Análise e Projeto de Sistemas de Informação

Andrêza Leite
andreza.lba@gmail.com

Roteiro

- Sistemas de Informação
- Ciclo de Desenvolvimento de SI
- Projeto
- Análise Estruturada
- Análise Orientada a Objetos
- Como fazer um bom projeto
- Padrões de Projeto

Sistema::Conceito

- Conjunto uniforme de grupos de itens independentes em interação regulamentada, tendo em vista a concretização de objetivo definidos.

- Conjunto organizado de doutrinas, idéias ou princípios, normalmente com a intenção de explicar a disposição ou um todo sistemático.

Informação::Conceito

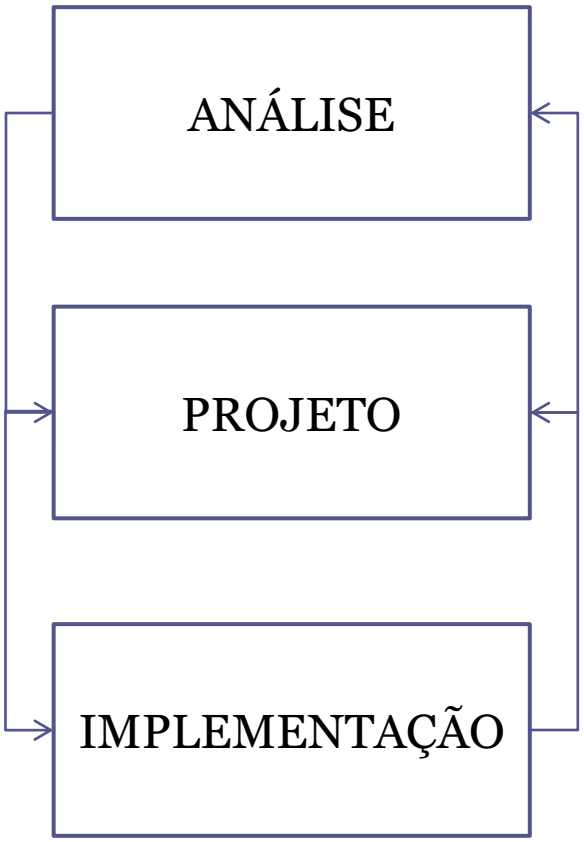
- Informação é o resultado do processamento, manipulação e organização de dados, de tal forma que represente uma modificação (quantitativa ou qualitativa) no conhecimento do sistema que a recebe (pessoa, animal ou máquina).

Sistemas de Informação::Conceito

- Um SI é uma combinação de pessoas, dados, processos, interfaces, redes de comunicação e tecnologia que interagem com o objetivo de dar suporte e melhorar o processo de negócio de uma organização com relação às informações.

- Sistema capaz de receber, processar, memorizar e produzir informação, de modo a colocá-la à disposição dos utilizadores, quando e onde necessário.

SI::Ciclo de desenvolvimento



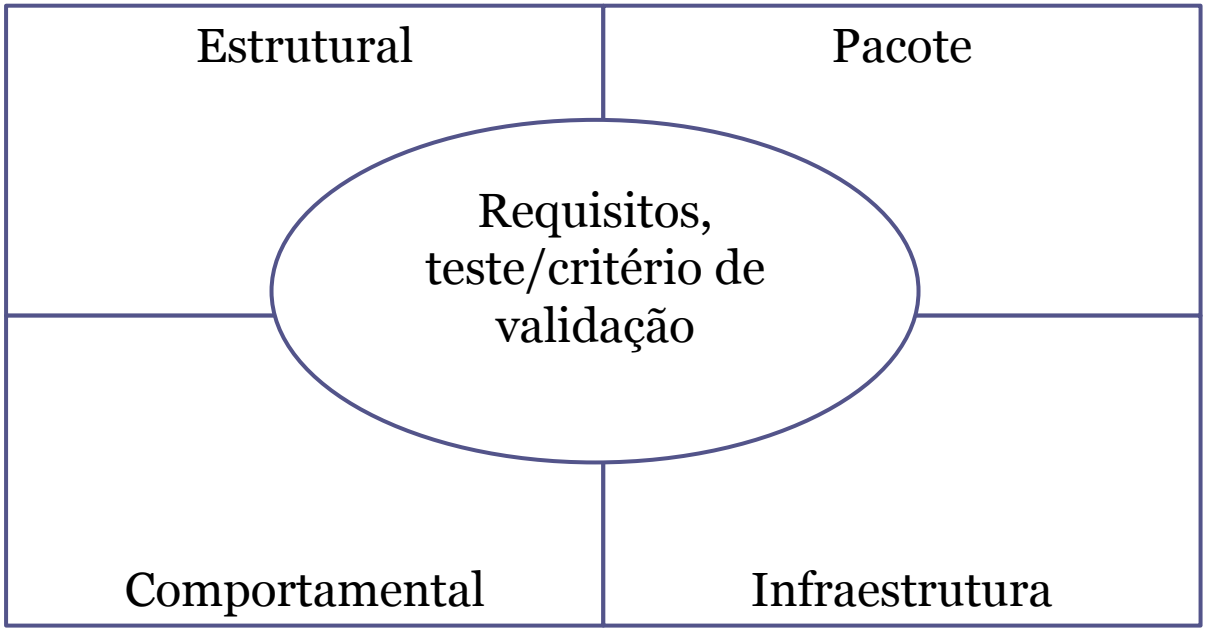
- Estudo do sistema;
- Proposta de soluções;
- Estudo de viabilidade;

- Definição das necessidades funcionais;
- Preparação das especificações para a implementação;

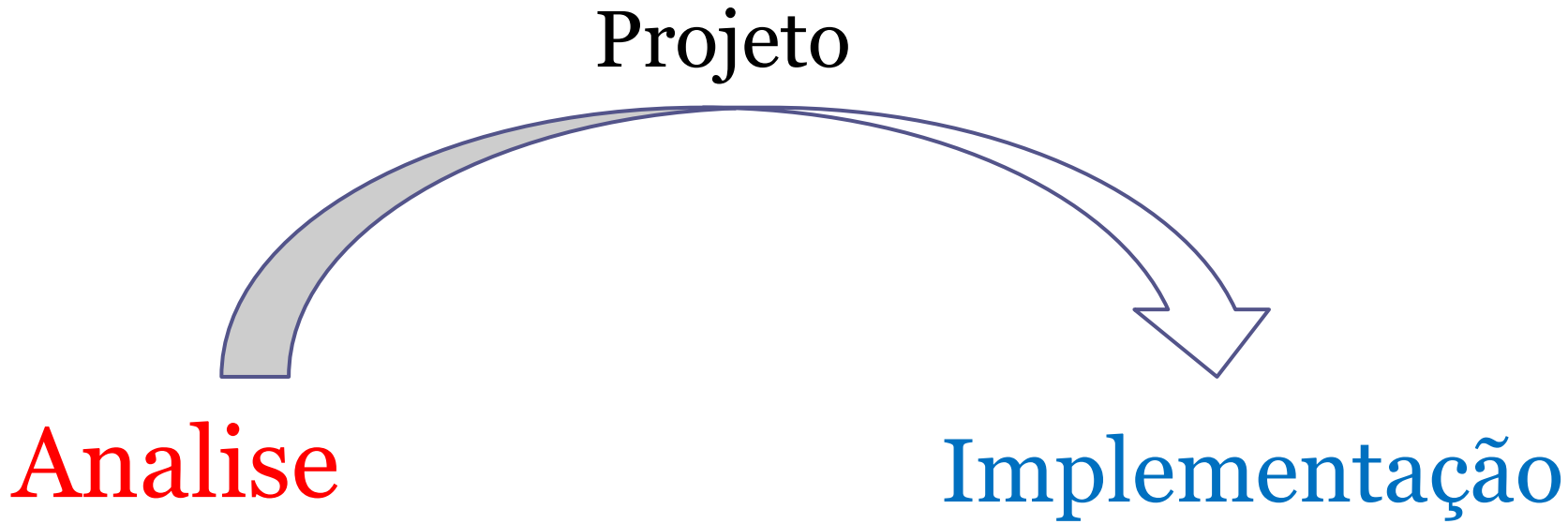
- Programação;
- Instalação.

