

Projeto Detalhado de Software

Projeto Orientado a Objetos

Princípios de projeto (separação de interesses, encapsulamento de informações, coesão e acoplamento)

Profa. Jacilane de Holanda Rabelo

jacilane.rabelo@ufc.br

Projeto de Software

- Projetar Software é o **processo de aplicar várias técnicas e princípios** com o propósito de se **definir um dispositivo, processo ou sistema**, com detalhes suficientes para permitir sua realização física
- O Projeto de software é o **núcleo técnico da Engenharia de Software**. É a única maneira de se traduzir "com precisão", os requisitos do usuário para um produto ou sistema acabado



Projeto de Software

- Projetar Software é o **processo de aplicar várias técnicas e princípios** com o propósito de se **definir um dispositivo, processo ou sistema**, com detalhes suficientes para permitir sua realização física
- O Projeto de software é o **núcleo técnico da Engenharia de Software**. É a única maneira de se traduzir "com precisão", os requisitos do usuário para um produto ou sistema acabado

Meta:

Traduzir requisitos numa representação de software



Projeto de Software - Princípios

- Desenvolver um projeto de software é **um processo que combina**:
 - Um conjunto de **princípios** e/ou **heurísticas** que guiam o desenvolvimento do modelo
 - Um conjunto de **critérios** que facilitam a verificação da qualidade
 - Um processo de **iteração** que conduz a uma representação do projeto final

Projeto de Software

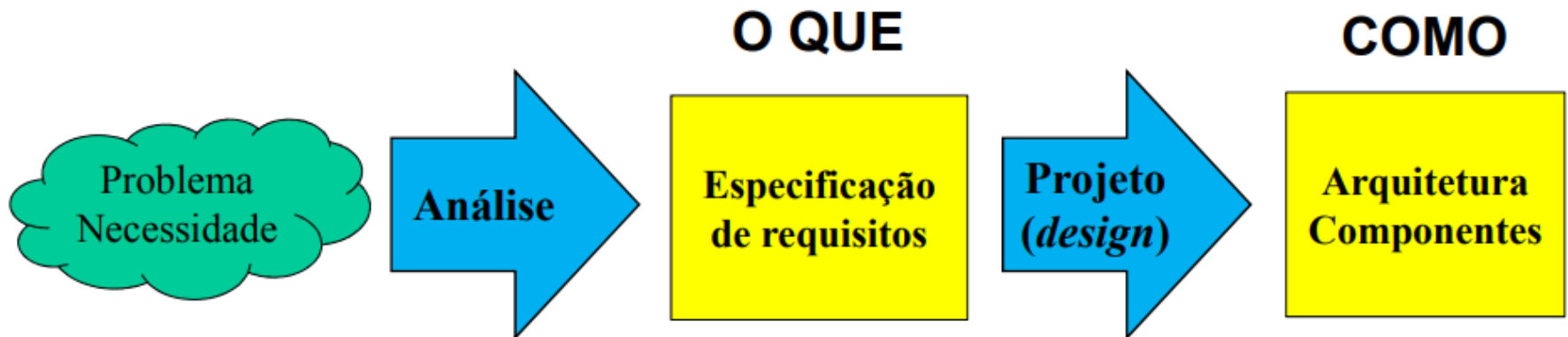
Perspectiva do Processo

- Atividade do ciclo de vida na qual os requisitos de software são analisados para **produzir uma descrição da estrutura interna do software** que servirá de base para a sua construção

Perspectiva do Resultado

- Descreve como um sistema é **decomposto organizado em componentes** e **descreve as interfaces entre esses componentes**
- Refina a descrição dos componentes até um nível de detalhamento que permita a **sua construção**
 - arquitetura do software
 - (componentes e interfaces entre componentes)

Análise vs Projeto



- **Análise:**

- Entendimento do problema
- Entendimento do escopo (âmbito) da solução
- Definição de uma solução conceitual (O QUE)

- **Projeto*:**

- Definição de uma solução lógica (COMO)

* “Projeto” em inglês pode ser ‘Project’ ou ‘Design’. ‘Project’ é uma iniciativa que envolve recursos para gerar um resultado. ‘Design’ é uma etapa do desenvolvimento de software entre a Análise e a Implementação; há autores que traduzem como ‘Desenho’.

Importância do Projeto de Software

- Fomentar a qualidade durante o processo de desenvolvimento
- Fornecer apresentações do software que podem ser avaliadas quanto à qualidade
- Traduzir com precisão os requisitos de um cliente num produto de software finalizado
- Detecção de problemas
 - Podem comprometer seu uso e até mesmo conclusão do mesmo
- Se forem detectados apenas na construção do software
 - Correções podem ser custosas e parte do trabalho pode ser perdida

FUNDAMENTOS

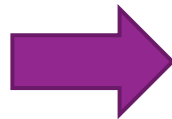
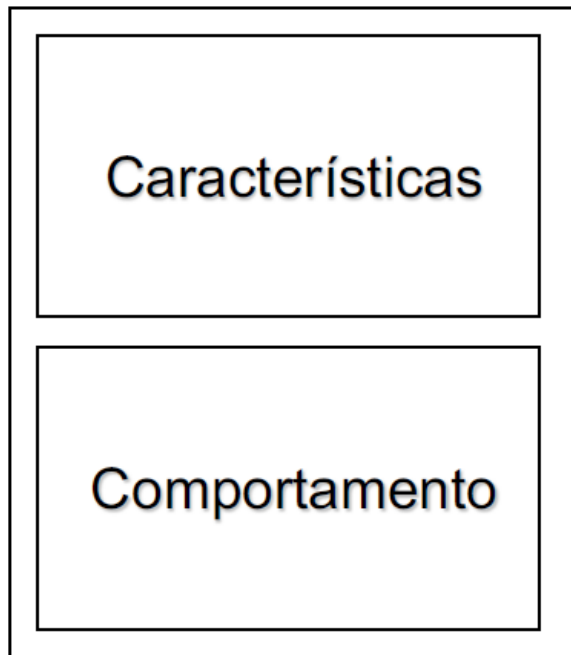
Objetos

- Podemos criar programa pensando em termos de **objetos ao invés de algoritmos**?
- O mundo é composto de objetos
 - Uma loja tem produtos, pedidos, estoque, etc.
 - Um restaurante tem mesas, garçons, comidas, bebidas, etc.
 - Uma universidade tem professores, alunos, disciplinas, etc.
 - Uma rodoviária tem ônibus, passageiros, bagagens, etc.
- E se **criarmos programas** basicamente **criando objetos** equivalentes ao mundo real, e fazendo com que esses **objetos se comuniquem**?



Objetos

Objeto no Mundo Real

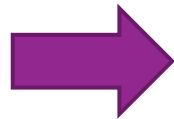
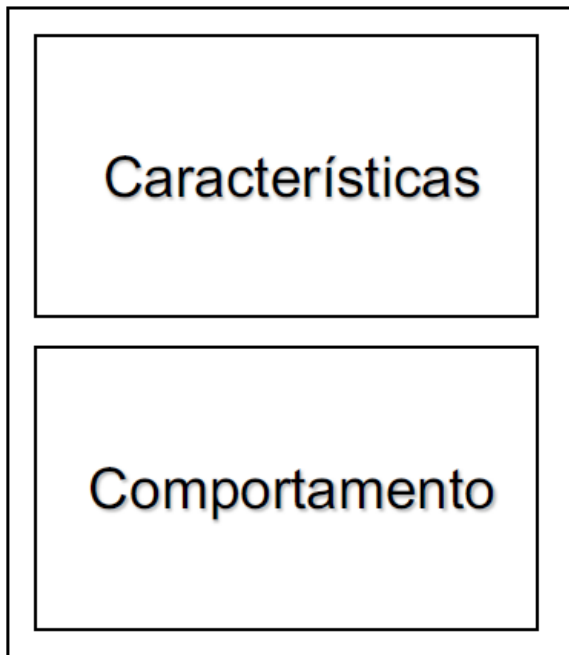


Exemplo de características do objeto **carro**

- Cor
- Marca
- Número de portas
- Ano de fabricação
- Tipo de combustível

Objetos

Objeto no Mundo Real

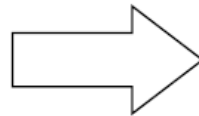
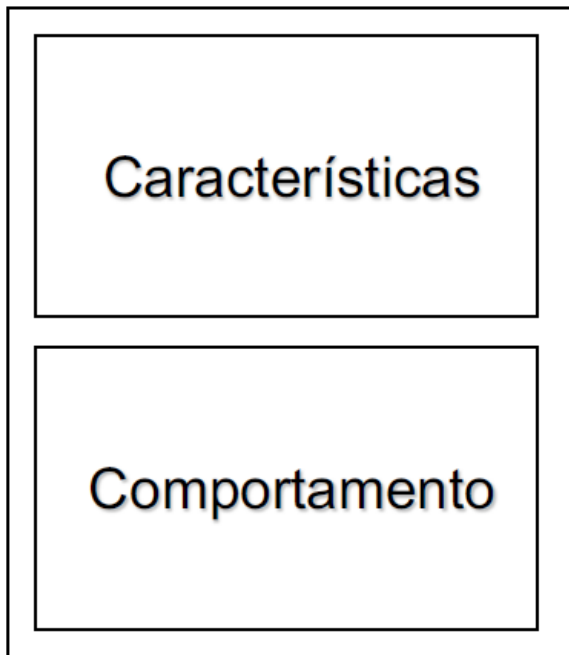


Exemplos de comportamento para o objeto **carro**

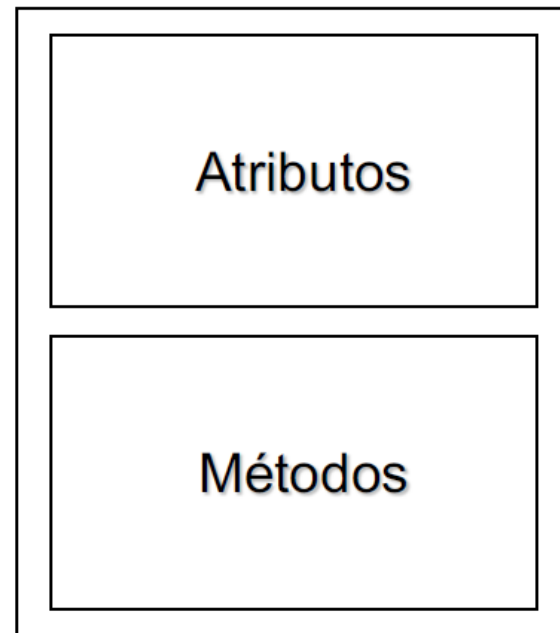
- Acelerar
- Frear
- Virar para direita
- Virar para esquerda

Objetos

Objeto no Mundo Real

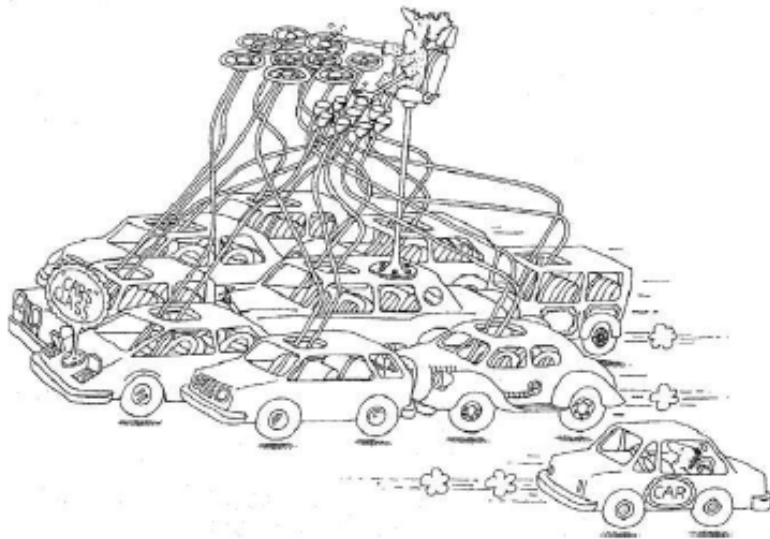


Objeto Computacional



Classes

- São especificações para objetos;
- Representam um conjunto de objetos que compartilham características e comportamentos comuns.



