Infnet
Projeto de Bloco

Projeto de Bloco

Engenharia Disciplinada de Softwares

Graduação em Engenharia de Software - 2020

Tarefas Solicitadas na Aula Passada - 30/07/2020

1. Elaborar o TP1 a ser entregue no dia 03/08/2020.
    - Mande um rascunho do TP1 para o professor dar uma olhada antes de entregar → `armenio.cardoso@prof.infnet.edu.br`
2. Estudar o capítulo 2 do livro-texto do Projeto de Bloco.

Etapa 2 Aula 1

Disciplined Agile Delivery - Inception

Inception (Concepção)

Inception (Concepção)

Montar a Equipe.

Especificação de Requisitos.

Especificação da Arquitetura.

Avaliar a Viabilidade.

Desenvolver uma Visão Compartilhada.

Configurar as Ferramentas.

Github

Infnet
Projeto de Bloco

Github

Criar uma conta no Github e criar um repositório para o VenturaHR.

Enviar o link do seu repositório de projeto para `armenio.cardoso@prof.infnet.edu.br`

Seja cuidadoso com a estrutura de diretórios e versione TODOS os artefatos do projeto.

http://blog.triadworks.com.br/aprenda-a-usar-o-github-como-seu-portfolio

https://www.udemy.com/course/git-e-github-para-iniciantes/

Mapas Mentais

O que são Mapas Mentais?

Os mapas mentais, ou “mapas da mente” é o nome dado para um tipo de diagrama criado pelo inglês Tony Buzan.

Os mapas mentais são voltados para a gestão de informações e de conhecimentos.

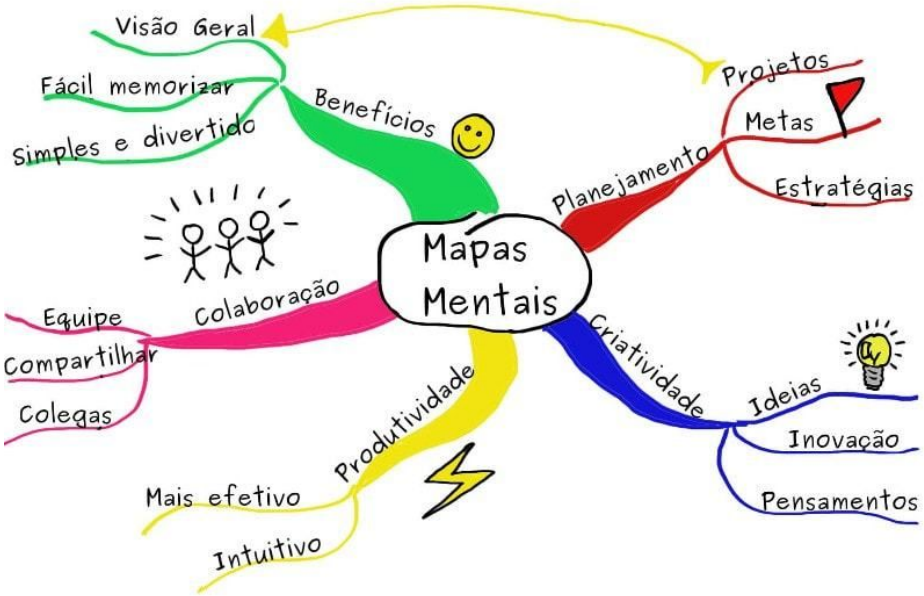
São representações livres de pensamentos que se dividem a partir de um conceito central, para compreensão e elaboração de soluções de problemas, melhorando a memorização e o aprendizado.

O que são Mapas Mentais?

Os mapas mentais podem variar de simples a mais complexos, podendo ser desenhados à mão ou no computador, incluindo fotos, desenhos, linhas curvas de espessuras variáveis e diversas cores.

Mapas mentais são essenciais na hora do estudo, pois reduz, simplifica e seleciona as informações que são mais relevantes ao que está sendo estudado, ajudando nosso cérebro a fazer novas associações velozmente.

O que são Mapas Mentais?

