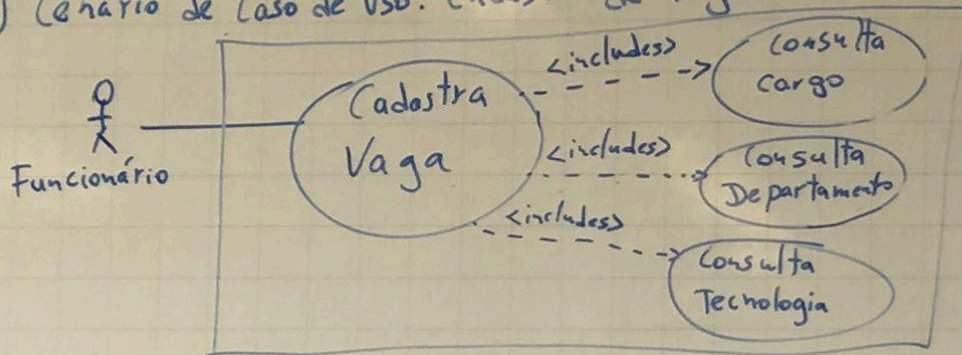
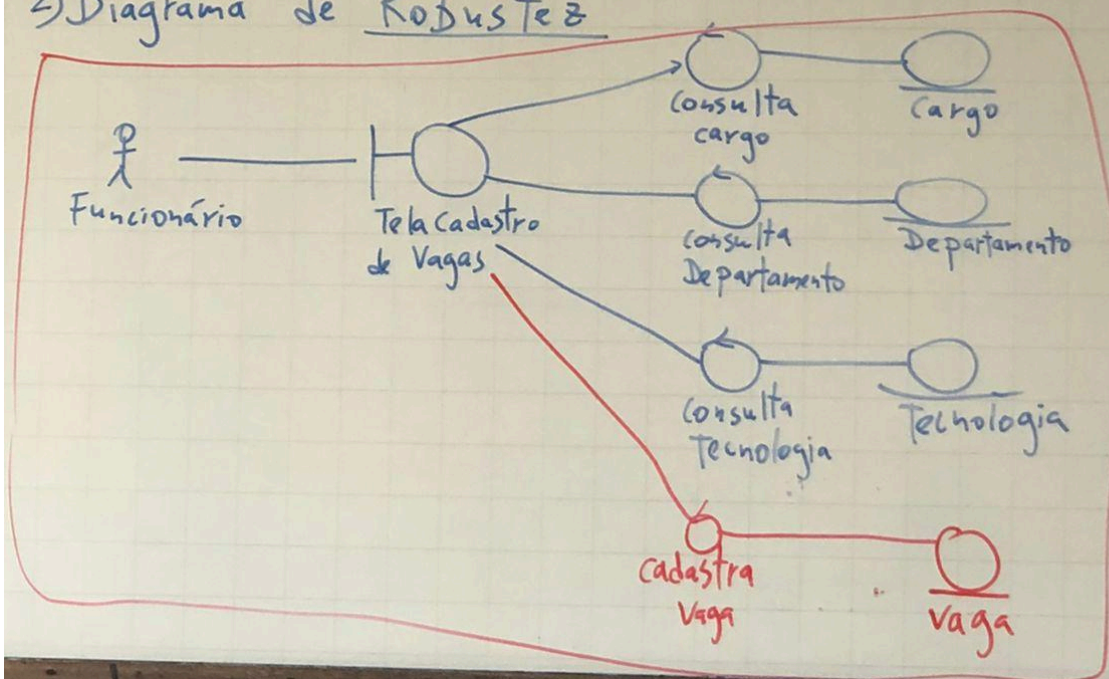


## Resposta Exercício 1:

### 1) Cenário de Caso de Uso: Cadastro de Vagas



### 2) Diagrama de Robustez



\* Exemplo de questão aberta:

funcionário  
+ seleção  
+ robustez

Seja o seguinte protótipo de um sistema de Seleção de Candidatos de uma empresa:

classes  
de de  
laboração

end).

note

**Cadastro de Vagas** O vaga

Cargo: O cargo

Analista  
Designer  
Programador  
...