Projeto::Conceito

- É uma descrição precisa de um sistema sobre uma variedade de perspectivas



Desenvolvimento de SI::Projeto



Desenvolvimento de SI::Projeto

- Definição das necessidades funcionais
- Descrição detalhada das funções
- Entradas e Saídas desejadas
- Disponibilidades de dados
- Ligações com outros sistemas de processamento de dados

Análise

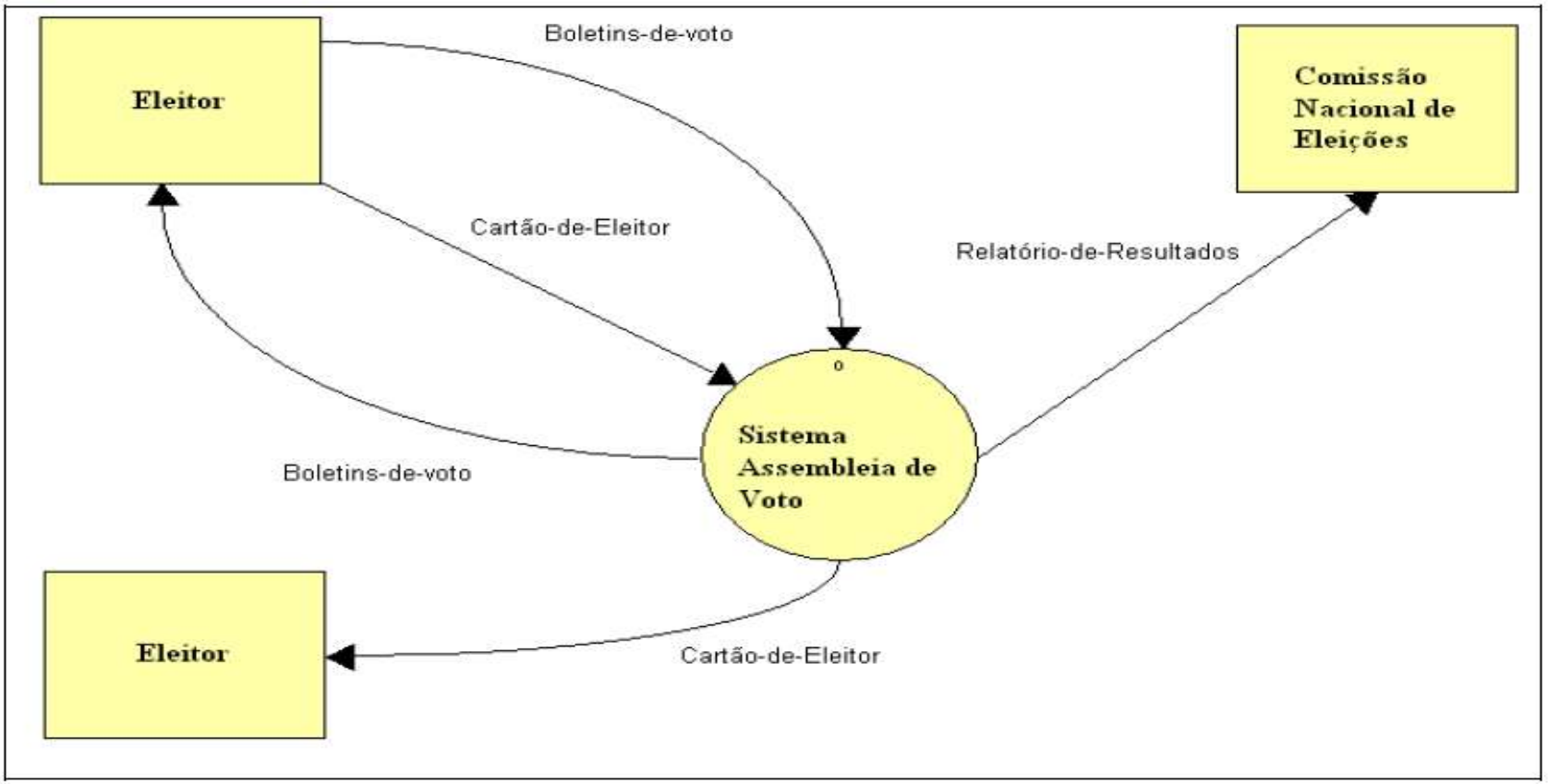
- Estruturada
  - Modelo Essencial ou Lógico constitui-se de dois sub-modelos (Modelo Ambiental e Modelo Comportamental) e um Dicionário de Dados.
  - Linguagens: Fortran, Cobol, C, etc.

- Orientada a Objetos
  - Modelo Funcional, Modelo Estrutural e Modelo Comportamental.
  - Linguagens: Java, C++, etc.

Análise Estruturada

- **Modelo Ambiental:** Define a fronteira entre o sistema e o seu ambiente externo.
  - Diagrama de contexto
- **Modelo Comportamental:** Descreve o comportamento do interior do sistema.
- **Dicionário de Dados**: É uma listagem organizada de todos os elementos de dados do sistema.

Modelo Ambiental::Diagrama de contexto

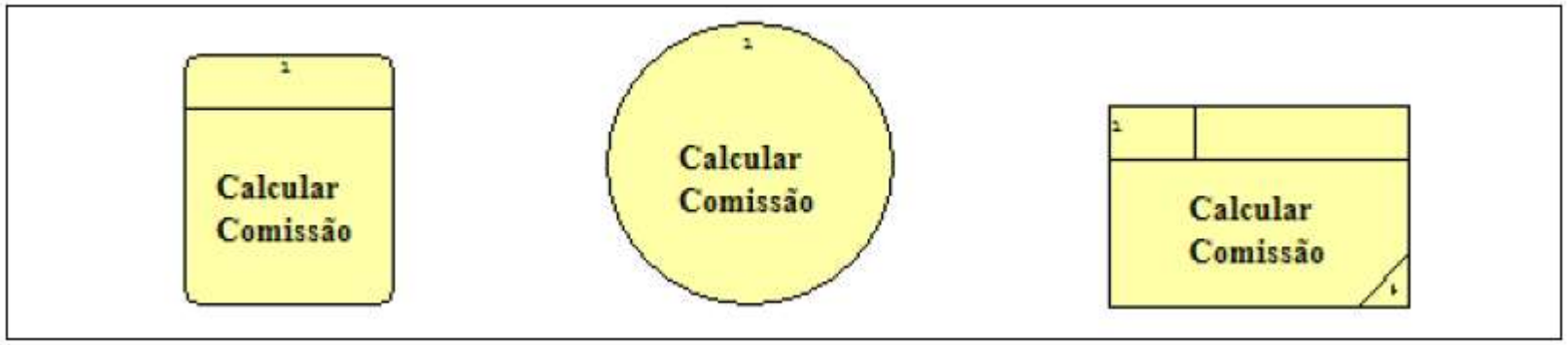


Análise Estruturada::Modelo Comportamental

- Diagramas de Fluxos de Dados (DFD);
- Diagramas de Entidades Relacionamento (ER);
- Diagramas de Transição de Estados (DTE).