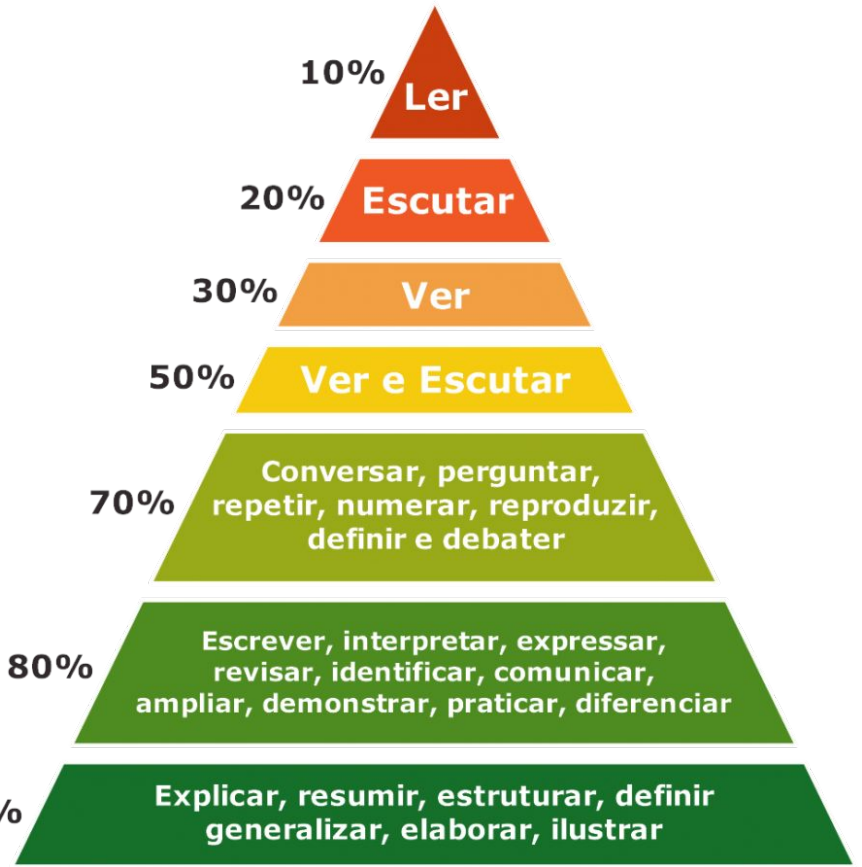


Pirâmide de Aprendizado de William Glasser

Aprendizado Passivo

Aprendizado Ativo

MM



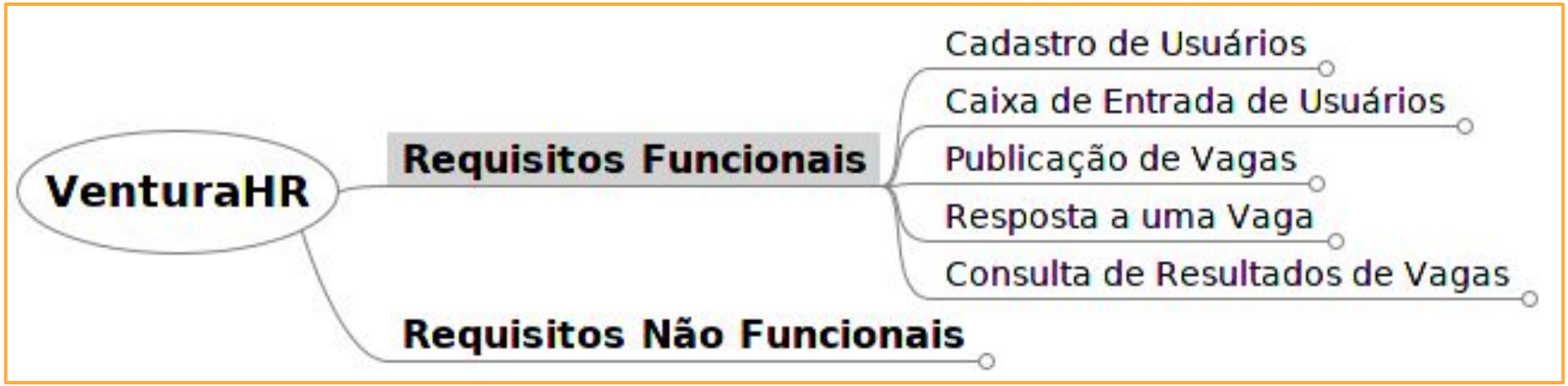
Especificação de Requisitos

Detalhamento dos Requisitos

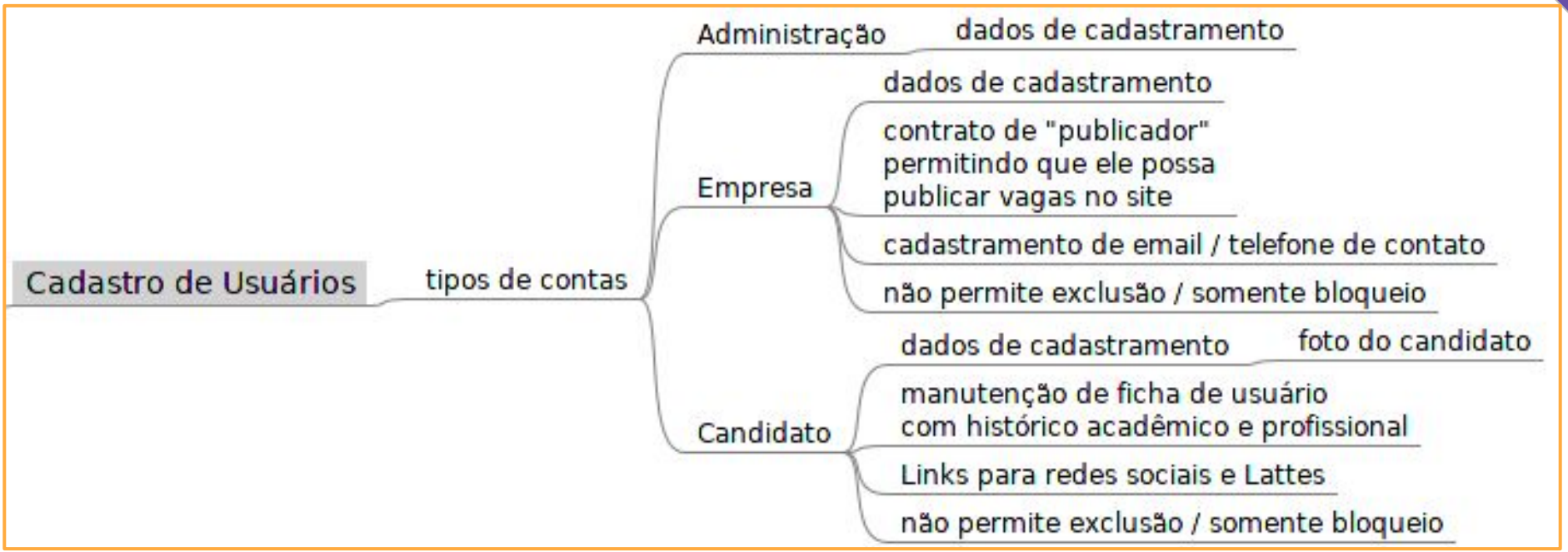
Depois de elaborar o Documento de Visão como uma especificação sumária do VenturaHR, a equipe de desenvolvimento fez uma reunião com o dono da VenturaSoft a fim de detalhar os requisitos do sistema.

O resultado desse detalhamento foi um Mapa Mental com várias informações relevantes e o detalhamento dos requisitos.