Departamento: O Departamento

Marketing  
Vendas  
Produção  
...

Experiência desejada:  anos

Inglês: ☐ Sim ☐ Não

Conhecimento em Tecnologias O tecnologia

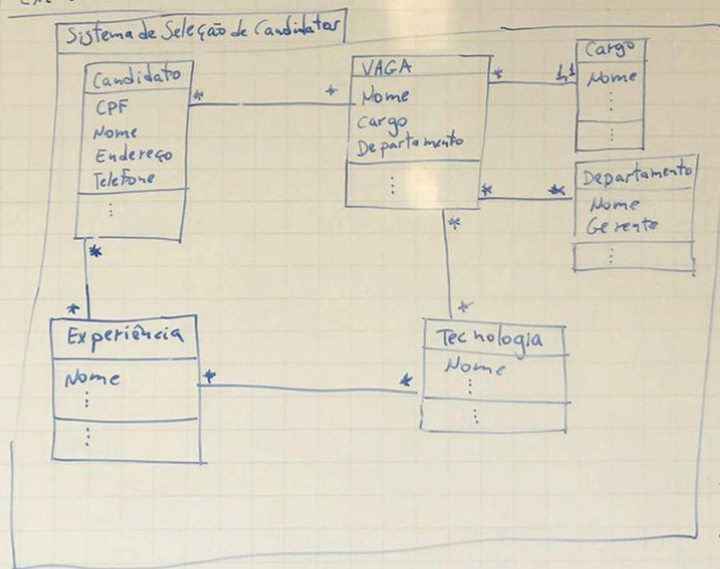
Nome	Categoria
Java	Intermediário
C#	Não
HTML/CSS	Sim
...	

Fazer:

- 1) Cenário de Caso de Uso.
- 2) Diagrama de Robustez.



### Exercício 2:



→ Sabendo-se que o diagrama de classes do Sistema de Seleção é mostrado ao lado. Faça o modelo de dados Relacionais pois o SGBD a ser usado é MySQL.

\* Chave Primária

+ Chaves Estrangeiras

# Cardinalidade → 1,1; 1,N; N,N

24/09 - Eng. Software II  
Prof. Sandro

\* Revisão para a prova:

1) Conceitos de Projeto de Software  
o que é? Metodologias?

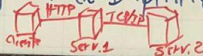
2) Projeto Orientado a Objetos (UML)

Projeto Preliminar  
Documento de Especificação

Cenários de caso de uso  
Proto-típos de telas  
Diagramas de Robustez

Projeto Detalhado

Cenários de caso de uso  
Diagrama de classes e pacotes  
Diagramas de interação de objetos  
Diagrama de Componentes  
Diagrama de Implantação



1971 - Projeto Estruturado  
Década de 80 - Projeto O.O. → UML → R.U.P.  
2000 → Projetos Baseados em Serviços

Processo Unificado

3) Projeto de A

Arquitetura

+ Demonstr

os requisitos fi

→ Arquitetura

+ Apresenta

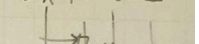
os componentes

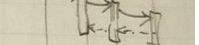
e sua implan

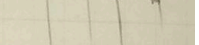
4) Uso de fi

entre: Pro

e

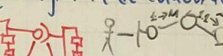






Interface  
Controles  
Entidades

- Diagrama de sequência  
- Diagrama de Colaboração



oferece Serviço  
consome Serviço



### R.U.P. 3) Projeto de Arquitetura

#### ↳ Arquitetura Lógica

\* Demonstra as Regras de Negócio, os requisitos funcionais do software.

#### → Arquitetura Física

\* Apresenta os requisitos não funcionais, os componentes tecnológicos do software e sua implantação.