Todo carro tem em comum:

Característica

Cor

Pneu

Direção

Comportamento

Dirigir

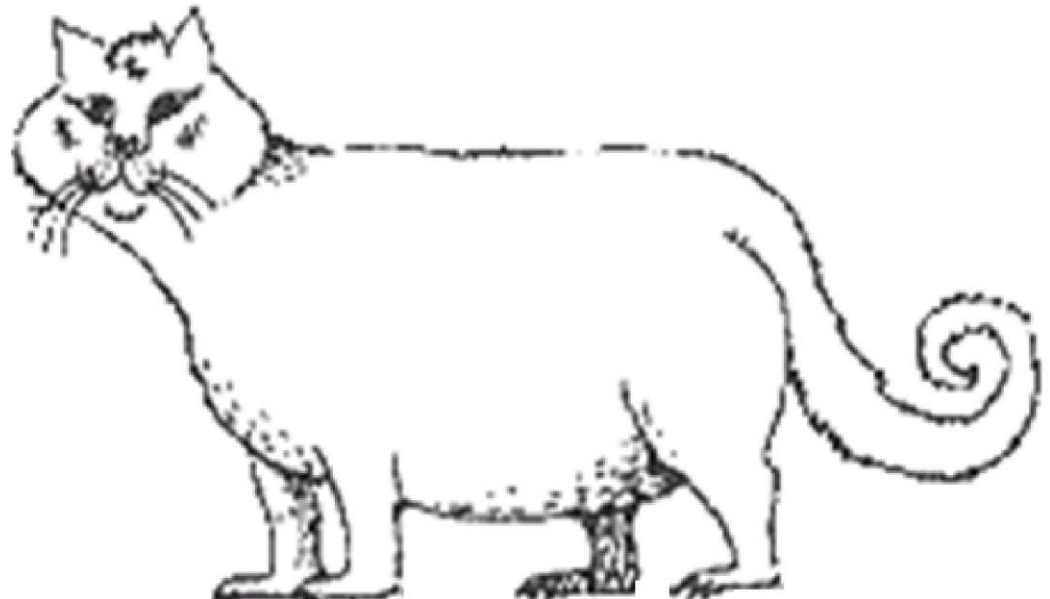
Frear

Abstração

- Abstração é o processo de **identificar as qualidades** ou **propriedades** importantes do problema que está sendo modelado
 - Através de um modelo abstrato, pode-se **concentrar nas características relevantes** e ignorar as irrelevantes
 - A abstração é fruto do raciocínio

Abstração

- Abstração é o processo de **identificar as qualidades** ou **propriedades** importantes do problema que está sendo modelado
 - Através de um modelo abstrato, pode-se **concentrar nas características relevantes** e ignorar as irrelevantes
 - A abstração é fruto do raciocínio
- O que você está vendo?



Abstração

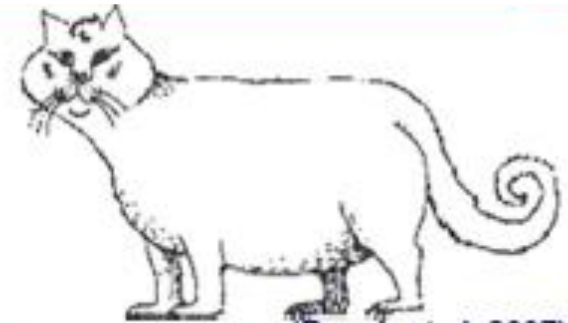
- Diferentes níveis de abstração

□ Animal

▣ Mamífero

■ Gato

■ Gato Persa

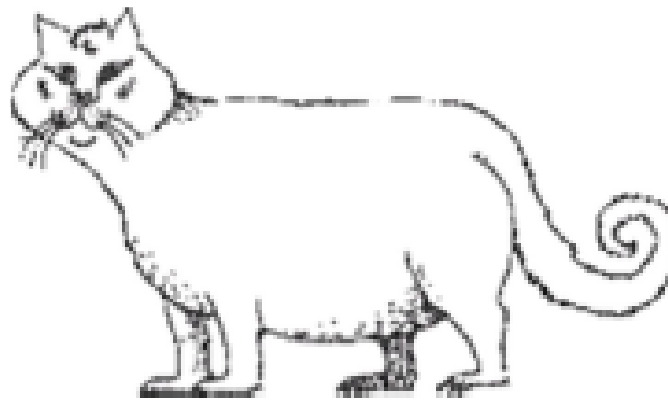


(Booch et al, 2007)

“A abstração está nos olhos de quem vê”

Abstração

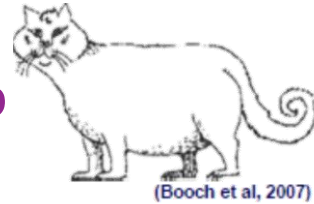
- Existem visões diferentes?
 - A seleção de quais aspectos são essenciais **depende do observador** e do **fenômeno observado**



(Booch et al, 2007)

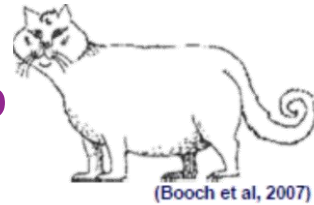
Abstração

- Existem visões diferentes?
 - A seleção de quais aspectos são essenciais **depende do observador** e do **fenômeno observado**

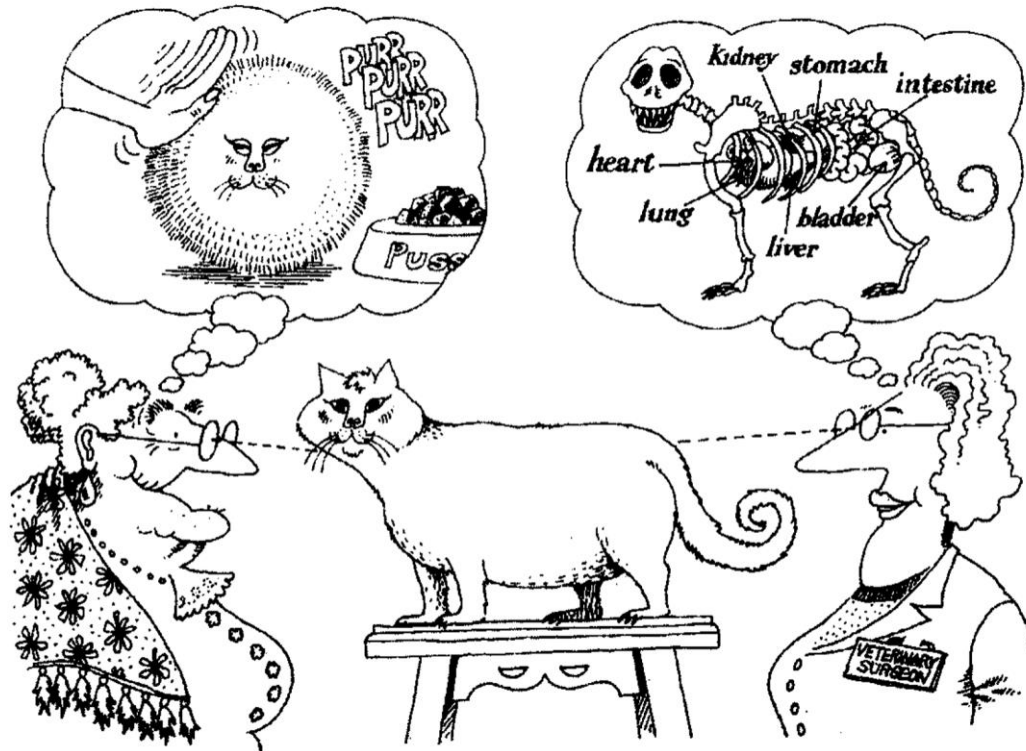


Abstração

- Existem visões diferentes?
 - A seleção de quais aspectos são essenciais **depende do observador** e do **fenômeno observado**