Modelo Comportamental::DFD

Processo

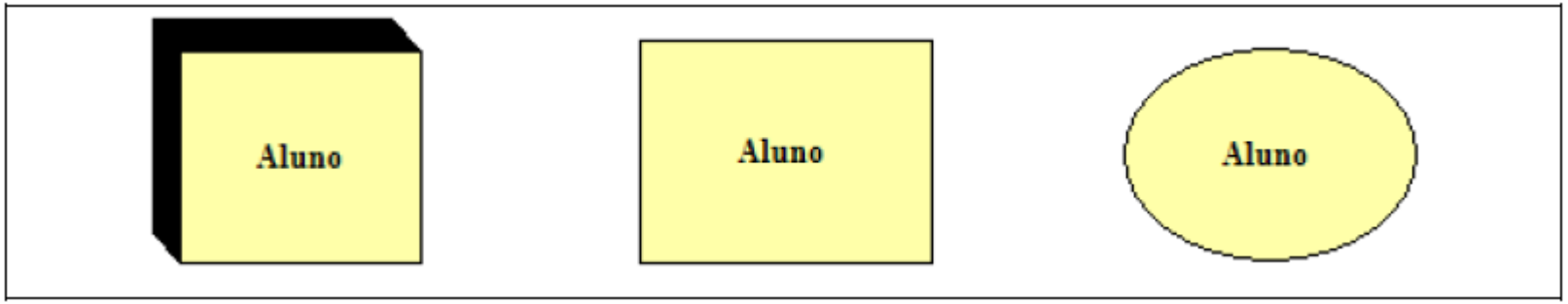


Fluxo de Dados

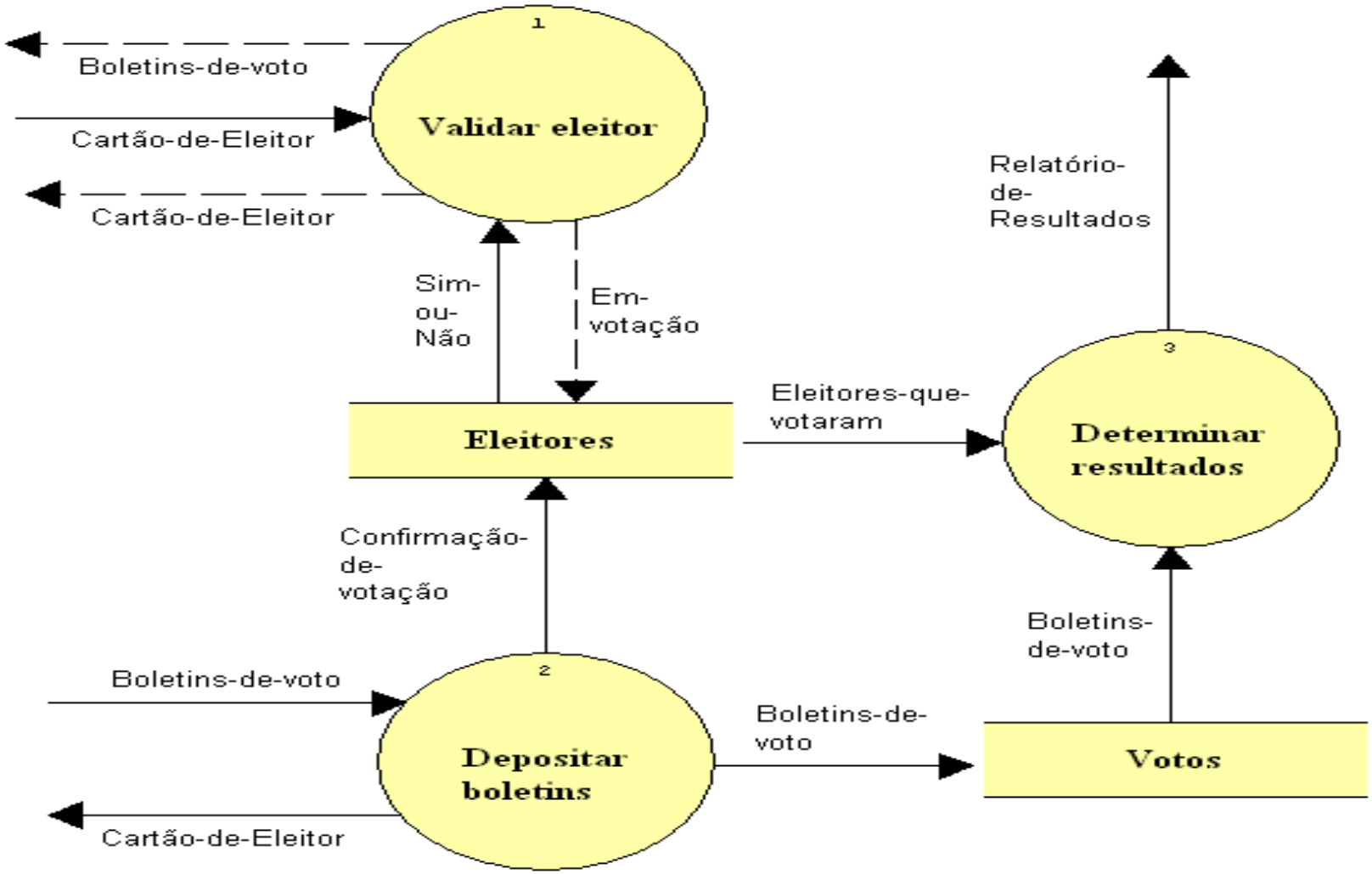
Deposito de Dados

Modelo Comportamental::DFD

Entidades Externas ou Terminadores



Modelo Comportamental::DFD



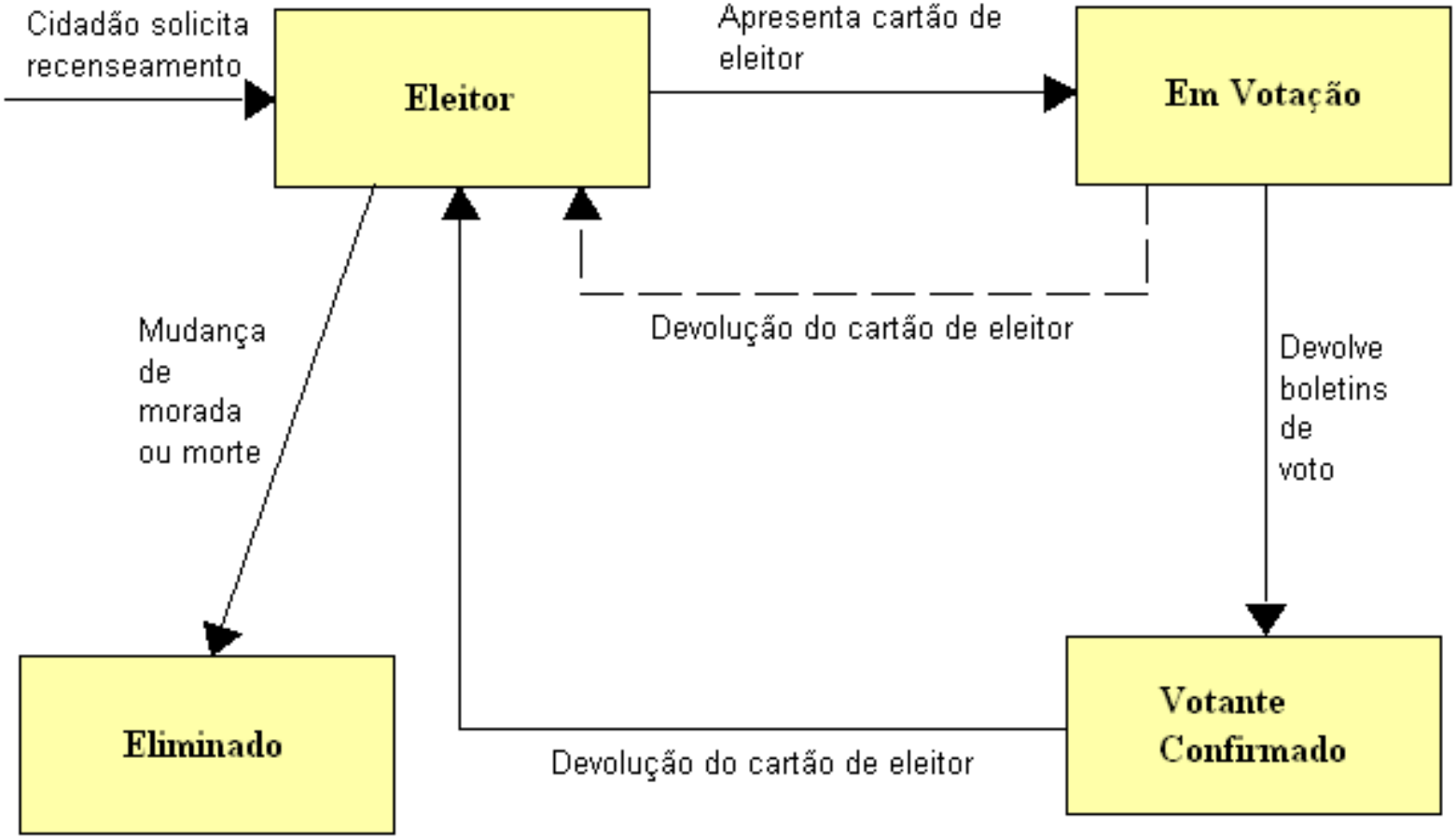
Modelo Comportamental::Diagramas ER

Desenvolvimento do Modelo de Dados Inicial

Modelo Comportamental::DTE

- Um DTE representa todas as seqüência de eventos que podem ocorrer durante o ciclo de vida de uma ocorrência de uma entidade no SI.