Casos de Uso  
Pacotes  
Classes + Sequência  
+ Colaboração  
+ Rotinas

Componentes  
Implantação

\* Exem

Sej

de

Co

C

D

E

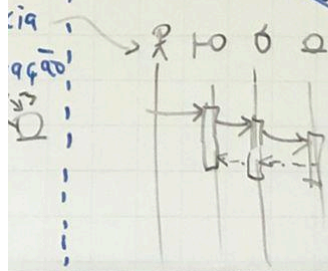
I

Co

### 4) Uso de frameworks e diferença

entre: Projeto de Interface e interação (front-end)  
e projeto de persistência (back-end)

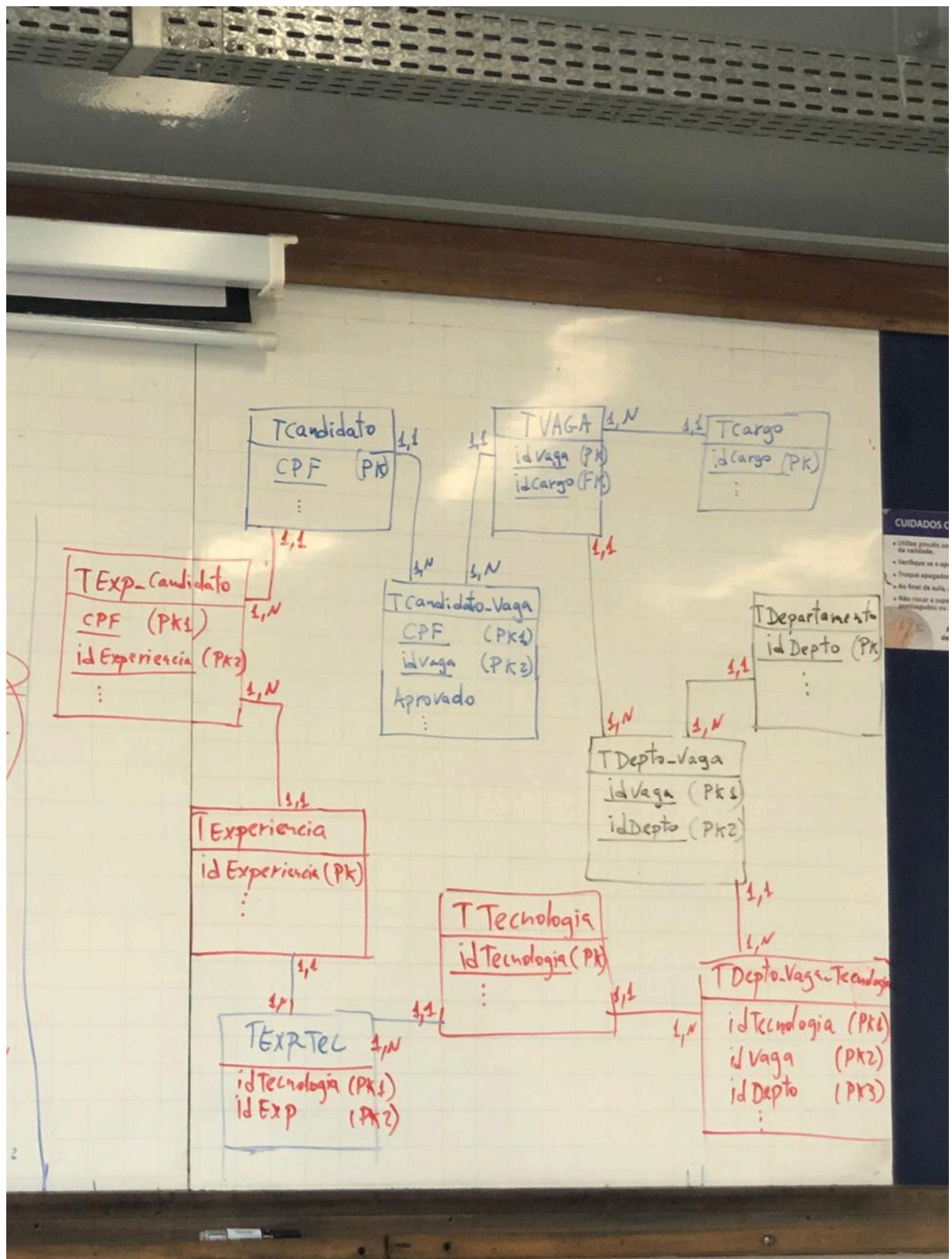
≠ Biblioteca de classes  
Modelo de Colaboração