Requisitos Funcionais

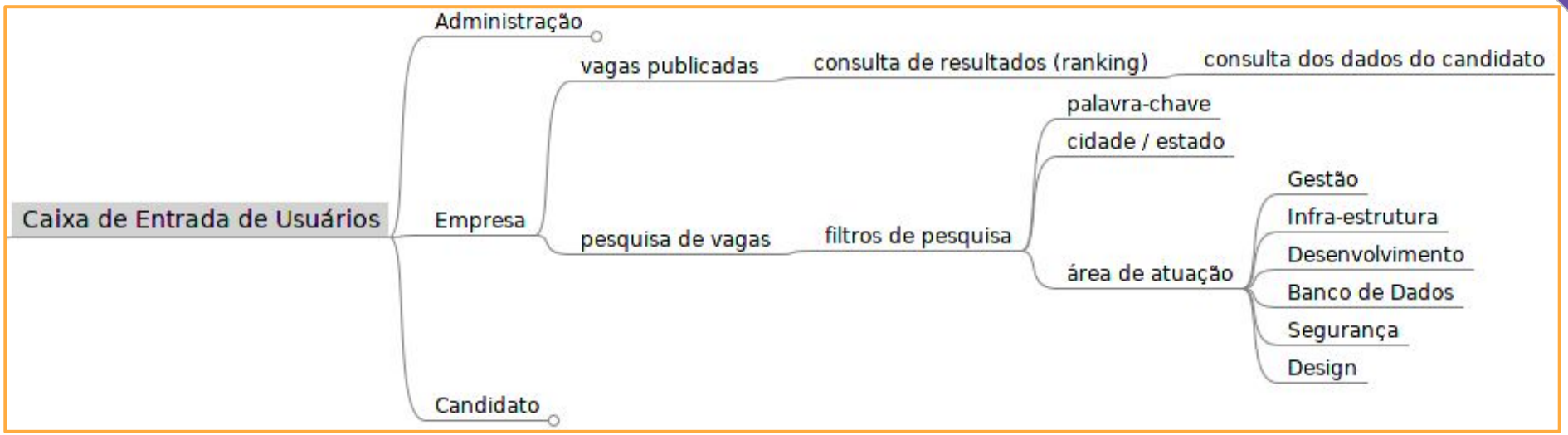


Cadastro de Usuários

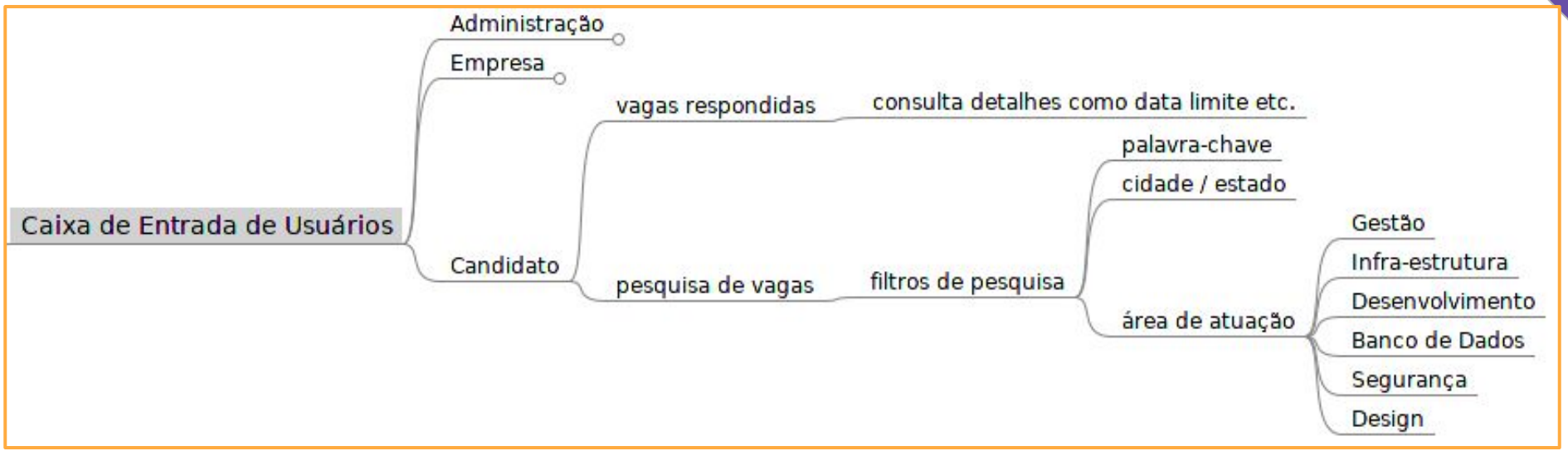


Caixa de Entrada de Usuários - Administração

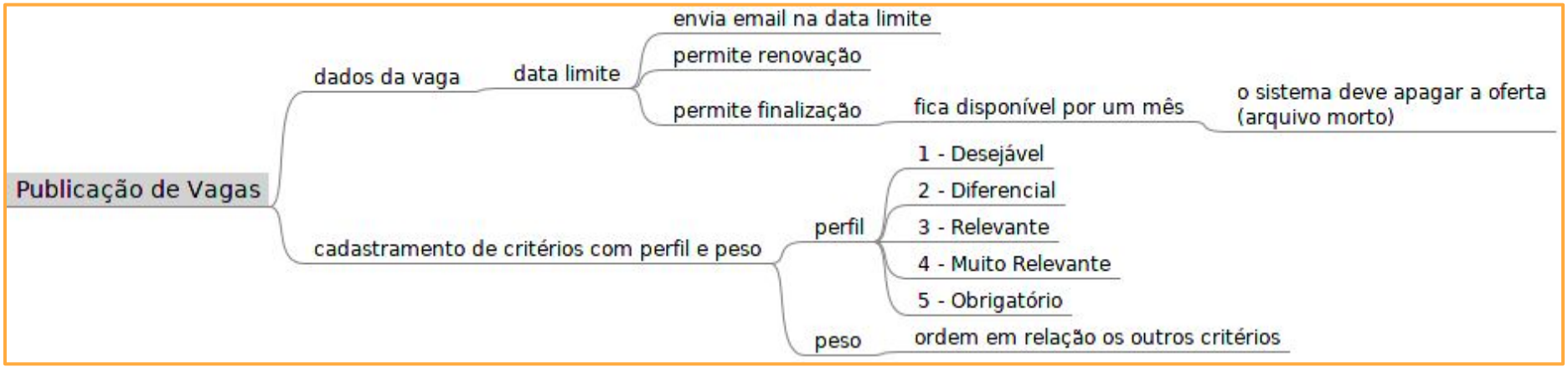
Caixa de Entrada de Usuários - Empresa



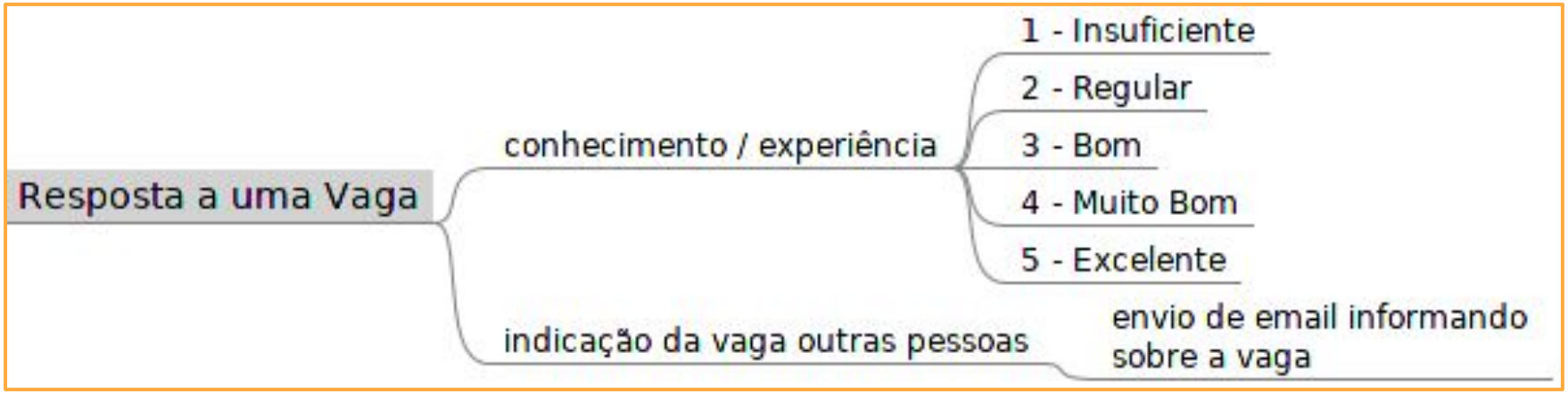
Caixa de Entrada de Usuários - Candidato



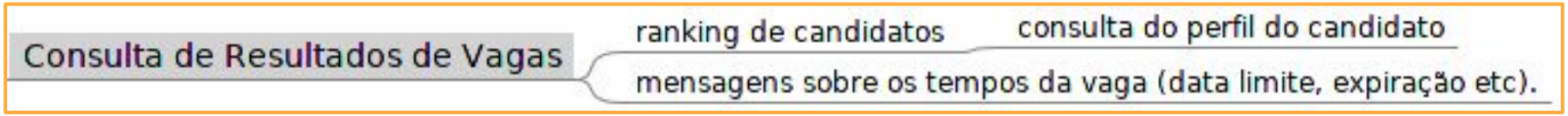
Publicação de Vagas



Resposta a uma Vaga



Consulta de Resultados de Vagas



Diagramas de Classes

O Que é Modelagem?

- “Construímos modelos de sistemas complexos porque não é possível compreendê-los em sua totalidade.”

- Objetivos:
  - Visualizar Sistemas em Detalhes.
  - Comunicar e Documentar Decisões.
  - Especificar a estrutura e o comportamento do sistema.

Atenção

Muitos acham que modelar é produzir a documentação do sistema.