Modelo Comportamental::DTE



Análise Estruturada::Dicionário de Dados

- O dicionário de dados é uma listagem organizada de todos os elementos de dados do sistema, com definições precisas e rigorosas.

Análise Estruturada::Dicionário de Dados

- Elementos de dados;
- Estruturas de dados – grupos de elementos de dados.
- Fluxo de dados - um pacote de dados;
- Depósitos – uma coleção de pacotes de dados;

Análise OO

- As metodologias de análise OO procuram sistematizar quer a informação do sistema quer o processamento que manipula essa informação, através de objetos do mundo real.
  - Modelo Funcional,
  - Modelo Estrutural e
  - Modelo Comportamental.

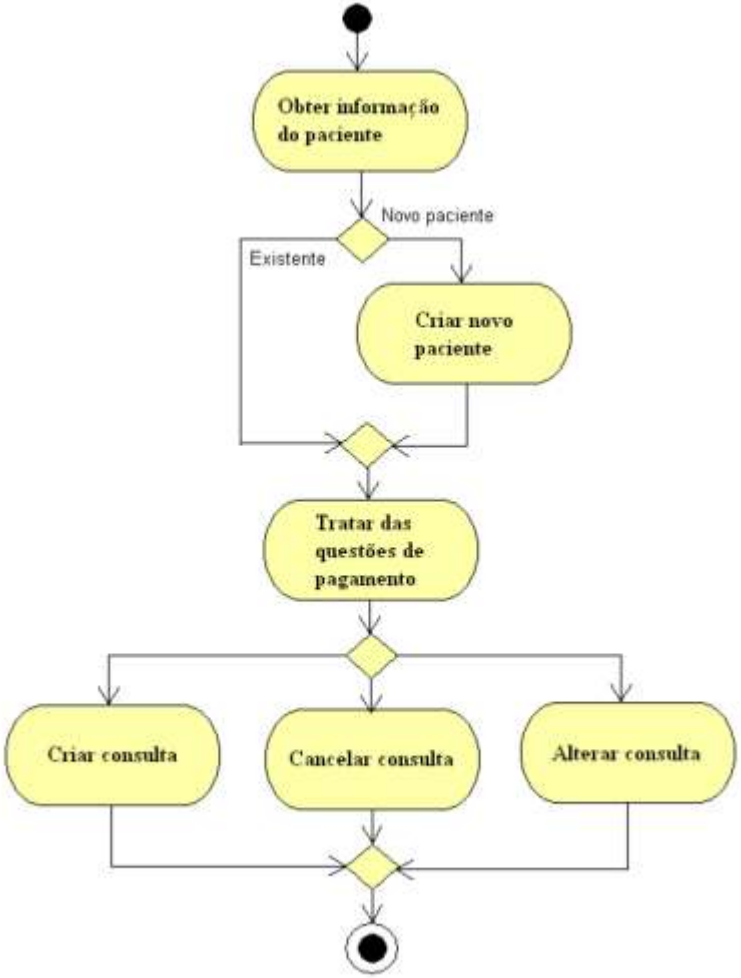
Análise OO::Modelo Funcional

- Descreve os processos de negócio e a interação do SI com o seu ambiente.
  - Diagramas de Atividades;
  - Diagramas de Casos de Uso.

Funcional::Diagrama de Atividades

- Determinar o contexto ou foco do processo a ser modelado de modo a encontrarmos um nome adequado para o diagrama.
- Identificar as atividades, fluxos de controle e fluxos de objeto que ocorram entre atividades.
- Identificar as decisões que fazem parte do processo a ser modelado.
- Identificar eventuais perspectivas de paralelismo no processo.
- Desenhar o diagrama.

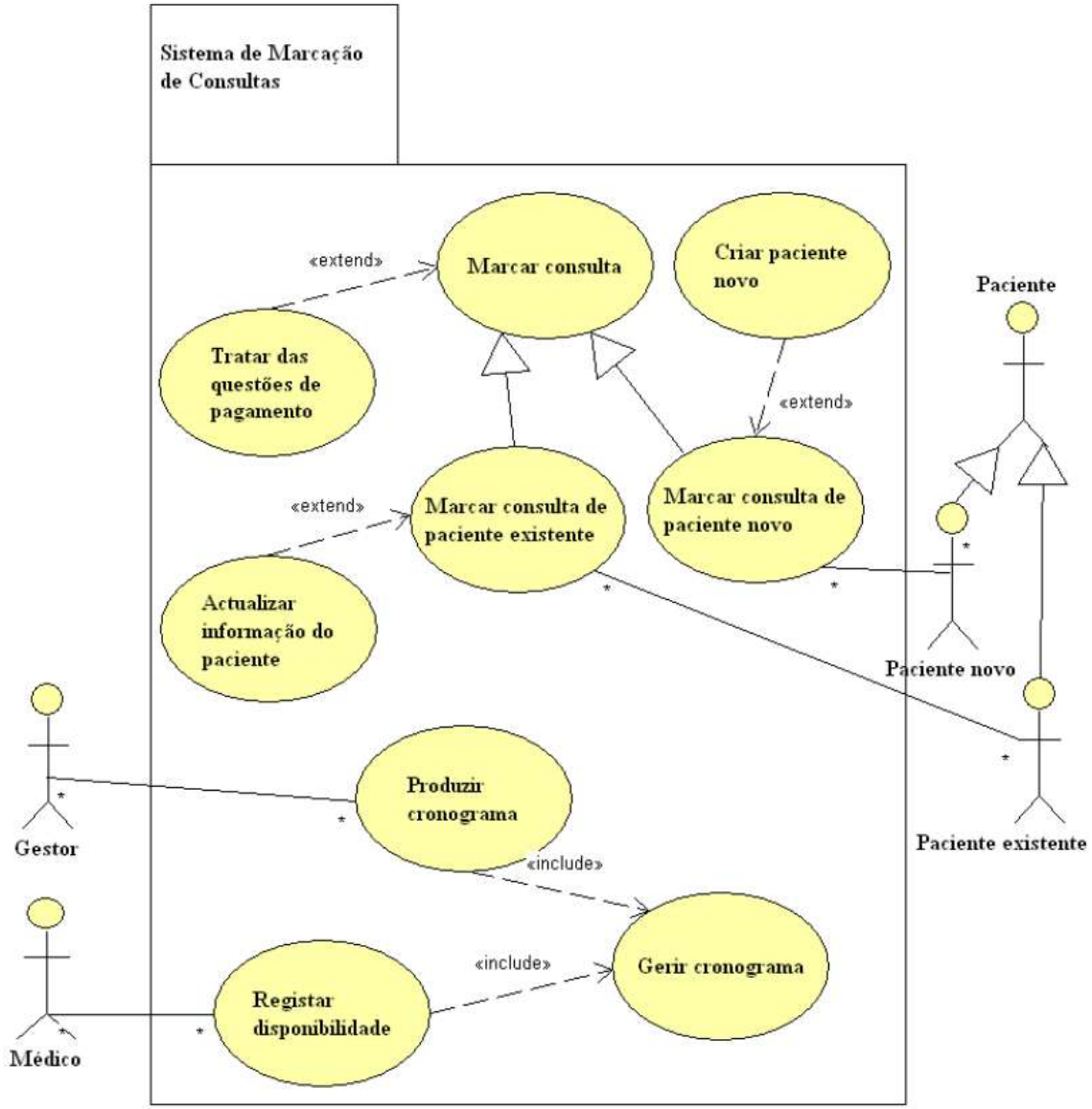
Funcional::Diagrama de Atividades



Funcional::Diagrama de casos de uso

- Identificar os casos de uso principais
- Expandir os casos de uso principais
- Confirmar os casos de uso principais
- Criar o Diagrama.