Caixa-branca → Open  
Caixa-preta → Fechado

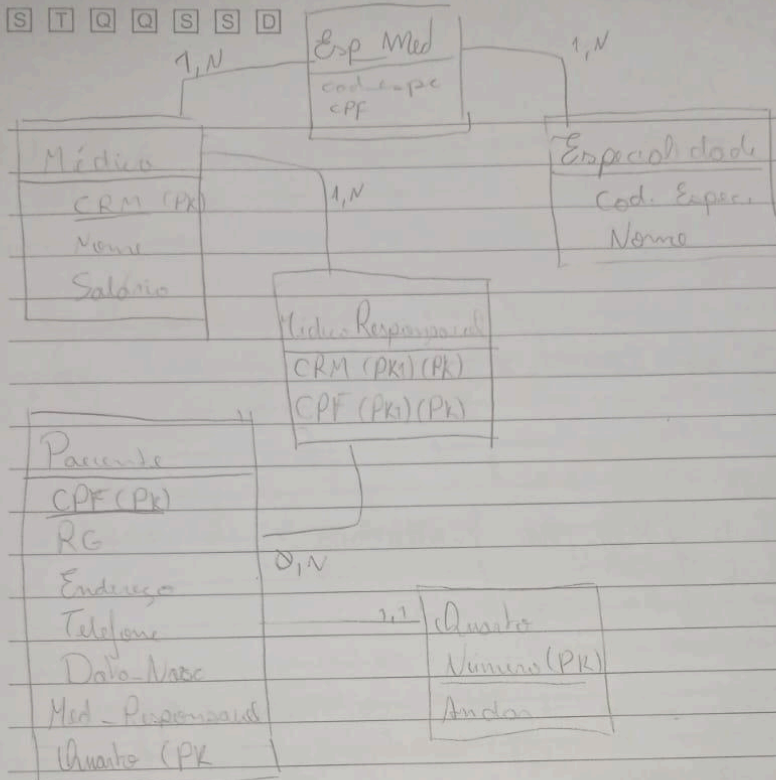
Empresarial → SAP, Totus

Front-end → React, Vue  
Backend: Struts, Hibernate





S T Q Q S S D



Aula 07. quinto. 29/02

Projeto de Persistência

↳ MOR - Mapeamento Objeto Relacional

↳ exemplos no canvas

• JPA - Java Persistence API

• Hibernate

\* Resumo até hoje

✓ Proj Preliminar

✓ Proj Detalhado

Projeto → Proj. de Software → Proj. OO.

↳ Proj. de Interface

↳ Proj. de Persistência

↳ MOR

↳ SGBD Relacional

↳ SGBD MySQL

↳ Arquivos

spiral

## Aula 03 quinta 15/02

Projeto Detalhado (Segundo o processo unificado - RUP)

Projeto preliminar { cenários para cada caso  
de uso primários  
+ Diagrama de Robustez  
Protótipos

• Visão de objetos: diagrama de interações de objetos

↳ Diagrama de colaboração/comunicação

↳ Para cada diagrama de Robustez, inserimos as interações entre objetos

↳ Diagrama de sequência

↳ Para cada cenário, modelamos a sequência de eventos com as interações de objetos

• Visão de implementação

↳ Para cada cenário, identificamos os componentes internos e externos relacionados (Diagrama de Componentes)