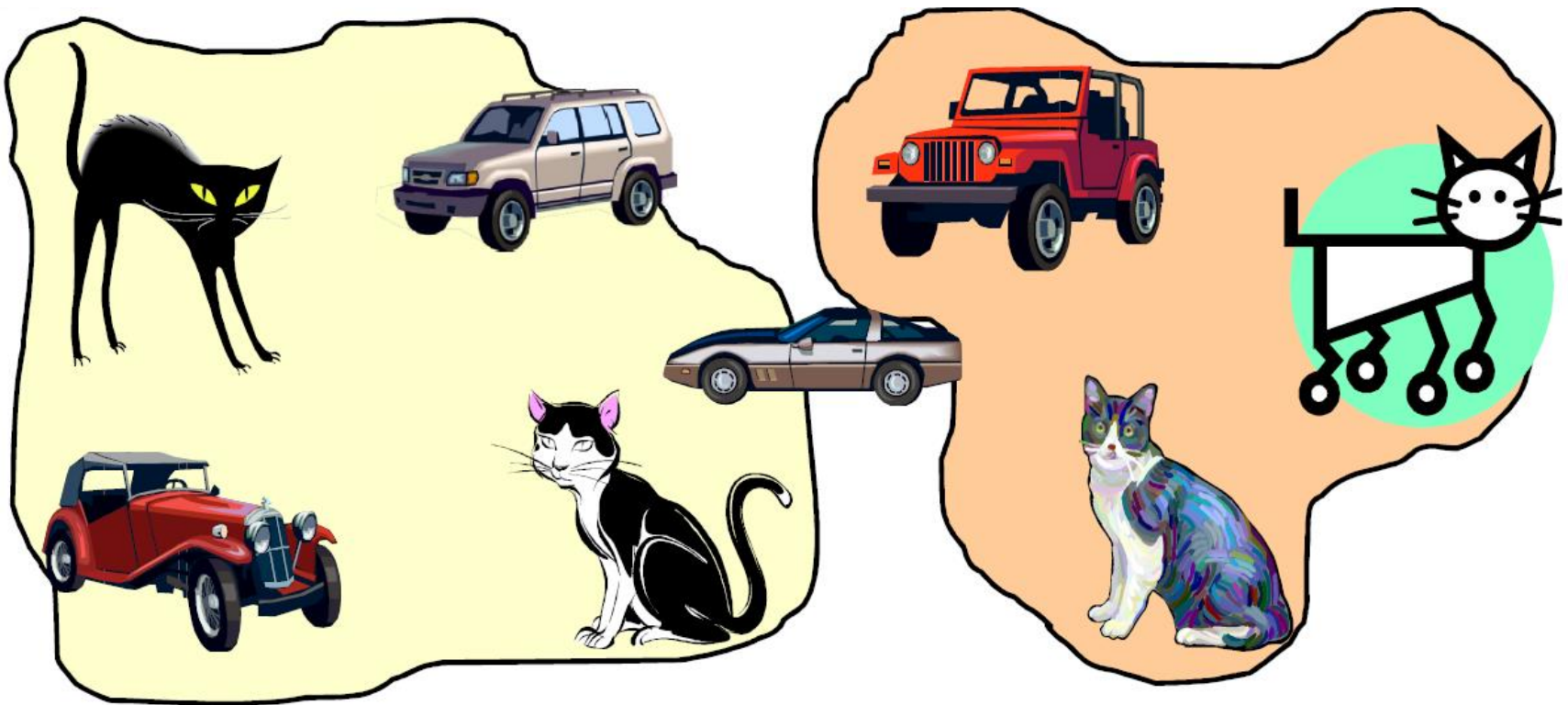
Diferentes
observadores



Diferentes
necessidades

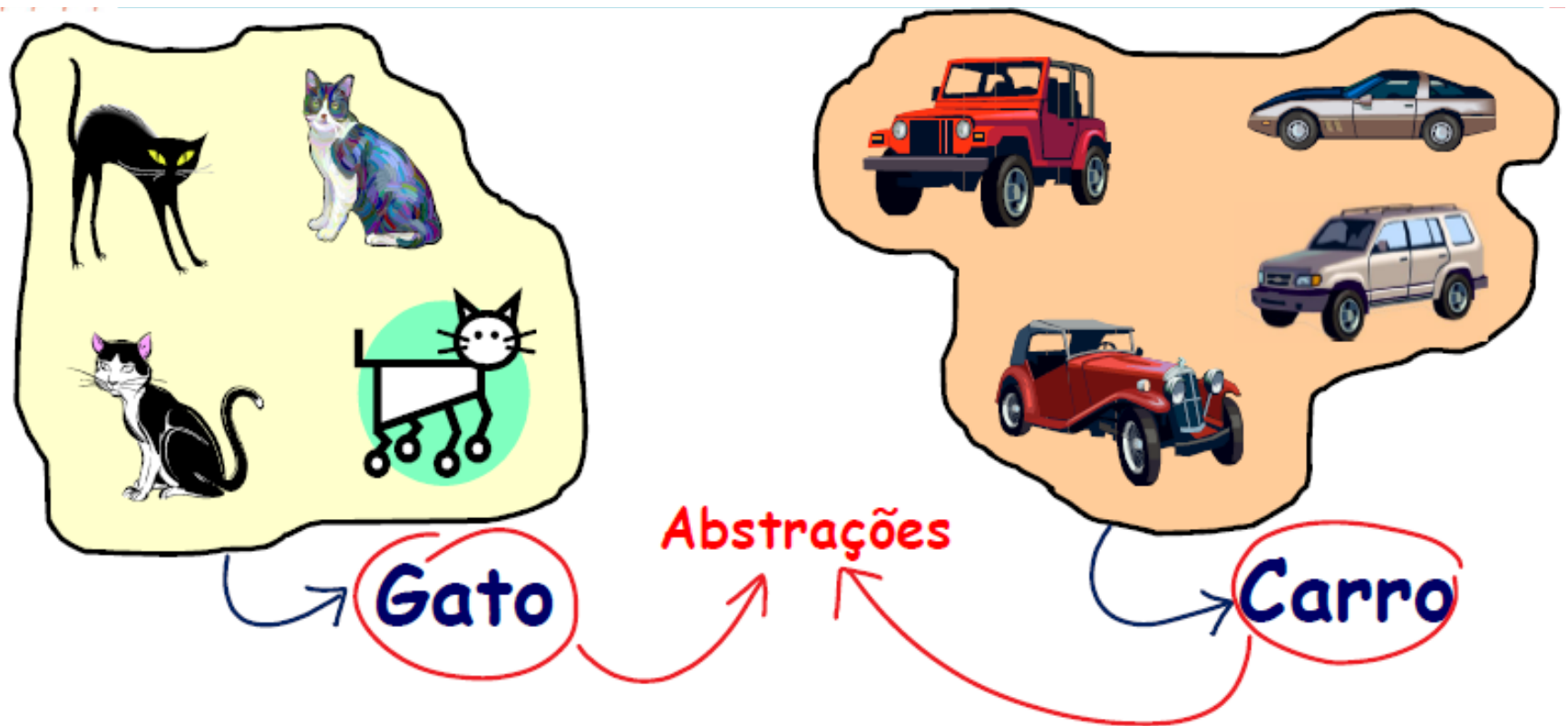
Abstração

- Como você classificaria estas “coisas”?



Abstração

- Como você classificaria estas “coisas”?



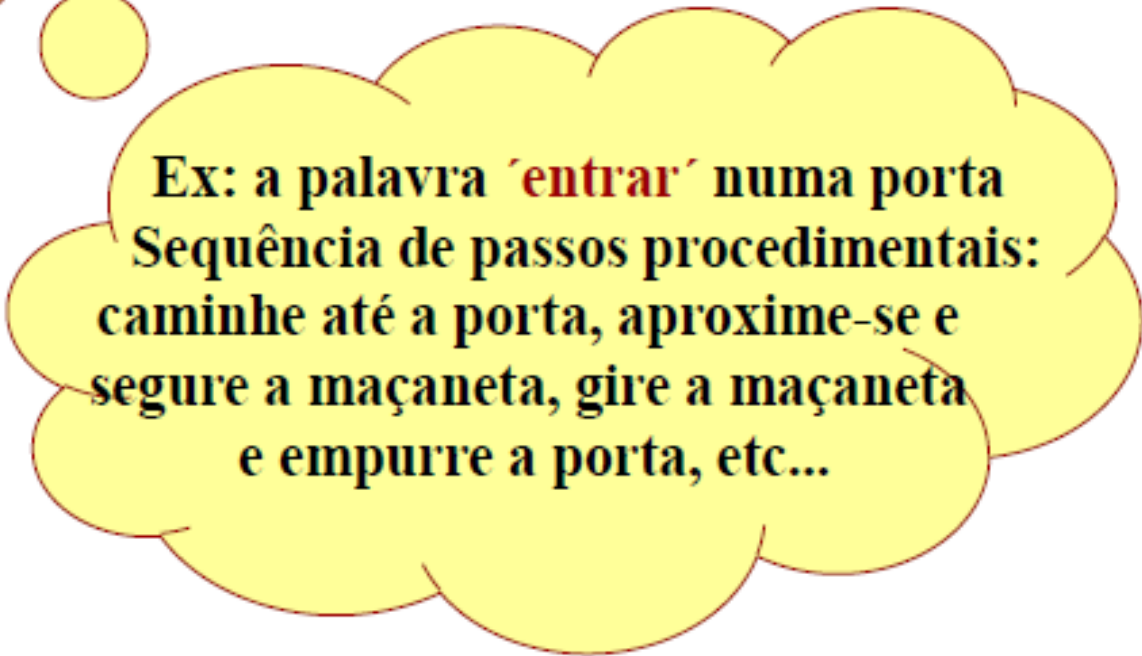
Abstração

- São criados somente os atributos e métodos necessários para o problema em mãos
- Quais seriam os atributos e métodos para o objeto Carro em cada uma das situações seguintes?
 - Sistema de uma locadora de carros
 - Sistema de uma revendedora de carros
 - Sistema de uma oficina mecânica
 - Sistema do DETRAN



Abstração Procedimental

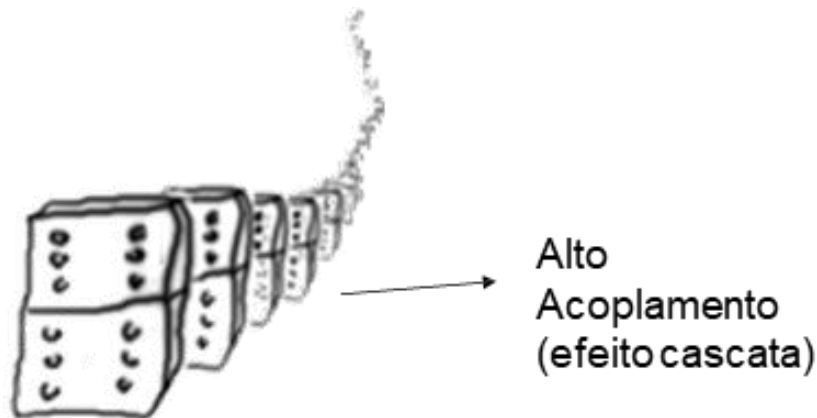
- Uma sequência de instruções designadas que têm uma função específica e limitada



Ex: a palavra 'entrar**' numa porta**
Sequência de passos procedimentais:
caminhe até a porta, aproxime-se e
segure a maçaneta, gire a maçaneta
e empurre a porta, etc...