Modelo Funcional::Diagrama de casos de uso



Análise OO:: Modelo Estrutural

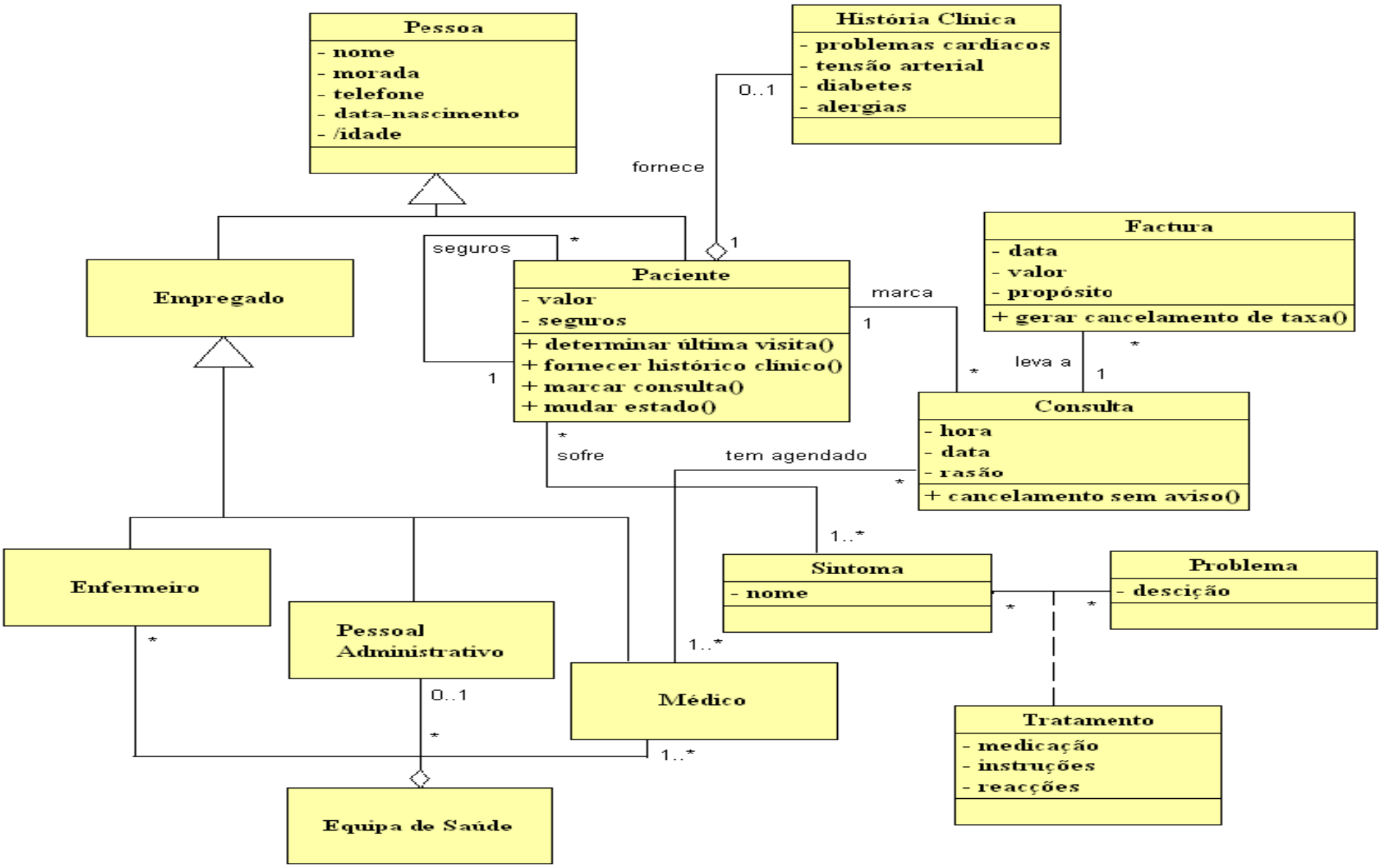
Descreve a estrutura dos dados que suportam os processos de negócio nas organizações.

- Descrição Classes-Responsabilidades-Colaborações (CRC);
- Diagramas de Classes;
- Diagramas de Objetos.

Modelo Estrutural:: Descrições CRC

organização lógica dos dados, sem indicar como os dados são criados, armazenados ou manipulados

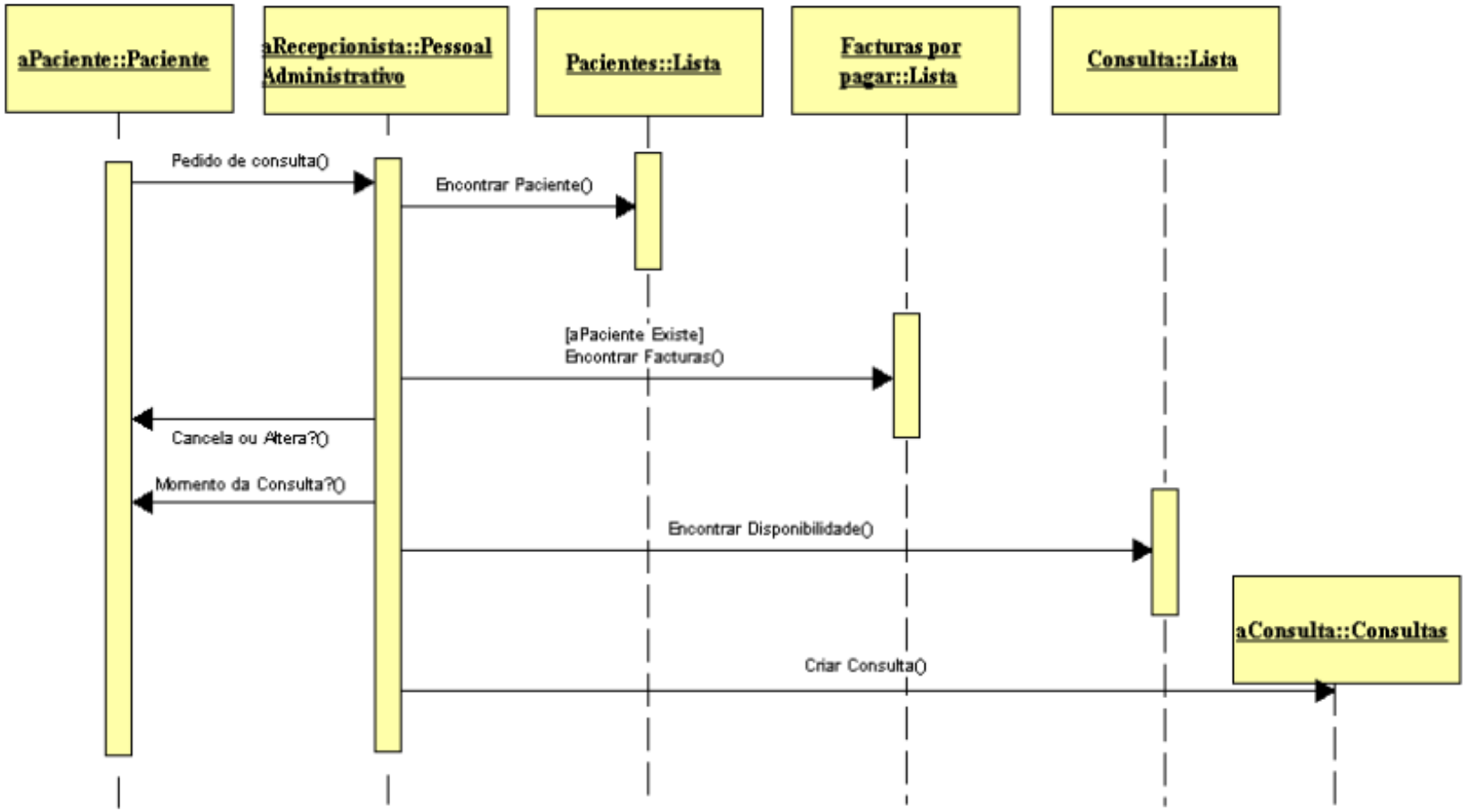
Nome da classe: Paciente	**ID:** 3	**Tipo:** Concreta, domínio
Descrição: Um indivíduo que necessita receber cuidados de saúde		**Caso de uso associado:** 2
Responsabilidades Marcar consulta Determinar última consulta Mudar estado Fornecer histórico clínico		**Colaboradores** Consulta História clínica
Atributos: Valor (duplo) Seguro (texto)		
Relacionamentos: **Generalização (um tipo de):** Pessoa **Agregação (é parte de):** História clínica **Outras Associações:** Consulta		