A documentação de um sistema é só mais um produto da modelagem.

Não é o objetivo principal.

Conceito → Responsabilidades

É o que a classe **sabe** e o que ela **faz**.

O que a classe **sabe** são as propriedades ou seus atributos.

O que a classe **faz** são os seus métodos ou funções.

Classes de Um Sistema

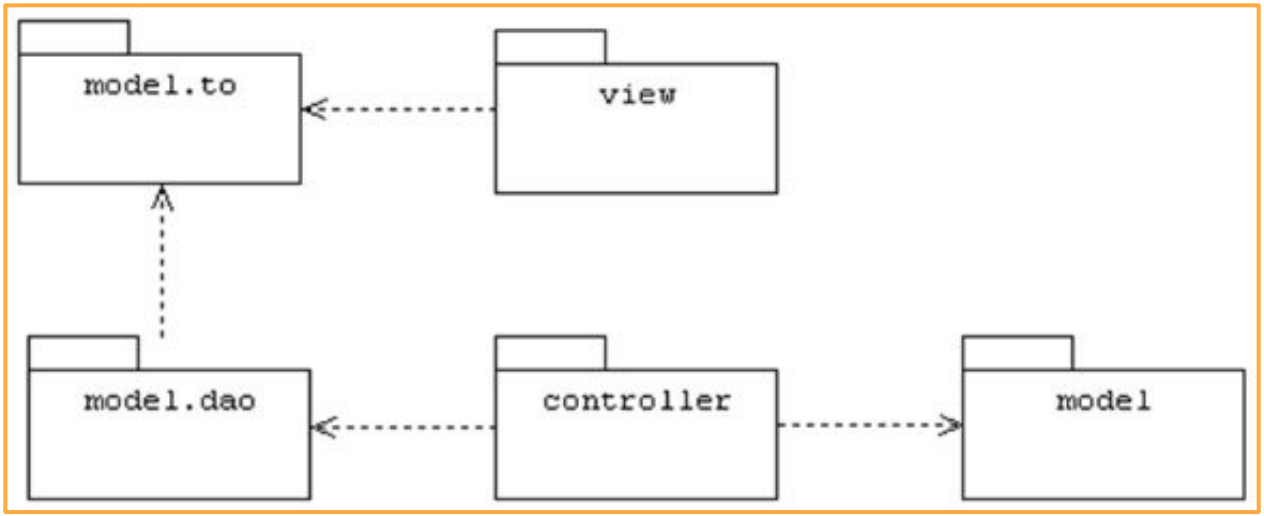
- **Classes de Fronteira** - interfaces com o usuário e outros sistemas (Botões, *Checkboxes*, Telas).
- **Classes de Entidade** – estruturas de dados representando o domínio do problema (Cliente, Pedido, Item de Pedido, Produto).
- **Classes de Controle** – classes que representam processos e elementos dinâmicos (Datas e Tempo, Operações em Bancos de Dados, Gerenciador de Impressão, Leitura e Gravação de Arquivos).