Conceitos acoplamento

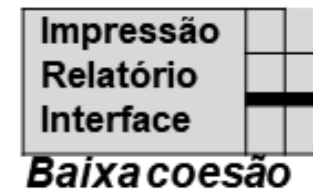
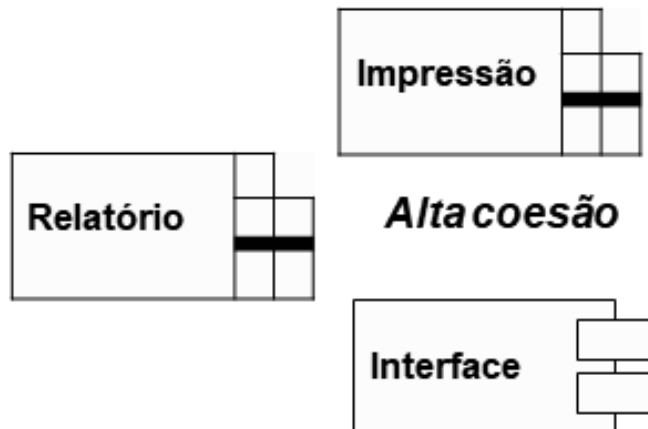
- Grau de interconexão entre diferentes pedaços de um sistema
- Pedaços menos acoplados são mais fáceis de entender, testar, reusar e manter
- Baixo acoplamento também promove o paralelismo de implementação



Conceitos

Coesão

- Quão proximamente são relacionadas as atividades dentro de um único pedaço (componente) ou entre um grupo de pedaços?
 - Componentes altamente coesos = relacionados a apenas UMA funcionalidade



Conceitos

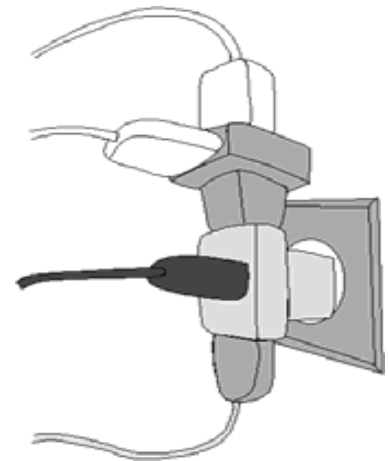
acoplamento e coesão

- **Lembrar:** prezar pela **alta coesão** e **fraco acoplamento**
- Uma das maneiras de se conseguir fraco acoplamento é com **interfaces**
- Coesão e acoplamento são conceitos interligados: **classes coesas tendem a gerar baixo acoplamento**

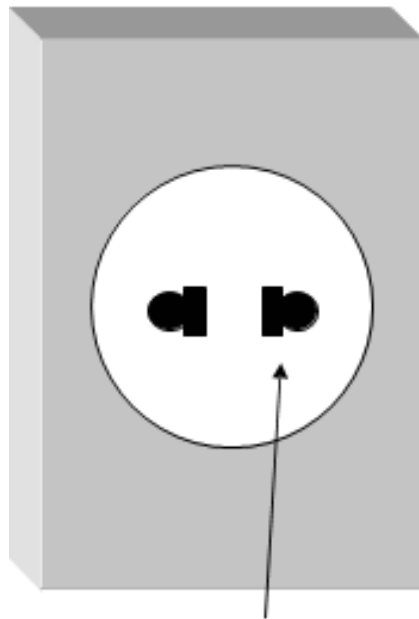


Conceitos interface

- As maneiras em que subsistemas dentro de um projeto maior interagem são claramente definidas.
- Idealmente, interações são especificadas de um modo que possam se manter relativamente estáveis ao longo do ciclo de vida do sistema.
- Um modo de alcançar isso é através de abstrações sobre a implementação concreta.