Pessoa
- nome
- morada
- telefone
- data-nascimento
- /idade
História Clinica
- problemas cardiacos
- tensão arterial
- diabetes
- alergias
0..1
fornece
1
seguros
*
Empregado
Paciente
- valor
- seguros
+ determinar última visita()
+ fornecer histórico clinico()
+ marcar consulta()
+ mudar estado()
1
marca
1
Factura
- data
- valor
- propósito
+ gerar cancelamento de taxa()
*
leva a
1
*
Consulta
- hora
- data
- razão
+ cancelamento sem aviso()
*
sofre
tem agendado
*
1..*
Enfermeiro
Pessoal Administrativo
Médico
Sintoma
- nome
Problema
- descição
*
*
1..*
*
0..1
*
1..*
Tratamento
- medicação
- instruções
- reacções
Equipa de Saúde

Análise OO::Modelo Comportamental

- Descreve os aspectos da dinâmica interna de um sistema de informação
  - Diagramas de seqüência;
  - Diagramas de comunicação;
  - Diagramas de transição de estados.

Comportamental::Diagrama de seqüência



Comportamental::Diagrama de comunicação

- Os diagramas de comunicação são equivalentes aos diagramas de seqüência, mas enfatizam o fluxo de mensagens ao longo de um conjunto de objetos, enquanto que os diagramas de seqüência focam-se na ordem temporal das mensagens que são passadas.

Comportamental::Diagrama de transição de estados

Projeto::Lógico x Físico

- Projeto lógico (independente de implementação) é executado para produzir um projeto que poderia ser implementado em diferentes plataformas (hardware, linguagem de programação, SGBD)
- Projeto físico (implementação específica) é executado para produzir um projeto que é específico para a plataforma escolhida.
- Algumas vezes se a plataforma é conhecida quando começa o projeto, não haverá o estágio de projeto lógico

Projeto::Compromissos

- É impossível alcançar todos os objetivos - Compromissos devem ser feitos
- Performance X Custo
- Portabilidade X Padrões X Investimento já realizado
- Usabilidade X Custo e Performance

Como fazer um bom projeto?

- Funcional;
- Eficiente ;
- Flexível ;
- Portátil;
- Seguro;
- Confiável;
- Econômico;
- Genérico;
- Possível de ser construído;
- Gerenciável;
- Fácil de Manter;
- Reutilizável;
- Útil.

O que é um padrão?

- Maneira testada ou documentada de alcançar um objetivo qualquer
  - Padrões são comuns em várias áreas da engenharia

- *Design Patterns*, ou Padrões de Projeto
  - Padrões para alcançar objetivos na engenharia de software usando classes e métodos em linguagens orientadas a objeto.

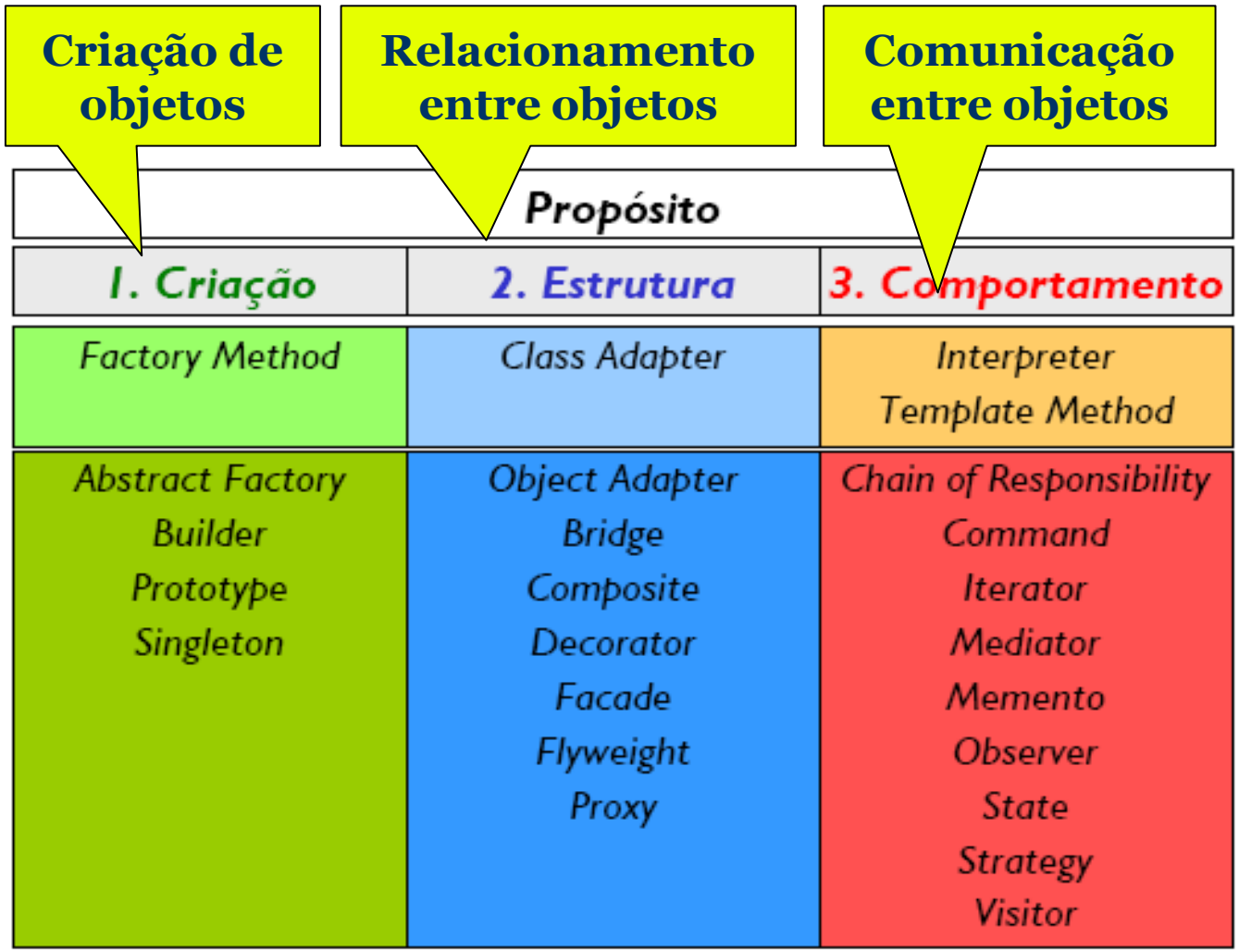
O que é um padrão de projeto?