Mais sabe do que faz

Mais faz do que sabe

Pacotes

- Os pacotes são agrupamentos de elementos: classes, componentes, atores etc.
- Um sistema é dividido em pacotes para melhorar o entendimento e para aumentar a produtividade da equipe de desenvolvimento.



Pacotes

- Convenções para nomes de pacotes:
  - Um pacote genérico, contendo classes que podem ser aproveitadas em vários sistemas diferentes, deve ser descrito pelo domínio da empresa em ordem reversa.
  - Um pacote que seja específico a um sistema e que não será aproveitado em outros poderá utilizar o nome do sistema como pacote.

br.com.infnet.gui

venturahr.gui

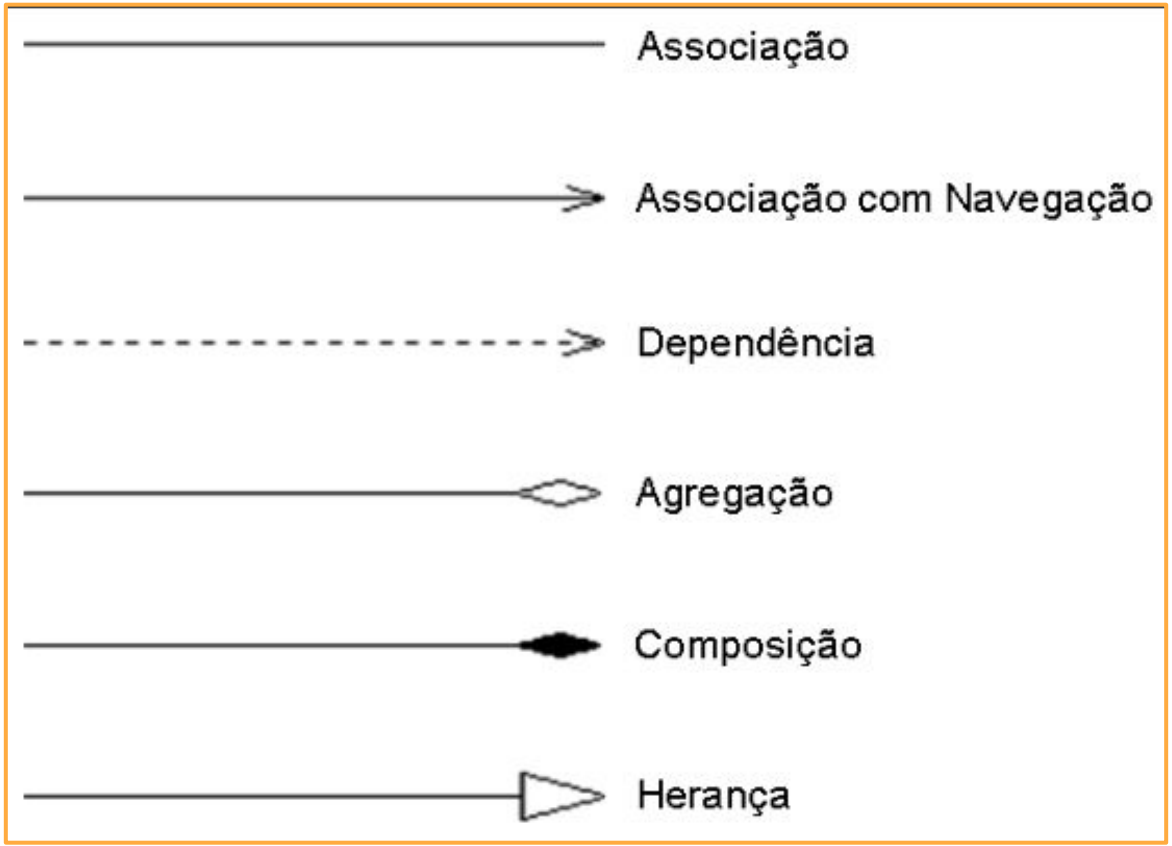
java.util

O artigo a ser estudado para a próxima aula vai ajudar na criação dos pacotes

Como Identificar Classes?

- As classes de um sistema podem ser identificadas a partir de um diagrama de casos de uso e de suas descrições.
- Liste todas as entidades (tipos complexos) que forem encontradas nas descrições.
- Verifique se, entre estas entidades, não existe alguma relação, como por exemplo:
  - Elas são sinônimas?
  - Uma contém a outra?
  - Ambas têm muitos métodos e atributos em comum?