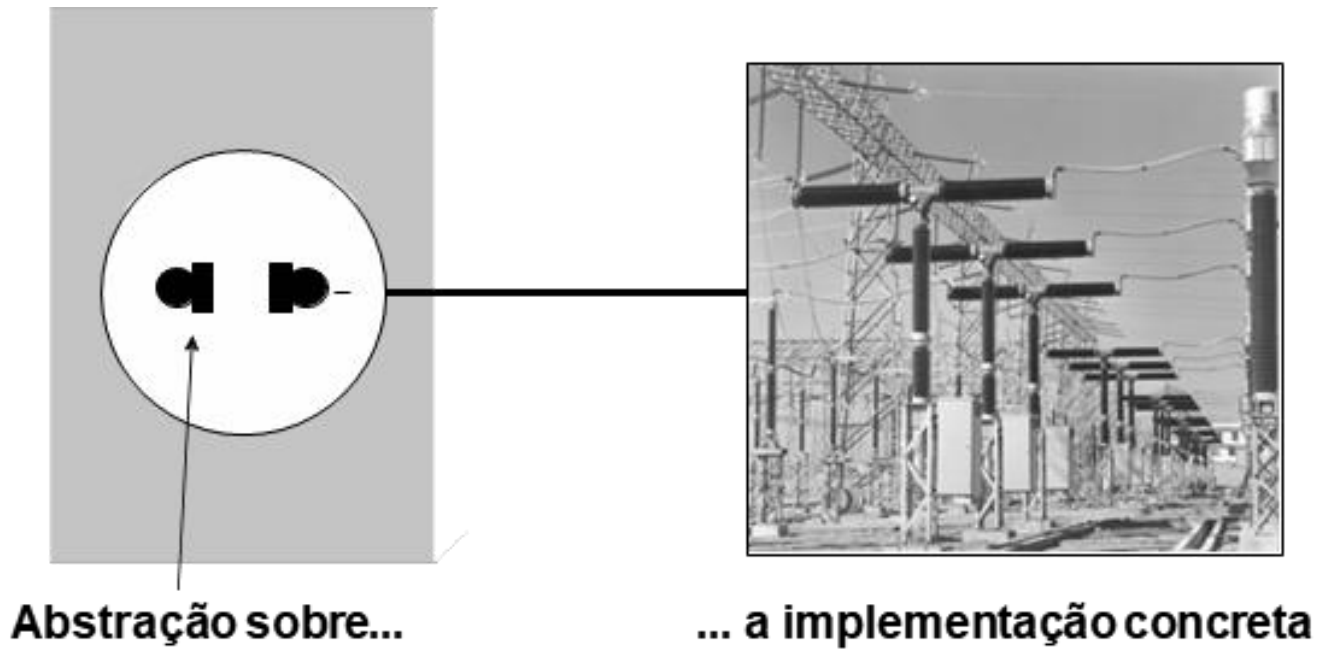
Conceitos interface



Interface bem definida

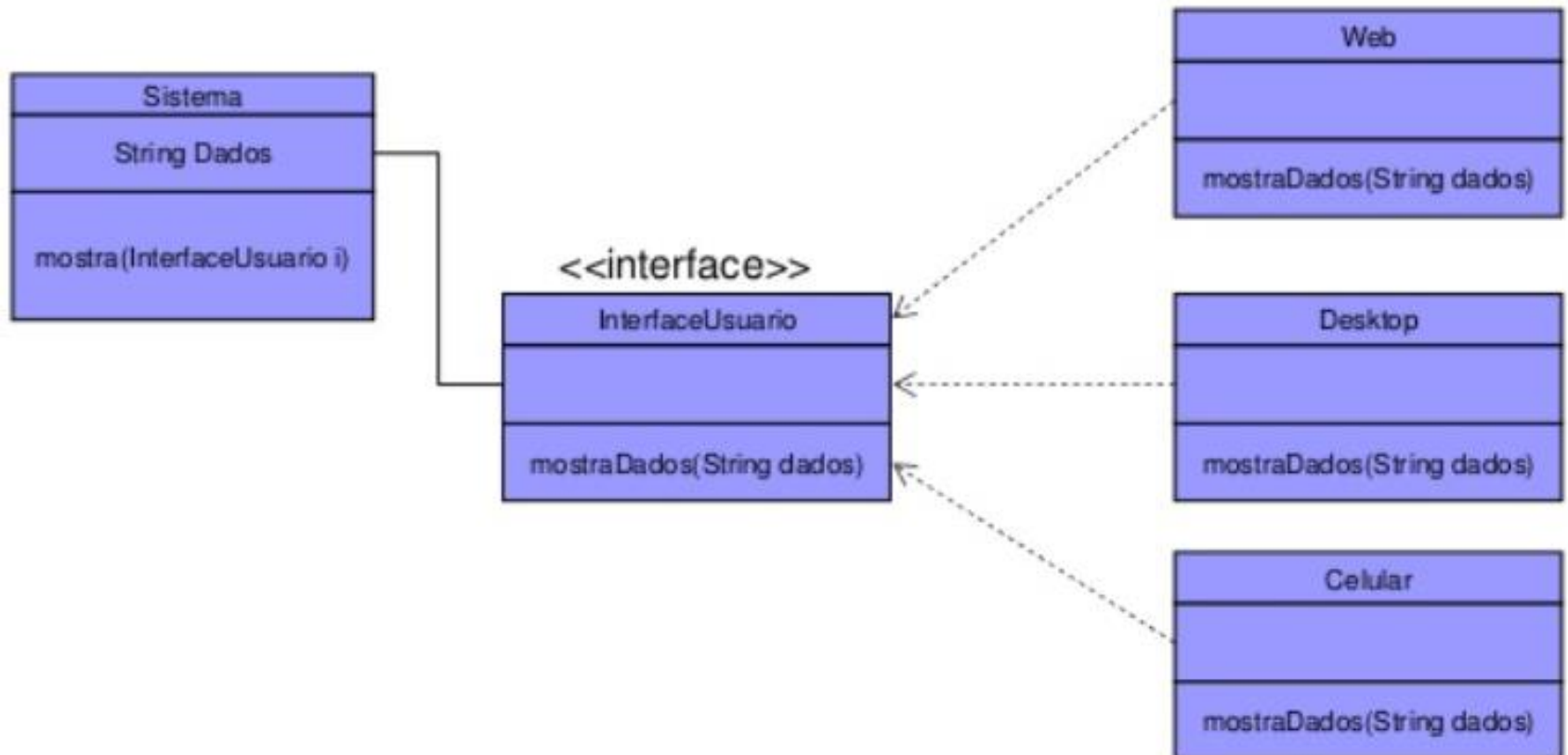


Conceitos interface



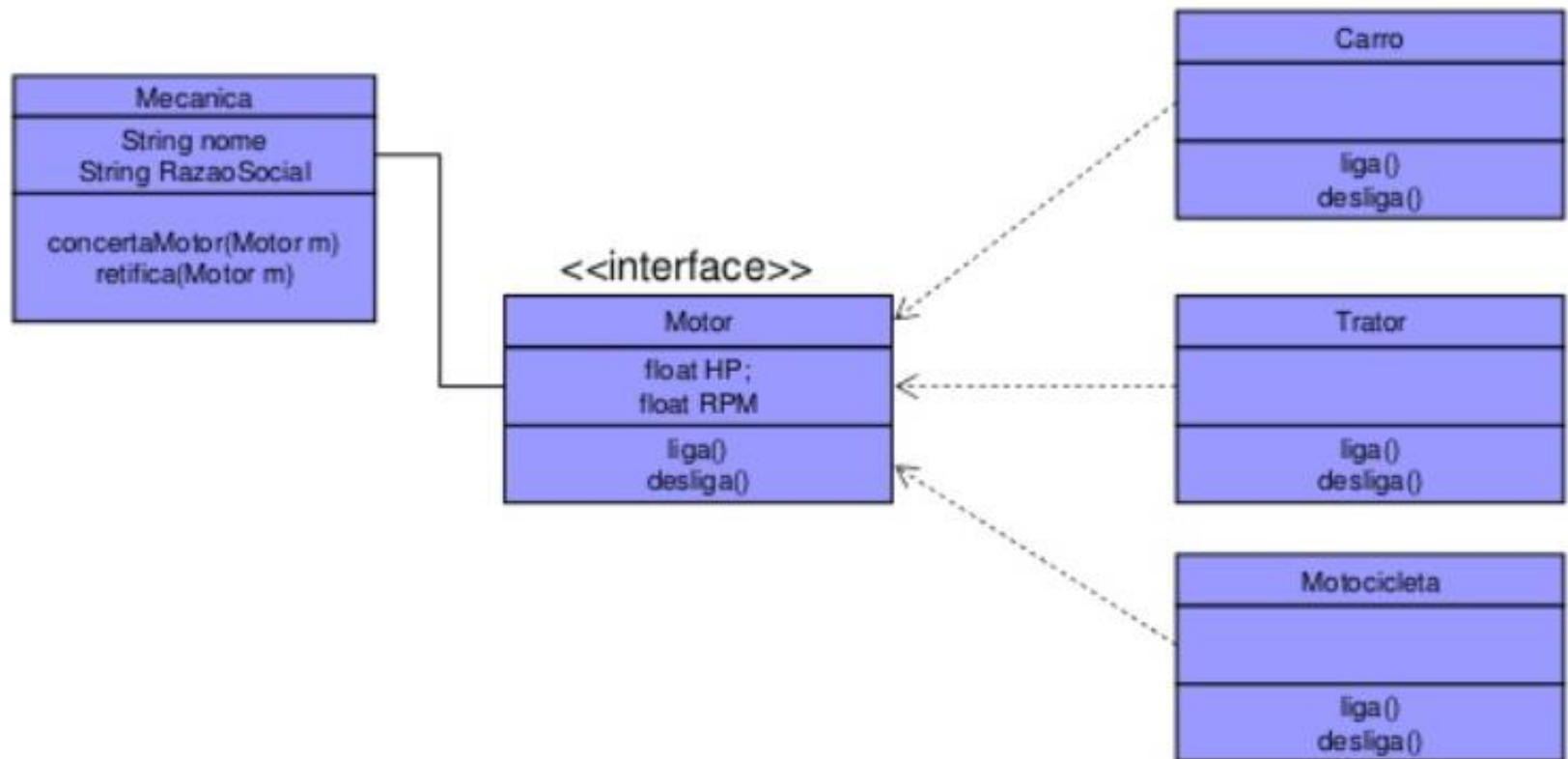
Conceitos interface

■ Exemplo de desacoplamento – Arquitetura

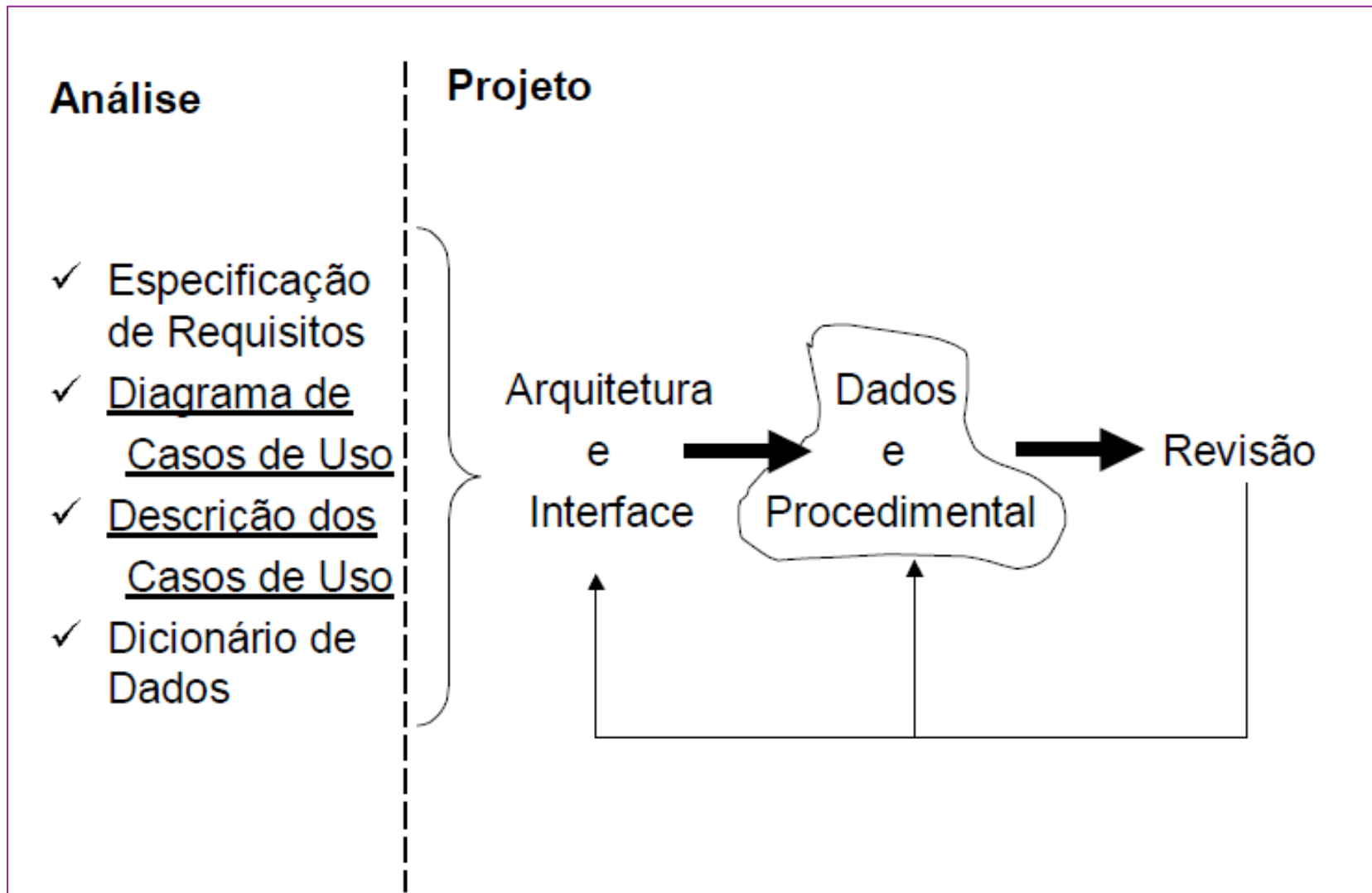


Conceitos interface

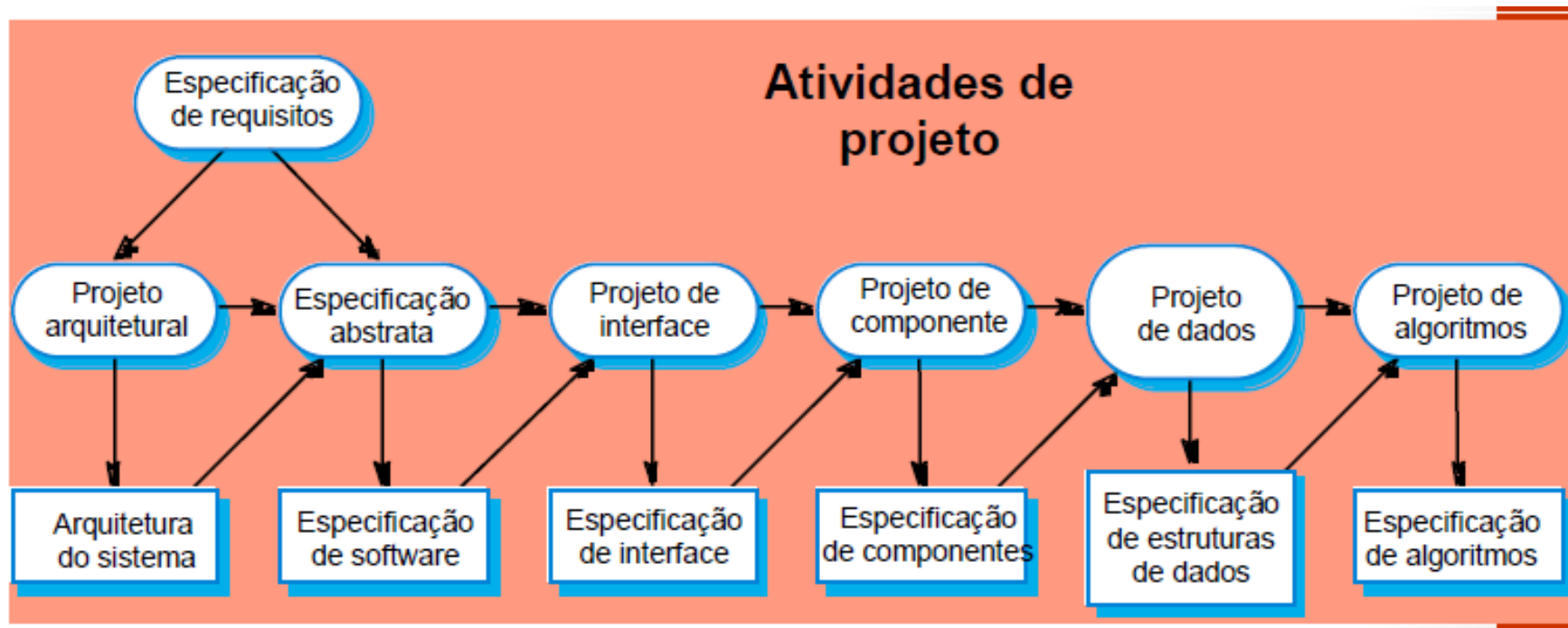
■ Exemplo de desacoplamento – Motores



Processo Orientado a Objetos

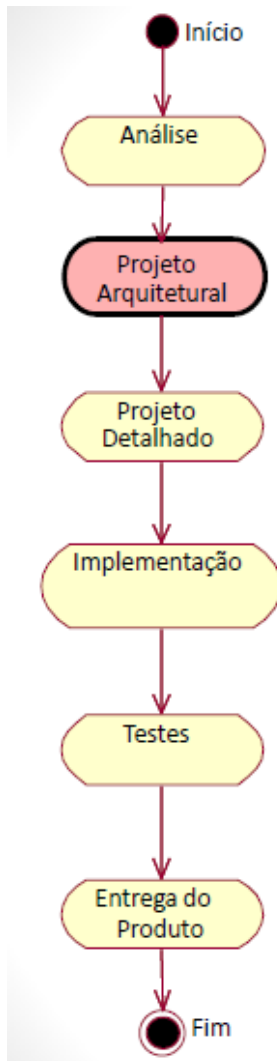


Processo de Software: Um modelo genérico de Processo



Um modelo genérico de Processo

Projeto arquitetural

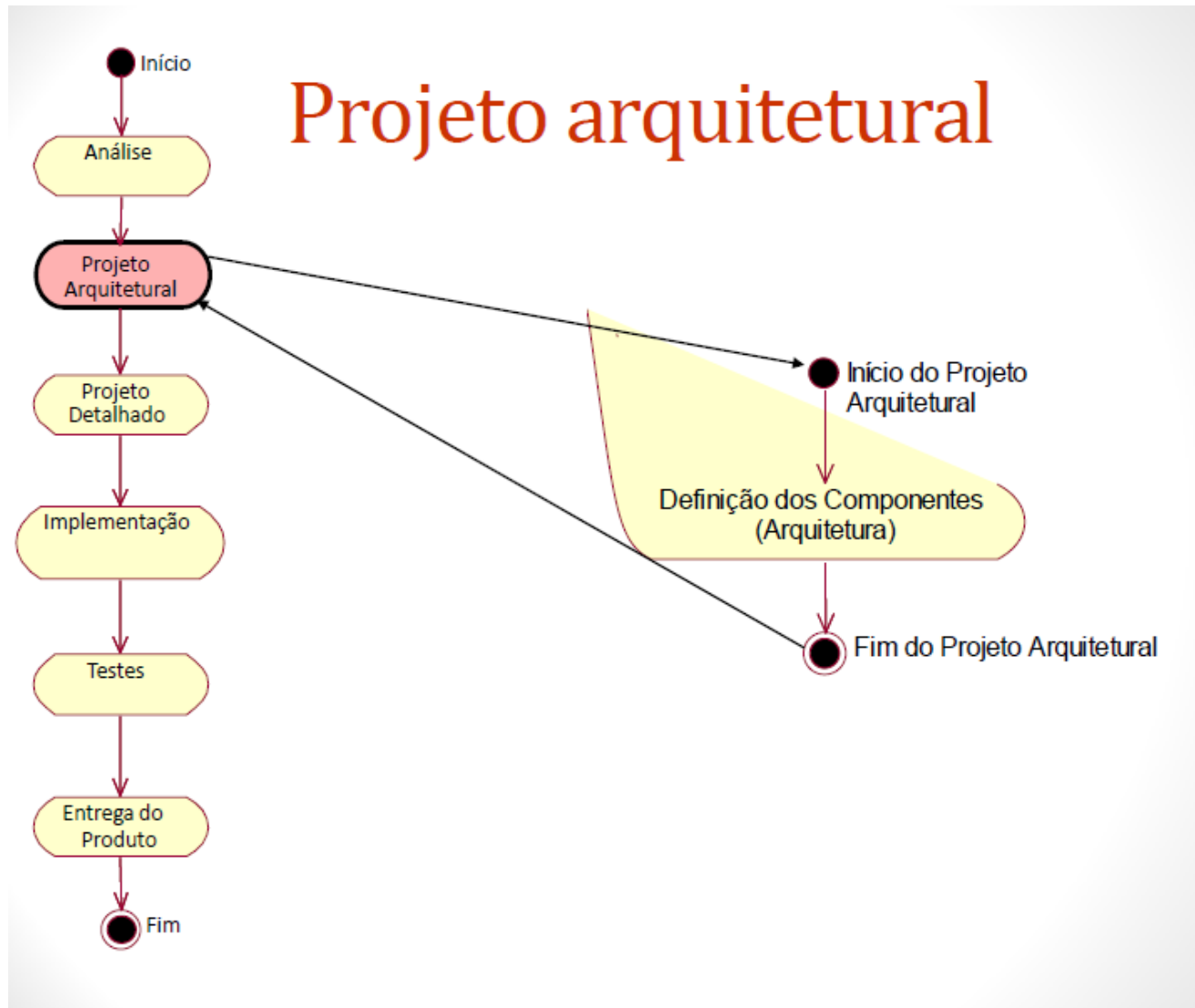


Entrada: Documentos Gerados na Análise
Documento de Especificação de Requisitos

Objetivo: Determinar os principais componentes do sistema e o relacionamento entre eles

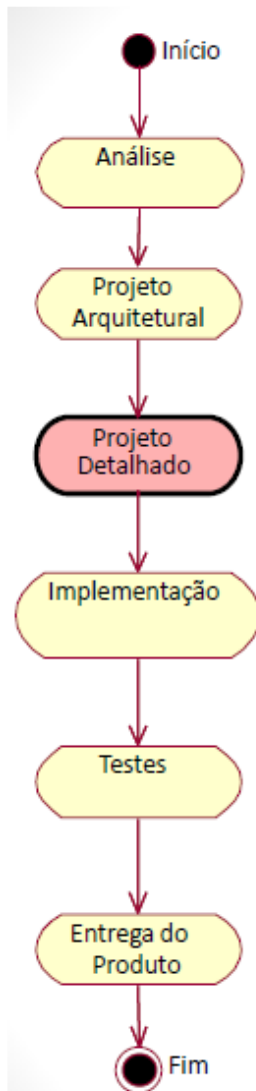
Saída: Diagrama Arquitetural (Alto Nível)