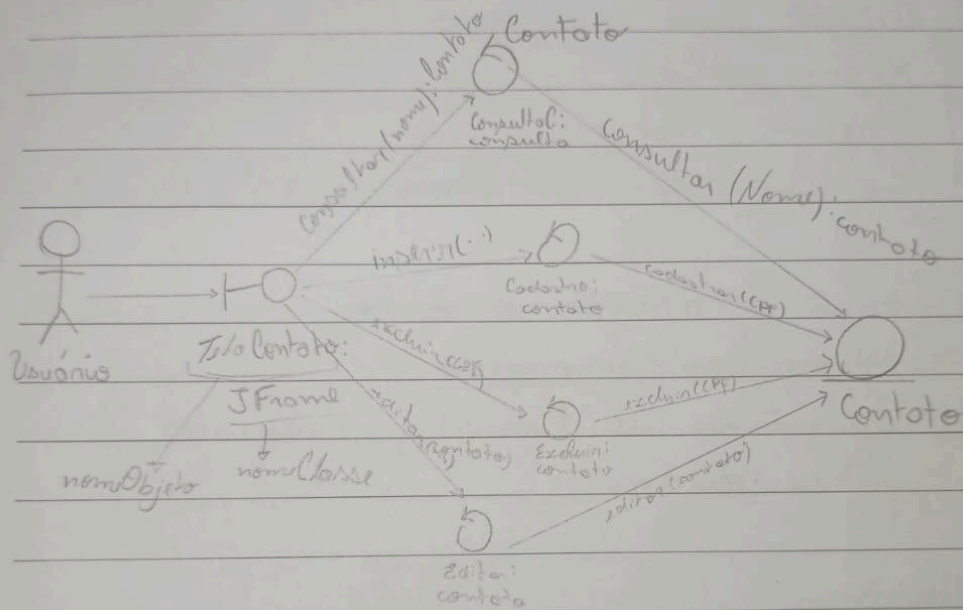
↳ Para cada cenário identificamos os processadores envolvidos e como estão relacionados (Diagrama de execução)



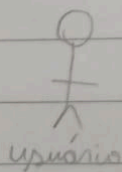
## Exemplos de diagramas: Agenda

### 1) Diagrama de colaboração

cenário: cadastro de contato



### 2) Diagrama de Sequência



1/1 3º pº 25 pº  
2º pº 30 pº

S T Q Q S S D

Aula 10. terça. 12/05

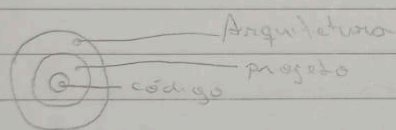
BDD

Cucumber - site

→ Arquitetura de Sistema

- Performance
  - Confiabilidade
  - Escalabilidade
  - Estabilidade
  - Manutenção
- } Req. não funcionais \*

~ Programar e Documentar é essencial para melhor progressão



Desempenho x Manutenção

• Representação gráfica

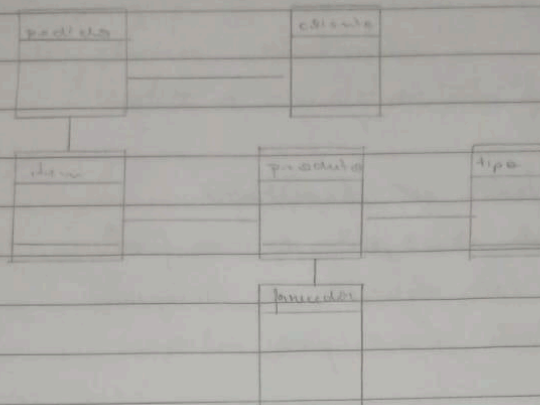
apresenta - apenas em texto  
com figuras ilustrativas

~ Tipos de Arquitetura

- Fluxo de dados (pipes)
- Sist. distribuído [com variação]



## Diagrama de Classes ②



## Protótipo ④

Pedido

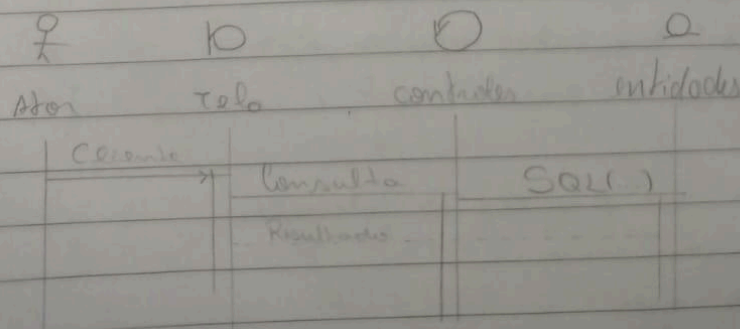
cliente

### Item

			(+)	(X)	(A)
			(-)	(X)	(P)

## Integração de Objetos

### ⑤ Sequência e Colaborações



Aula 11. quinta. 14/03  
(note)

Aula 12 quinta. 21/03

\* Revisão para 1ª Prova

\* Que Condições (Ponto extra)