- Padrões são um repertório de soluções e princípios que ajudam os desenvolvedores a criar software e que são codificados em um formato estruturado consistindo de:
  - **Nome**
  - **Problema que soluciona**
  - **Solução do problema**

Padrões::Classificação

- Padrões de criação
  - Tratam do processo de criação de objetos
- Padrões estruturais
  - Tratam da composição de classes ou objetos.
- Padrões comportamentais
  - Caracterizam interações e distribuição de responsabilidade entre classes e objetos.

Padrões::Classificação



Padrões::Na Prática::Façade

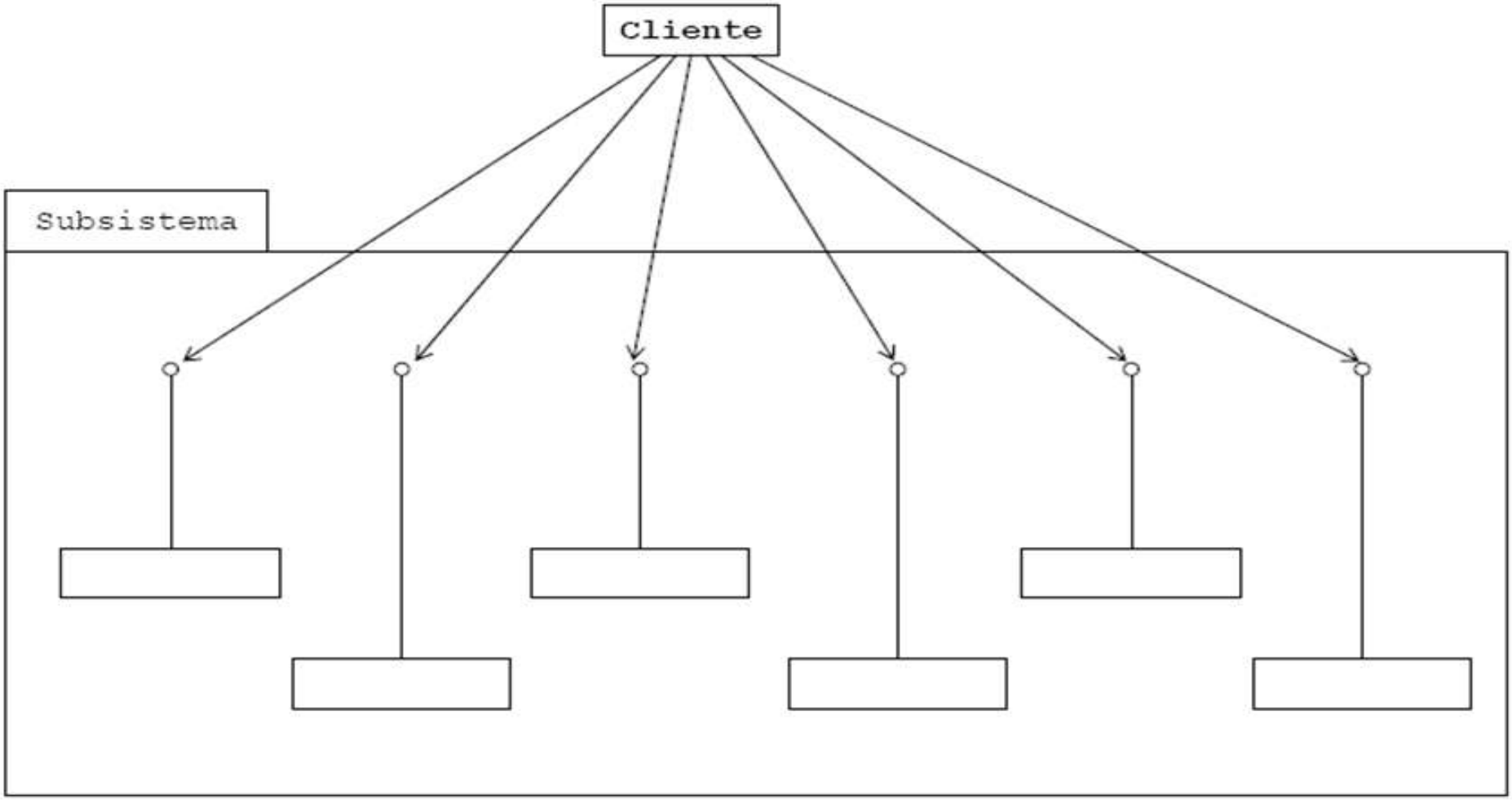
- Problema: Comunicação e a dependência entre os subsistemas forçando o cliente a conhecer muitos detalhes para poder utilizá-lo.

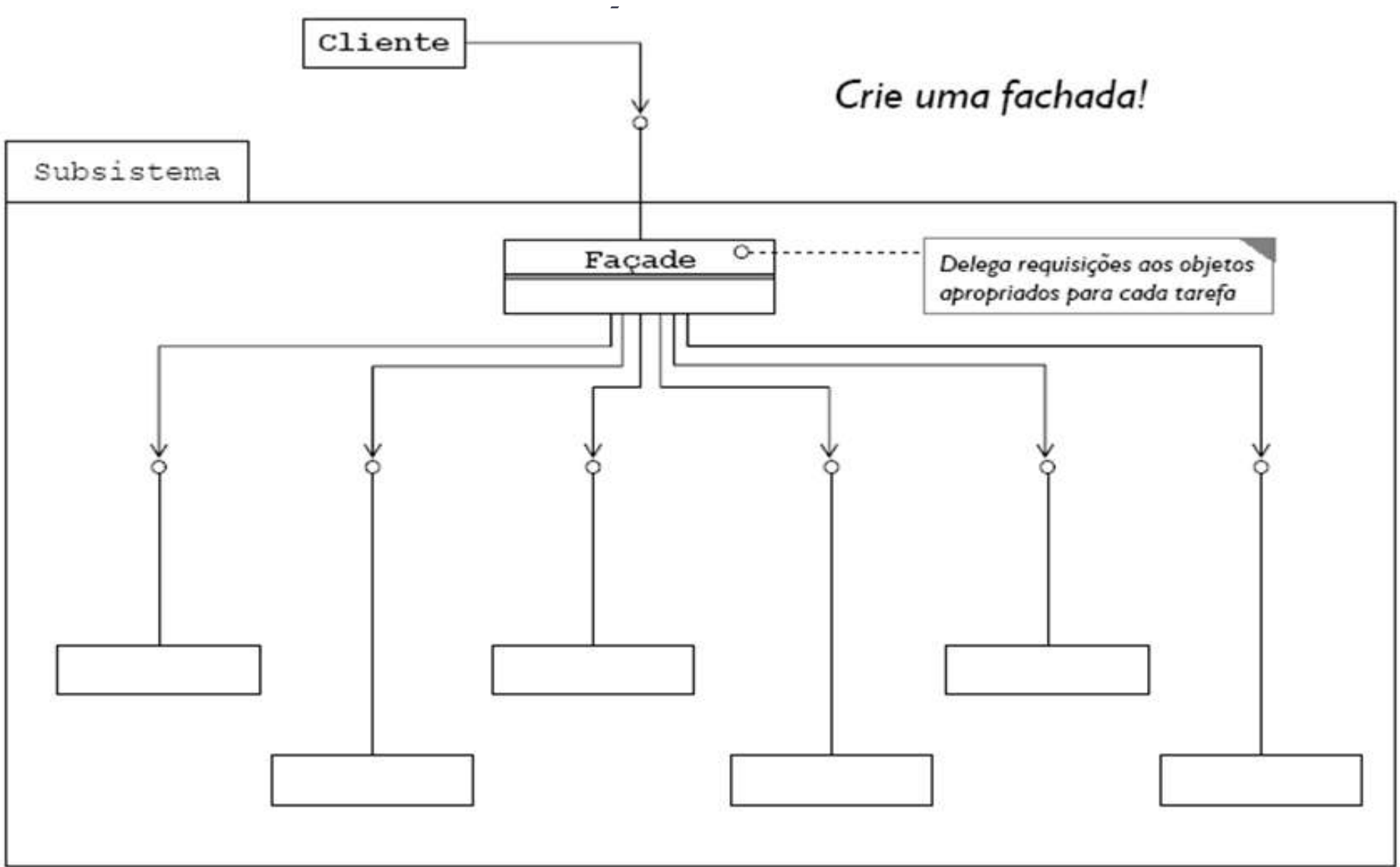
- Aplicações: Quando necessário definir um ponto de entrada para cada nível de subsistema.