Um modelo genérico de Processo



Processo Orientado a Objetos

Projeto detalhado

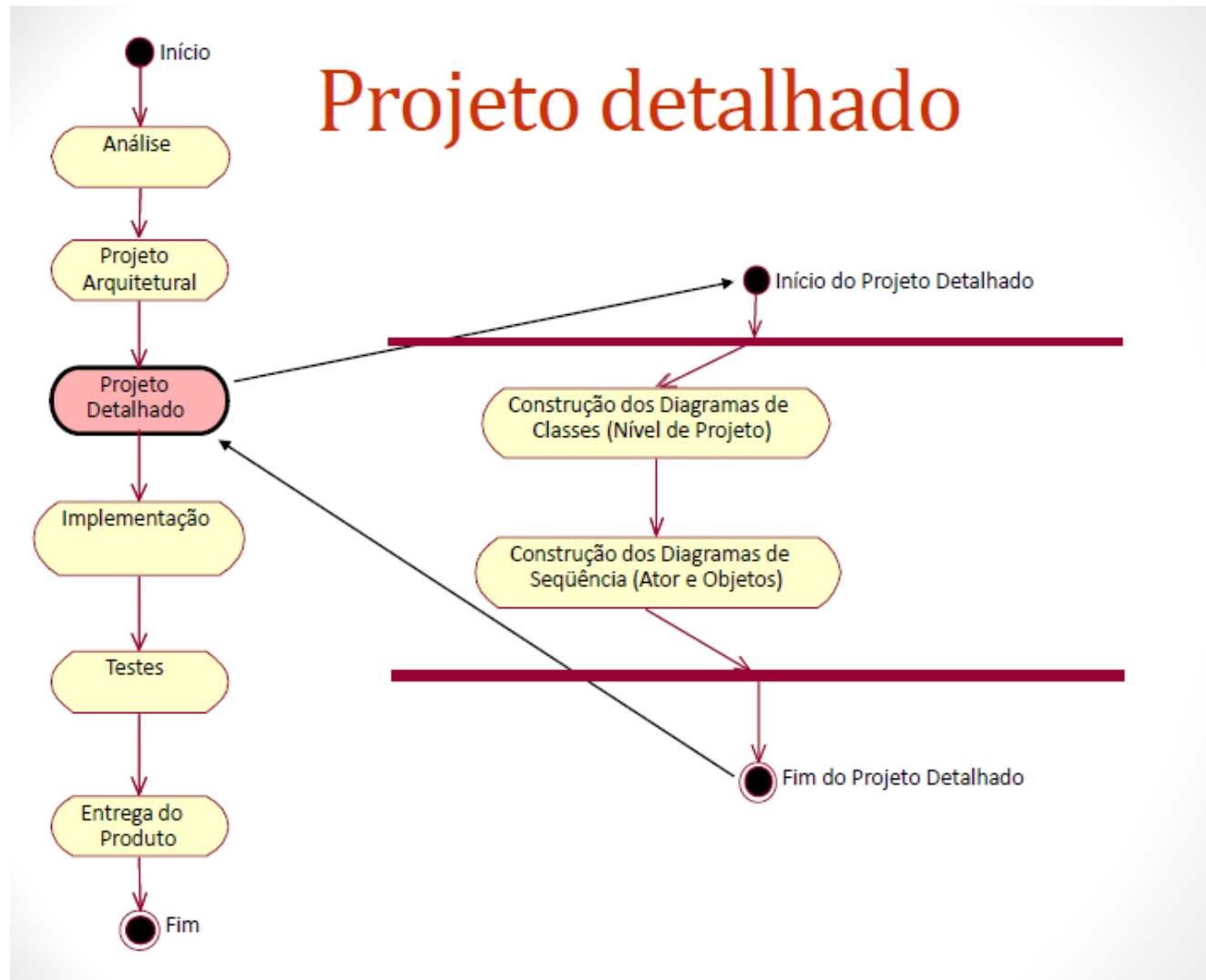


Entrada: Documentos Gerados na Análise
Documento de Especificação de Requisitos

Objetivo: Determinar uma solução para o sistema baseando-se nos documentos gerados na análise

Saída: 1) Diagrama de Classes (Nível de Projeto)
2) Diagramas de Seqüência (ator e objetos)

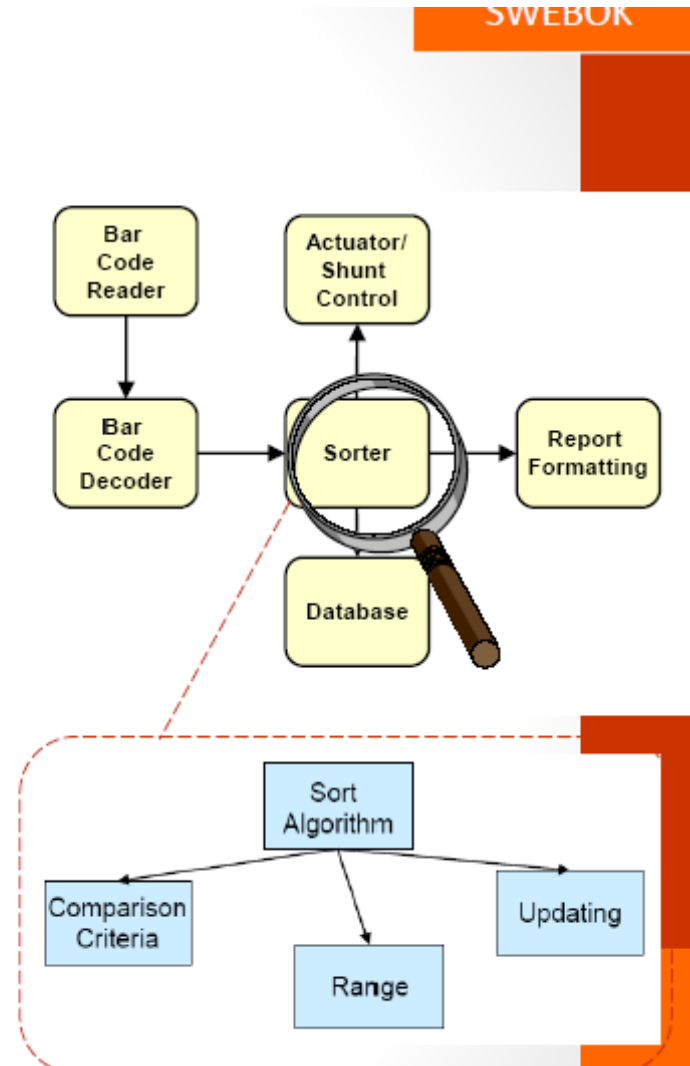
Processo Orientado a Objetos



Processo Orientado a Objetos

Projeto de software Perspectiva de resultado

- Projeto arquitetural
 - descrição da estrutura e organização de nível mais alto do software, e
 - identificação de componentes e relacionamentos
- Projeto detalhado
 - descrição de cada componente em nível de detalhe suficiente para permitir a sua construção
 - estrutura e comportamento



O que é APOO?