- Conceitos de Projeto e histórico
  - Projeto de Interface
  - Diagrama de Requisitos
- Projeto OO → Projeto Preliminar
  - Projeto Detalhado
    - Interação de objetos → Diag Sequência
    - Diagrama de Pacotes e Componentes → Diag Colaboração
    - Diagrama de Execução/Deploy
- Projeto de Interface e Interação
  - Foco para design System
- Projeto de Persistência
  - Foco para ORM
    - Mapeamento Objeto Relacional
- Projeto de Arquitetura
  - Arquitetura Lógica → Example Work
  - Arquitetura Física

→ Exercícios: Distribuído (passando pelo ambiente)

Stack atual: Delphi (objetos parciais)

⑧

SGBD PostgreSQL

Stack proposta: Framework Swift (JavaScript) + Servidor + Arquivos (baseado em JS)

Back-END: SGBD PostgreSQL com atualização das tabelas + API C# + Sequência

Execução: Janela sobre a Web

spirati



## Exercício de Revisão

### Projeto detalhado

#### ↳ protótipos:

- Não são descartados após demonstrações
- pode ser aproveitado como parte do cod. final
- mesmo sem captar todos requisitos de desempenho e de eficiência, ainda ótimo para validação do projeto
- Desenvolvimento rápido e iterativo
- demonstra conceitos, testa opções de projeto e busca problemas e suas possíveis soluções

### → Diagramas

#### ↳ Baixa complexidade

combinado  
para melhor  
resultados

- atribuição de responsabilidade de suporte
  - Menor dependências entre as classes
  - Alteração em uma classe com menor impacto possível as outras
  - Maior potencial de Reutilização

#### ↳ Alta coesão

- manter objetos focados, gerenciáveis e compreensíveis
  - responsabilidade de um determinado elemento está fortemente relacionados
- altamente focados

### → Critérios Recomendáveis no projeto de interação

- tempo consumido com erros
- frequência de uso de ajuda
- tempo para realizar uma tarefa

S T Q Q S S D

→ Reutilização de código  
Padrões de design  
Flexibilidade  
Produtividade  
Manutenção

→ Framework

↳ projeto de subsistema composto por uma conjunção de classes abstratas

Estabelece a arquitetura para aplicações em um domínio

→ Documentação/  
comunicação, bib, Segun  
es, cab, classe

Para O.O

projeto preliminar → projeto detalhado → projeto de arquitetura (física e lógica)

Prototipagem

• demonstra conceitos e funções antes de buscar problemas e suas soluções

• desenvolvimento rápido e iterativo (curto ciclo, mais feedback)

• entrega de protótipo de qualidade oferece melhor produto final

• não captura requisitos relacionados ao desempenho e eficiência, ainda assim oferece um bom emprego na elaboração e na validação das req. do sistema

Design Visual

↳ a função do site e a informação devem ser sublinhadas antes o desenho.

Design System

↳ projeto de interface e interação padronizada

↳ web e mobile usam o mesmo padrão para apresentar produto

↳ facilita criação e manutenção do projeto

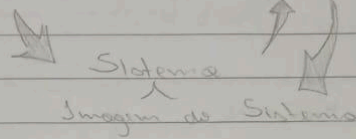


aula 04. terça. 20/02

Projeto de Interface e Interação

Modelo Design  
 Designer

Modelo do Usuário  
 Usuário



•  $U_x$   
 $T_{USER}$   
 ↳ experience

★ Interface ★  
 Forma  
 Estrutura  
 Função

Design ↑ valor ↑

Design

- ↳ Conformidade → inovação de maior produtividade
- ↳ Anticulação → transformação do produto (evolução do celular)
- ↳ Referência →

Design Atômico

"Você deve ter uma boa  
 razão para quebrar o padrão"

↳ partir dos  
 menores (objetos, componentes)  
 para o produto inteiro final

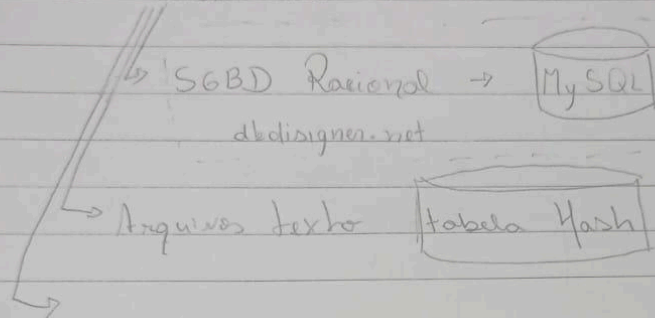
Identificar → Selecionar → ...