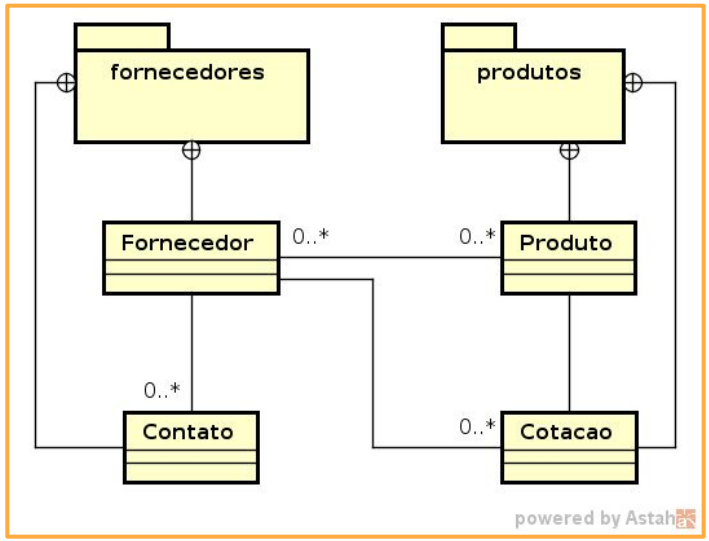
Tipos de Relacionamentos



Associação

- A associação é o relacionamento entre classes, representada por um traço simples.
- As associações podem expressar relações bidirecionais entre classes.
- Faz parte das responsabilidades de um objeto de uma das classes determinar os objetos correspondentes da outra classe.
- Uma associação é implementada com atributos. Assim, se um pedido está relacionado a um cliente, este relacionamento pode ser implementado colocando-se um atributo do tipo cliente dentro da classe pedido.

Associação



Fornecedor.java (~/Imagens/i592) - gedit

Arquivo Editar Ver Pesquisar Ferramentas Documentos Ajuda

Fornecedor.java

```
package br.com.infnet.fornecedores;

import br.com.infnet.contatos.Contato;
import br.com.infnet.produtos.Produto;
import br.com.infnet.cotacoes.Cotacao;
import java.util.Date;
import java.util.List;

public class Fornecedor {

   private Long id;
   private String nomeFantasia;
   private String razaoSocial;
   private String cnpj;
   private Endereco endereco;
   private String tipoInscricao;
   private String inscricao;
   private String observacao;
   private List<Contato> contatos;
   private List<Produto> produtos;
   private List<Cotacao> cotacoes;

   public Long getId() {
      return id;
   }

   public void setId(Long id) {
      this.id = id;
   }

   public String getNomeFantasia() {
      return nomeFantasia;
   }

   public void setNomeFantasia(String nomeFantasia) {
      this.nomeFantasia = nomeFantasia;
   }

   public String getRazaoSocial() {
```

Infnet
Projeto de Bloco

As classes mantêm atributos que representam o relacionamento

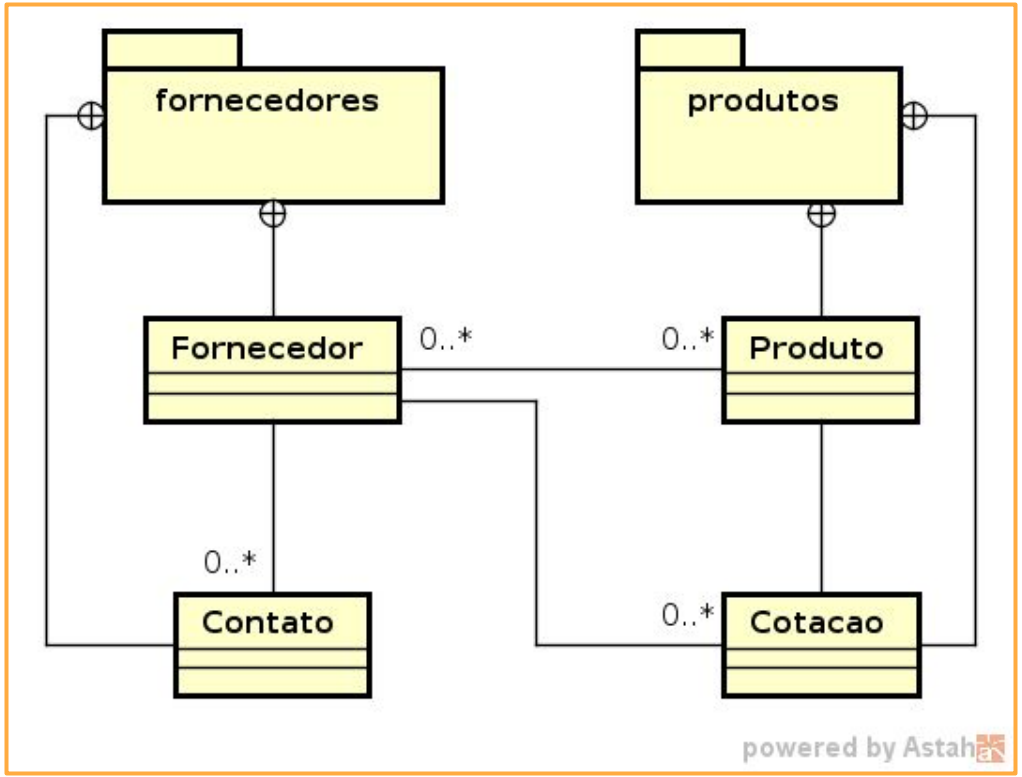
Multiplicidade

- A multiplicidade de um participante em um relacionamento indica quantos objetos de uma classe se relacionam com cada objeto da outra classe.