- n Na essência, É CONSIDERAR um problema e uma solução dentro da perspectiva de objetos, coisas ou conceitos.

- n O que é AOO?

- Investigação dos objetos de domínio e seus relacionamentos.

Descritos no *Modelo de Objetos de Domínio*

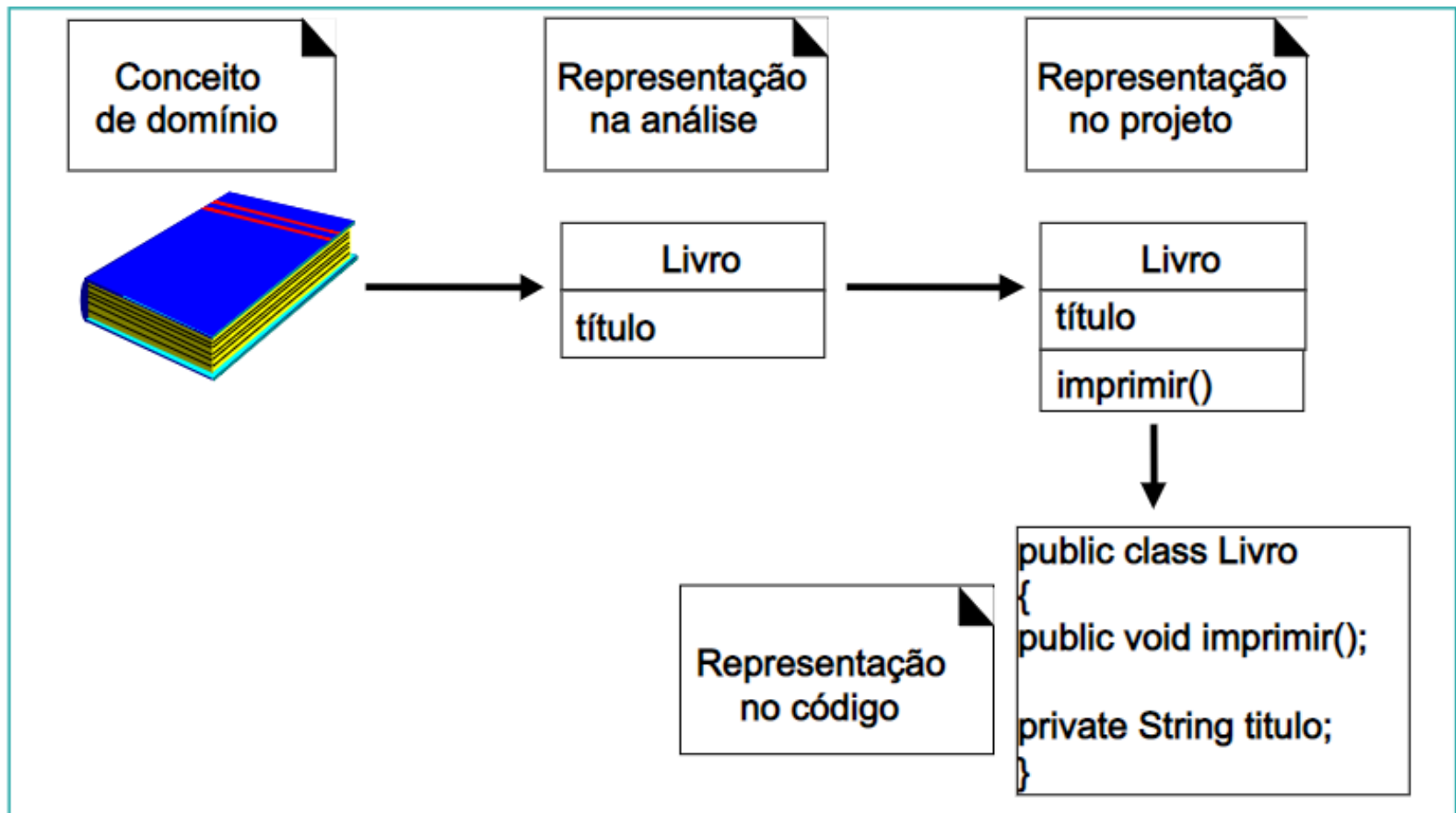
- n O que é POO?

- Elaboração de uma solução lógica em termos de componentes de software e suas colaborações e responsabilidades.

Descritos em *Diagramas de Classes* e *Diagramas de Interação*

Representação de um Conceito na APOO

Ex.: O conceito “Livro” em um sistema de biblioteca



Uma Analogia: Organizando os Negócios de uma Empresa

Analogia	APOO	Documentos Associados
Quais são os processos de negócio?	Análise de requisitos	Casos de uso
Quais são os papéis dos empregados?	Análise do domínio	Modelo conceitual
Quem é responsável por o quê? Como eles interagem?	Atribuição de responsabilidades, projeto das interações	Diagramas de classes de projeto, diagramas de colaboração / interação

Modelagem na APOO: um exemplo

Definir casos
de usos

Definir modelo
de domínio

Definir diagramas
de interação

Definir diagramas
de classes
de projeto

Exemplo: jogo de dados

- n Um jogo de dados no qual um jogador lança dois dados. Se o total for sete, ele vence; caso contrário, perde.



Um Exemplo — Jogo de Dados

**Definir casos
de usos**

Definir modelo
de domínio

Definir diagramas
de interação

Definir diagramas
de classes
de projeto

Caso de uso: Jogar

Atores: Jogador

Descrição: Este caso de uso começa quando um jogador pega e lança os dados. Se a soma do valor das faces dos dados totalizar sete, ele vence; caso contrário, perde.

- n **Casos de uso:** são descrições narrativas de processos do domínio no formato de prosa estruturada.

Um Exemplo — Jogo de Dados

Definir casos
de usos

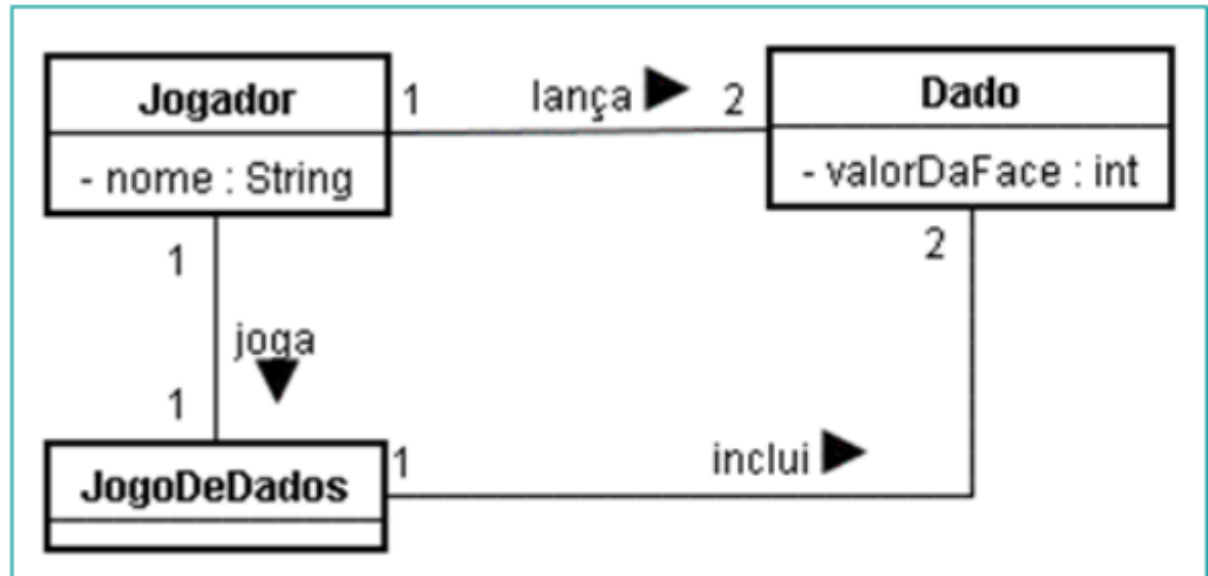
Definir modelo
de domínio

Definir diagramas
de interação

Definir diagramas
de classes
de projeto

Modelo conceitual:

Conceitos, atributos, e associações que são considerados importantes no domínio da aplicação.



- Um modelo conceitual descreve conceitos do mundo real, não componentes de software!

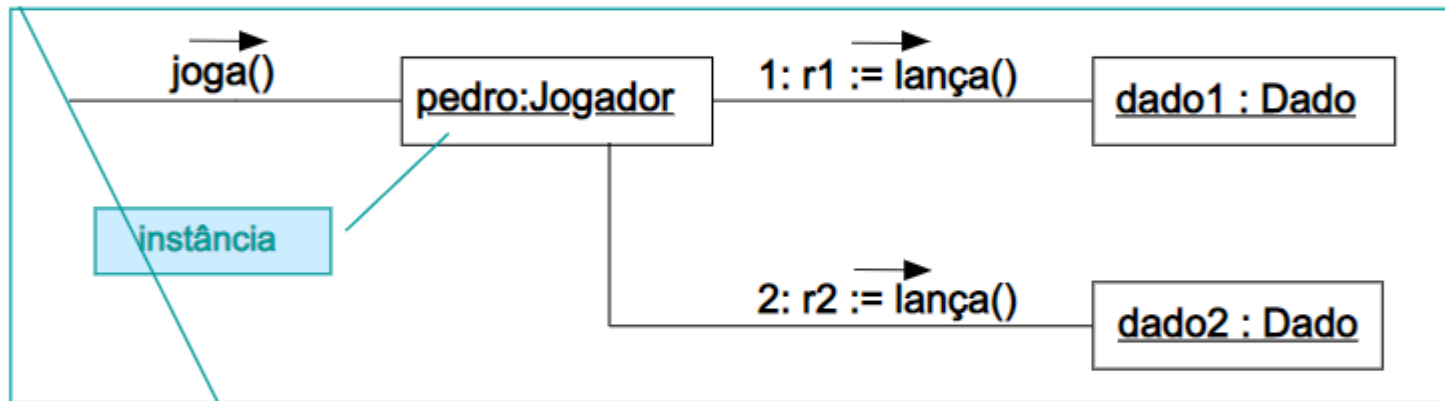
Um Exemplo — Jogo de Dados

Definir casos de usos

Definir modelo de domínio

Definir diagramas de interação

Definir diagramas de classes de projeto



- n Alocação de responsabilidades para objetos ilustrando como eles interagem via mensagens.
- n Mostram o fluxo de mensagens entre instâncias e a invocação de métodos.