- Objetivo: Fornecer uma interface unificada para um conjunto de interfaces em um subsistema. Diminuir o acoplamento entre as classes.

Padrões::Na Prática::Façade

Problema: Cliente precisa conhecer muitos detalhes




Cliente
Crie uma fachada!
Subsistema
Façade
Delega requisições aos objetos apropriados para cada tarefa

```
class Aplicação {
    ...
    Facade f;
    // Obtem instancia f
    f.registrar("Zé", 123);
    f.comprar(223, 123);
    f.comprar(342, 123);
    f.fecharCompra(123);
    ...
}
```

```
public class Facade {
   BancoDeDados banco = Sistema.obterBanco();
   public void registrar(String nome, int id) {
      Cliente c = Cliente.create(nome, id);
      Carrinho c = Carrinho.create();
      c.adicionarCarrinho();
   }
   public void comprar(int prodID, int clienteID) {
      Cliente c = banco.selectCliente(cliente ID);
      Produto p = banco.selectProduto(prodID) {
      c.getCarrinho().adicionar(p);
    }
    public void fecharCompra(int clienteID) {
       Cliente c = banco.selectCliente(clienteID);
       double valor = c.getCarrinho.getTotal();
       banco.processarPagamento(c, valor);
    }
}
```

```
public class Carrinho {
  static Carrinho create() {...}
  void adicionar(Produto p) {...}
  double getTotal() {...}
}
```

```
public class Produto {
  static Produto create(String nome,
                        int id,double preco) {...}
  double getPreco() {...}
}
```

```
public class Cliente {
 static Cliente create(String nome,
                       int id) {...}
 void adicionarCarrinho(Carrinho c) {...}
 Carrinho getCarrinho() {...}
}
```

```
public class BancoDeDados {
 Cliente selectCliente(int id) {...}
 Produto selectProduto(int id) {...}
 void processarPagamento() {...}
}
```

Padrões::Na Prática::Singleton

- Problema: Garantir que apenas um objeto exista, independente do número de requisições que se receba para criá-lo.

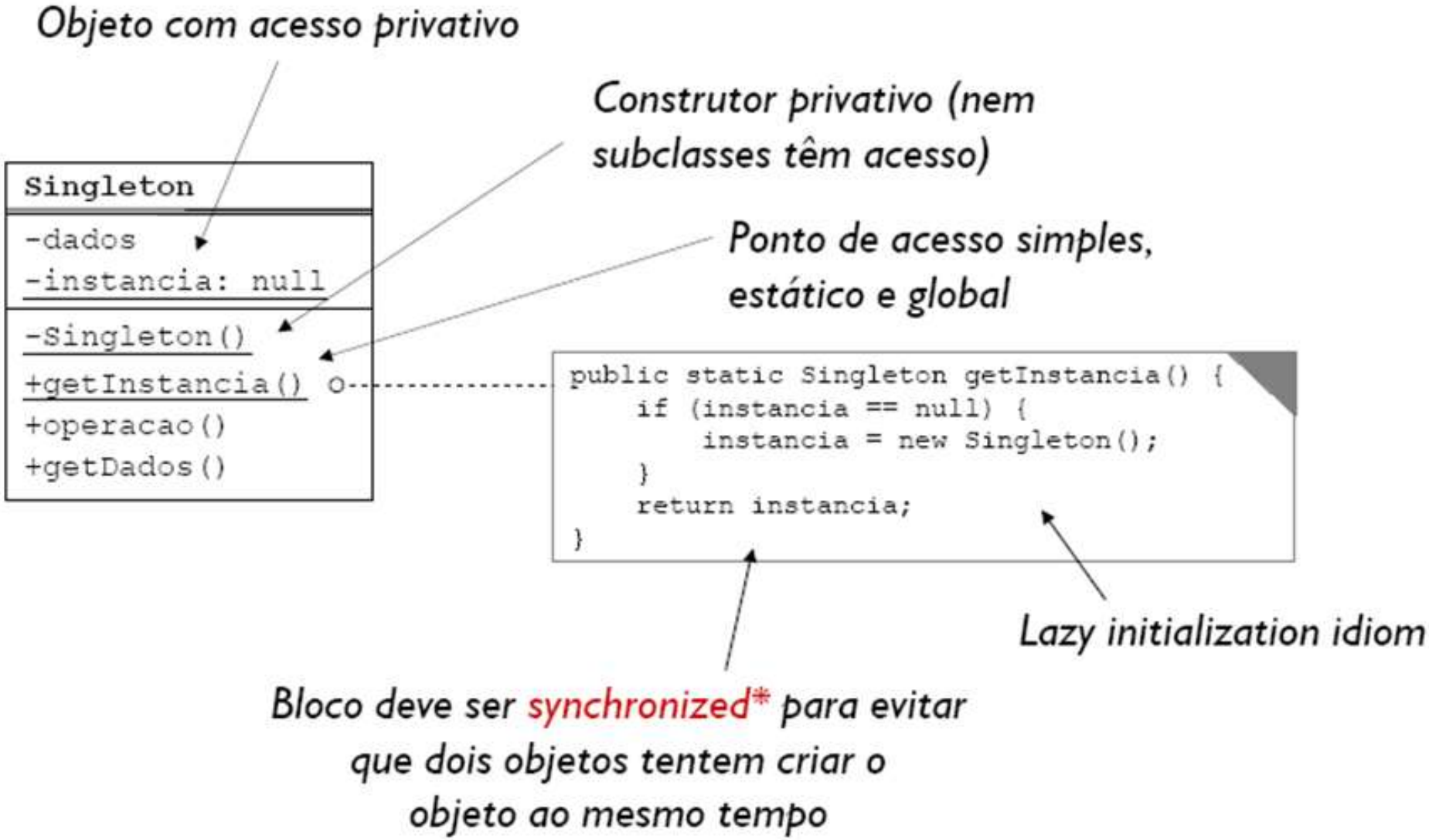
- Aplicações:
  - Uma única conexão ao banco de dados;
  - Um único acesso ao arquivo de logs.

- Objetivo: Garantir que uma classe tenha apenas uma única instância

Padrões::Na Prática::Singleton

- O padrão de criação de objetos *Singleton* garante que para uma dada classe possa haver somente uma instância. A classe *Singleton* deve:
  - Armazenar a única instância existente;
  - Garantir que apenas uma instância será criada;
  - Prover acesso a tal instância.

Padrões: Na Prática: Singleton



Padrões::Na Prática::Singleton

```
public class Highlander {
  private Highlander() {}
  private static Highlander instancia = new Highlander();
  public static synchronized Highlander obterInstancia() {
        return instancia;
  }
}
```

Esta classe implementa o design pattern Singleton

Esta classe cria apenas um objeto Highlander

```
public class Fabrica {
  public static void main(String[] args) {
    Highlander h1, h2, h3;
    //h1 = new Highlander(); // nao compila!
    h2 = Highlander.obterInstancia();
    h3 = Highlander.obterInstancia();
    if (h2 == h3) {
       System.out.println("h2 e h3 são mesmo objeto!");
    }
  }
}
```

Obrigado!

Primeia Atividade

- Pesquisar sobre os padrões de projetos e apresenta-los em seminário/workshop para a turma.
  - Em:
  - Válido para avaliação da Primeira VA

Análise e Projeto de Sistemas de Informação

Andrêza Leite
andreza.lba@gmail.com