0..1	Zero ou um
1	Somente um
0..*	Zero ou muitos
1..*	Um ou muitos

Multiplicidade 1 é o *default* e a sua representação é opcional

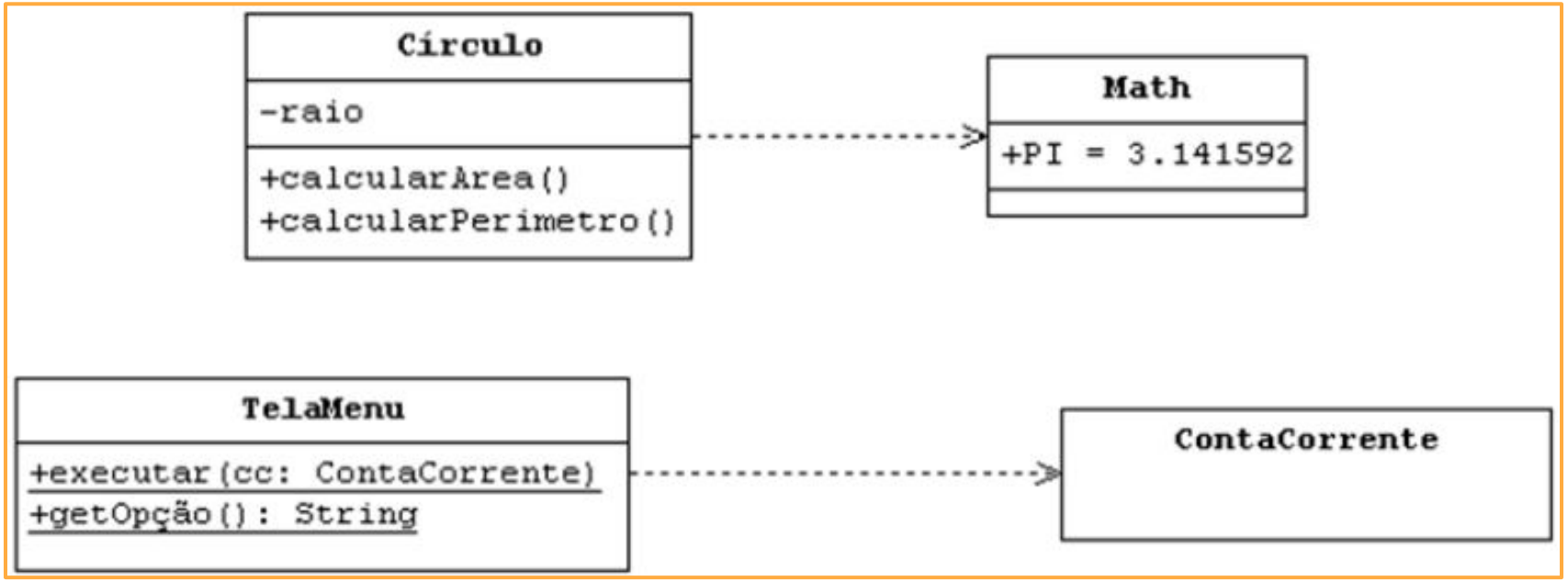
Multiplicidade



Dependência

- Relacionamento de dependência é uma relação fraca, indicando que uma classe usa outra, mas não possui nenhum atributo permanente dela.
- É representada por um traço pontilhado e direcionado.
- Como identificar dependências?
  - Quando uma classe tem uma operação que usa uma instância de outra classe como parâmetro;
  - Uma classe chama uma operação que é escopo de outra classe.
  - Uma operação retorna um objeto de outra classe

Exemplos de Dependência

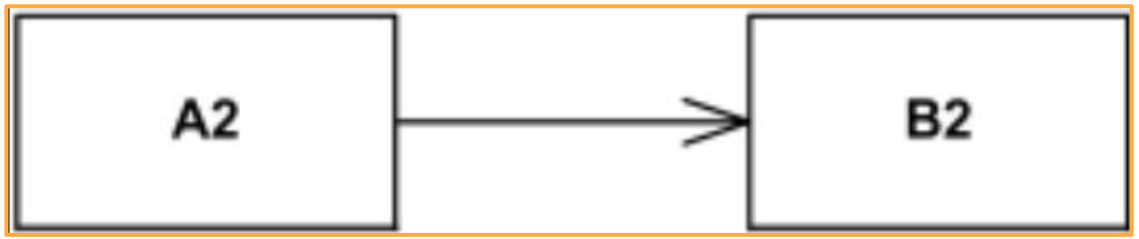


Círculo depende de *Math* para as operações de cálculo.

TelaMenu depende de ContaCorrente por causa do método executar

Navegabilidade

- Todas as classes de um relacionamento de associação “conhecem” as outras, ou seja, possuem atributos das outras classes.
- Para indicar uma direção para a associação é usada a navegabilidade ou a associação direta.

