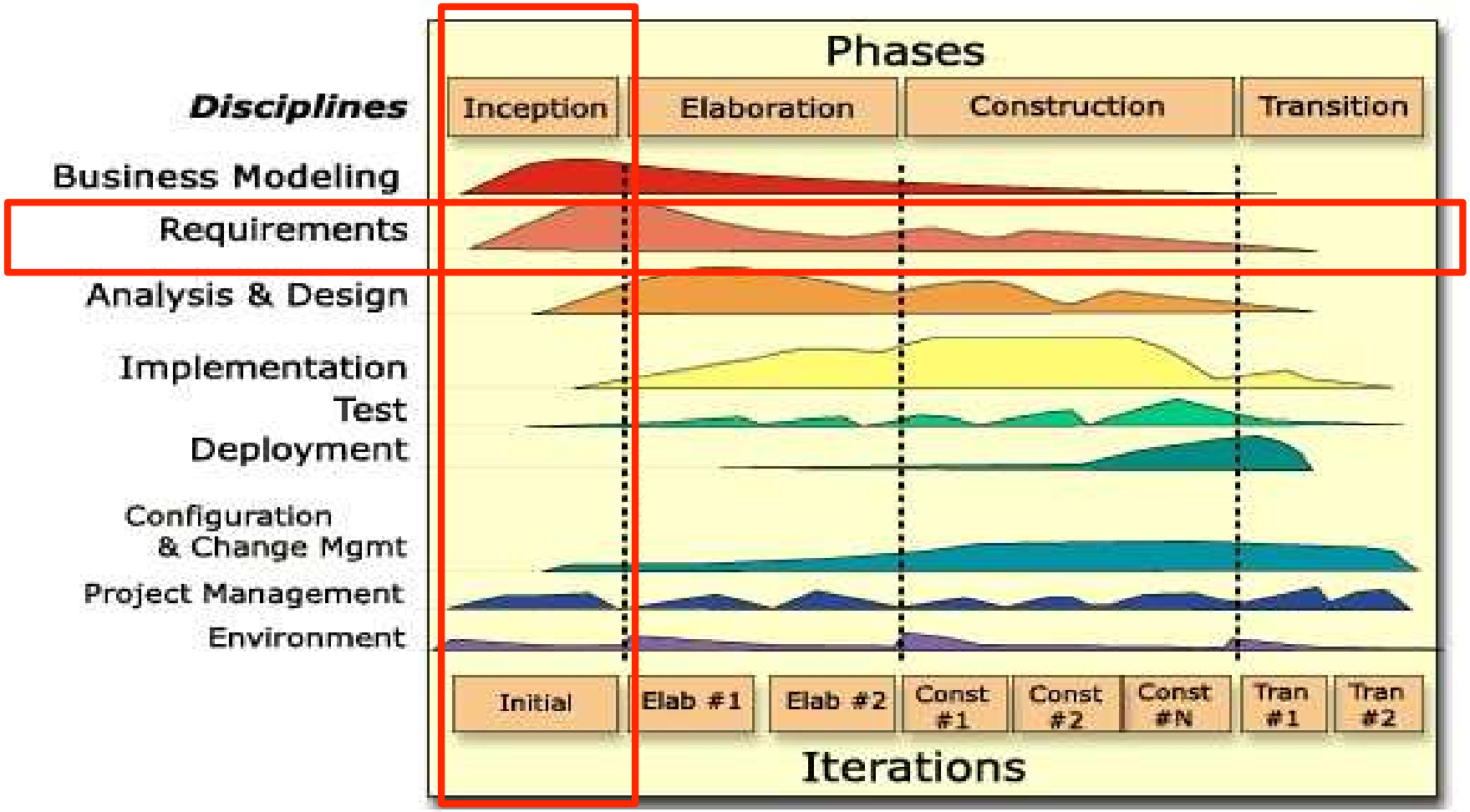
Modelos do Domínio: POS



O Catálogo de produtos contém descrições de produtos, não podendo estar vazio. Cada descrição de produto inclui o seu identificador, uma descrição e o preço.

"Cada funcionário de caixa trabalha, a cada momento, numa máquina com identifi-cação única".

Próximos passos…



Modelação do Domínio

Sumário:

- O que é um Modelo de Domínio
- Identificação de Entidades
- Identificação de Relações
- Representação de Modelos de Domínio em UML – Diagramas de Classe
- Notação básica dos Diagramas de Classe
  - Classes
  - Dependências
  - Associações simples
  - Generalizações