→ Integração Cartográfica

SIG

Aula 06. terça. 27/02

Projeto de Persistência (Back-end)



\* Objetos Transientes x Persistentes

↳ somente na memória

principais

Framework → ORM — BR — MOR, Mapeamento Objeto Relacional

MO x MR <sup>relacional</sup>

• Tabelas: ex: Clínica Médica

• Médico

• Paciente

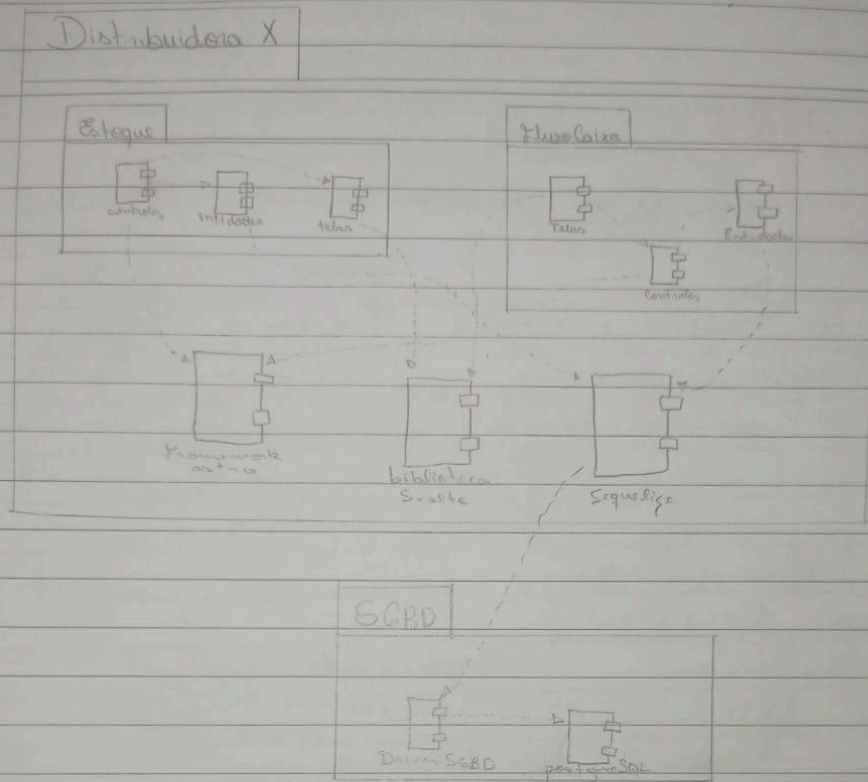
• Especialidade

• Médico Responsável

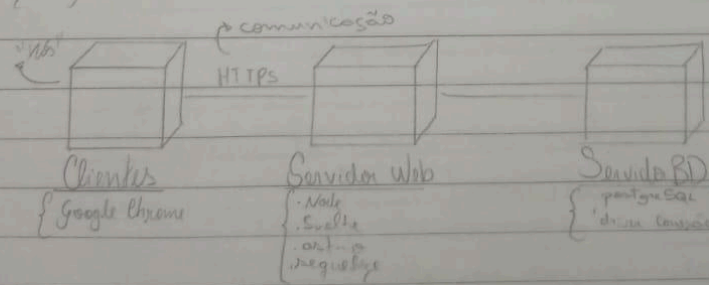
• Quarta



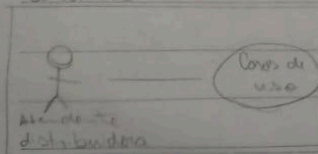
## Diagrama de pacotes e componentes



## Diagrama de Execução



## Cenário



## Diagrama de Robustez