Diagrama de Colaboração

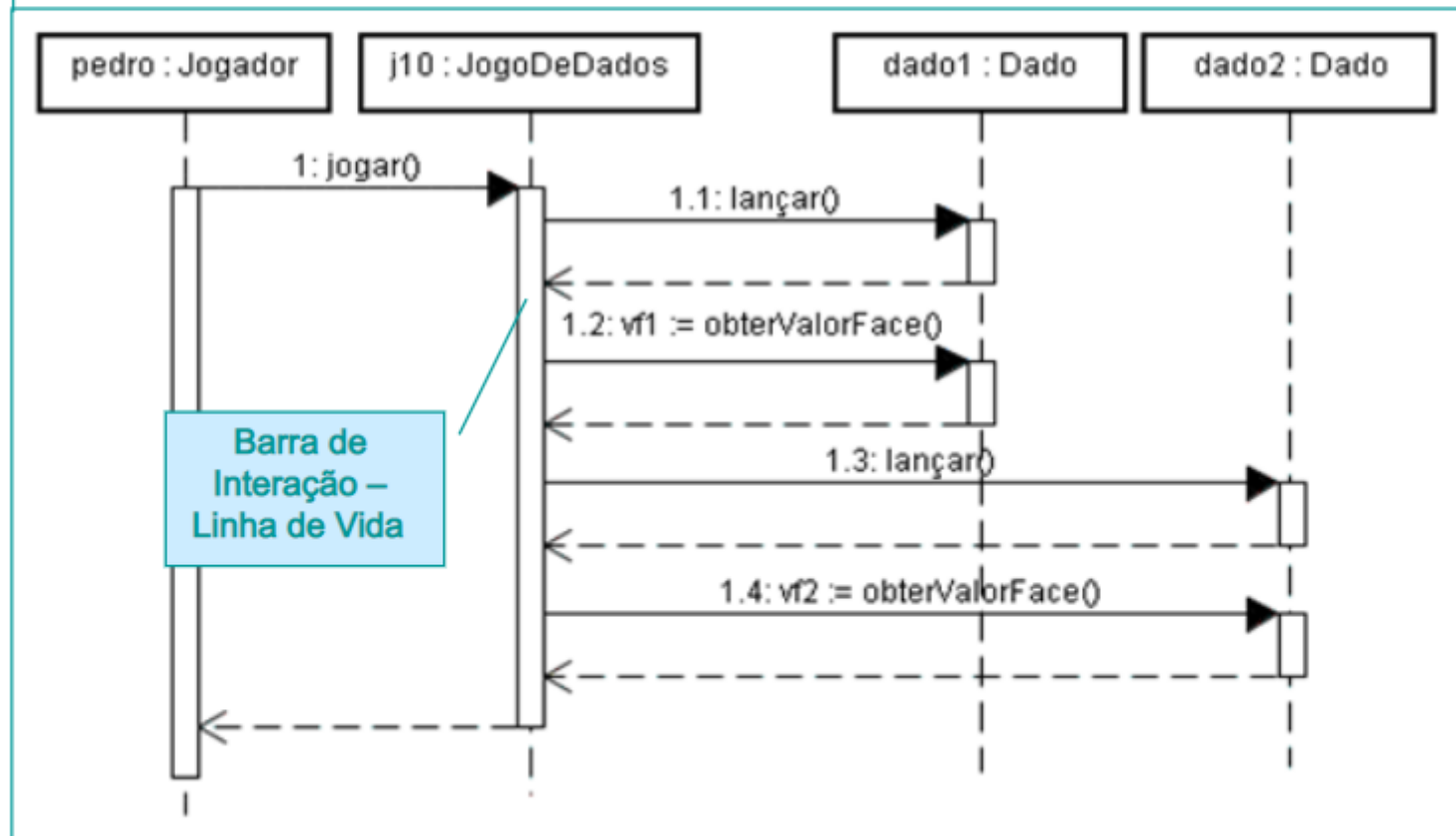
Exemplo — Jogo de Dados

Definir casos
de usos

Definir modelo
de domínio

Definir diagramas
de interação

Definir diagramas
de classes
de projeto



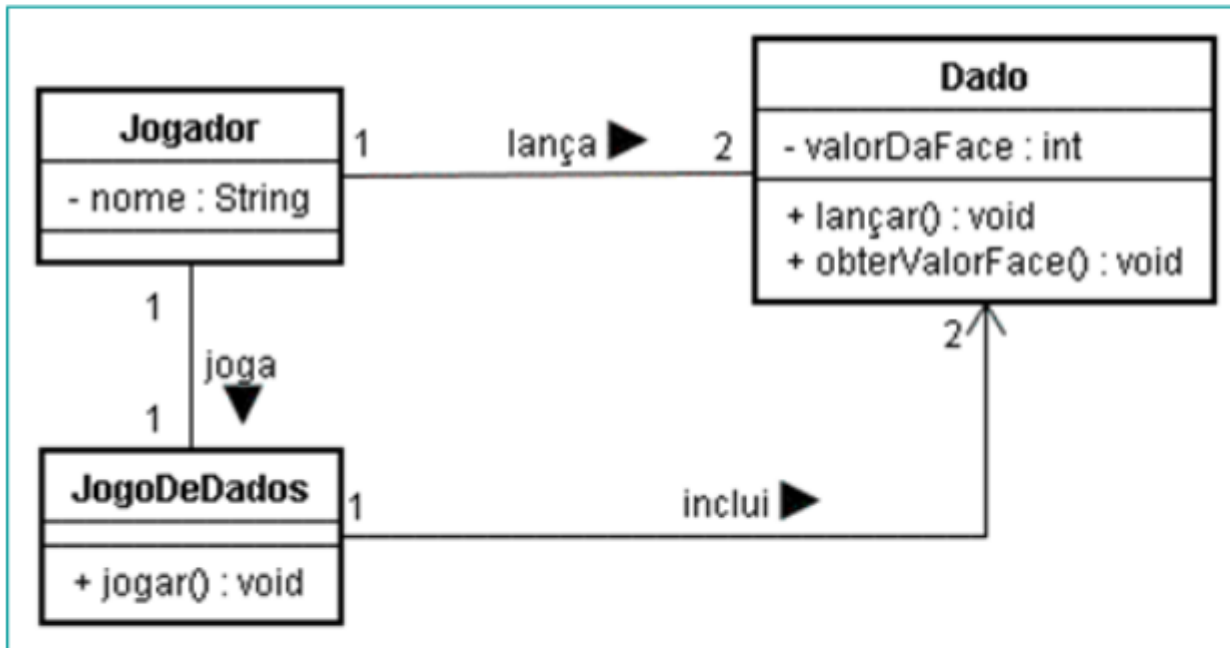
Um Exemplo — Jogo de Dados

Definir casos
de usos

Definir modelo
de domínio

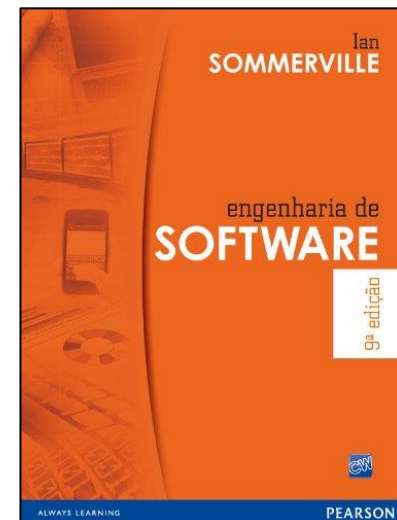
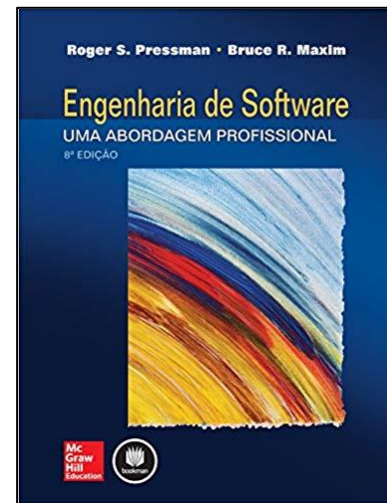
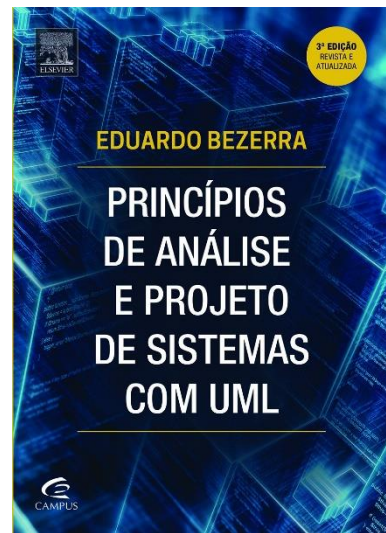
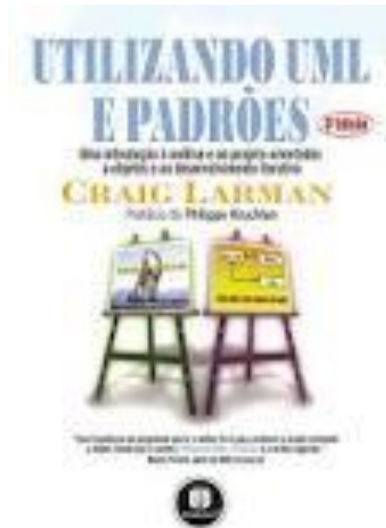
Definir diagramas
de interação

Definir diagramas
de classes
de projeto



- n Como os objetos (de software) se conectam?
- n Quais são os métodos de uma classe?

Referências Bibliográficas



Obrigada!



- **Perguntas?**

E-mail: jacilane.rabelo@ufc.br