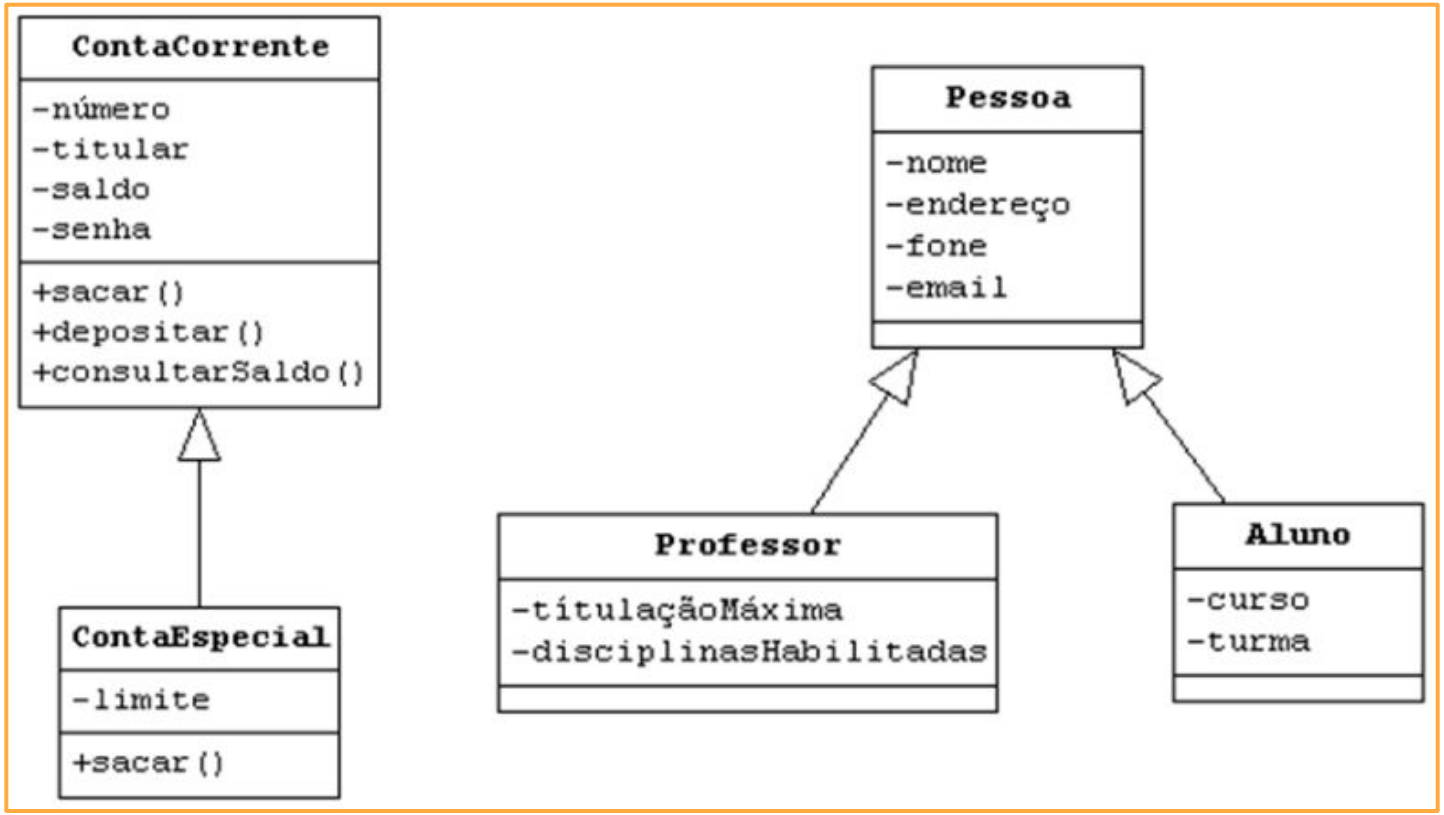


Objetos da classe A2 conhecem os objetos da classe B2, mas B2 não conhece os objetos da classe A2

Herança

- O relacionamento de herança existe entre classes de natureza mais geral (chamadas de superclasses ou classes bases) e suas especializações (chamadas de subclasses ou classes derivadas).
- As superclasses contêm atributos ou operações comuns a um grupo de subclasses.

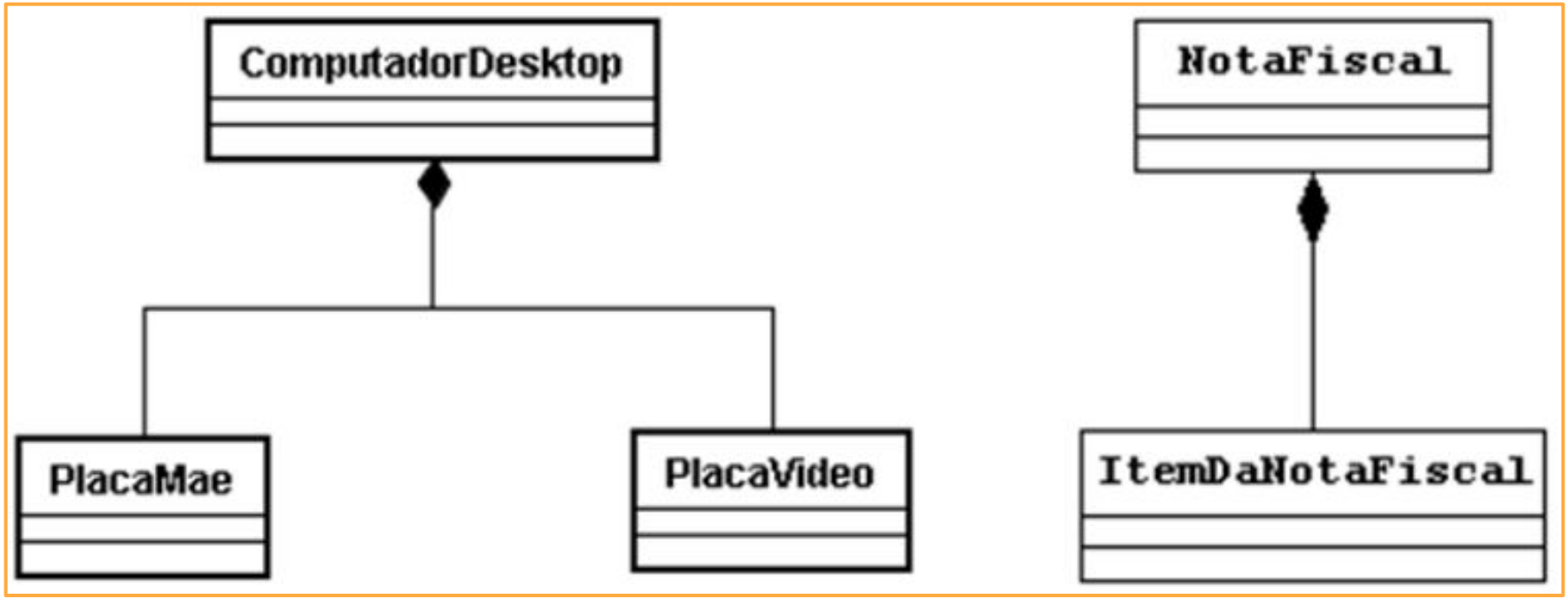
Herança


ContaCorrente
-número
-titular
-saldo
-senha
+sacar()
+depositar()
+consultarSaldo()
ContaEspecial
-limite
+sacar()
Pessoa
-nome
-endereço
-fone
-email
Professor
-títulaçãoMáxima
-disciplinasHabilitadas
Aluno
-curso
-turma

Associações Todo-Parte → Composição

- É um tipo mais forte de relacionamento “todo-parte”, onde **o todo é composto pelas partes**.
- O Todo não existe sem as partes.
- As Partes têm existência independente do Todo.

Associações Todo-Parte → Composição

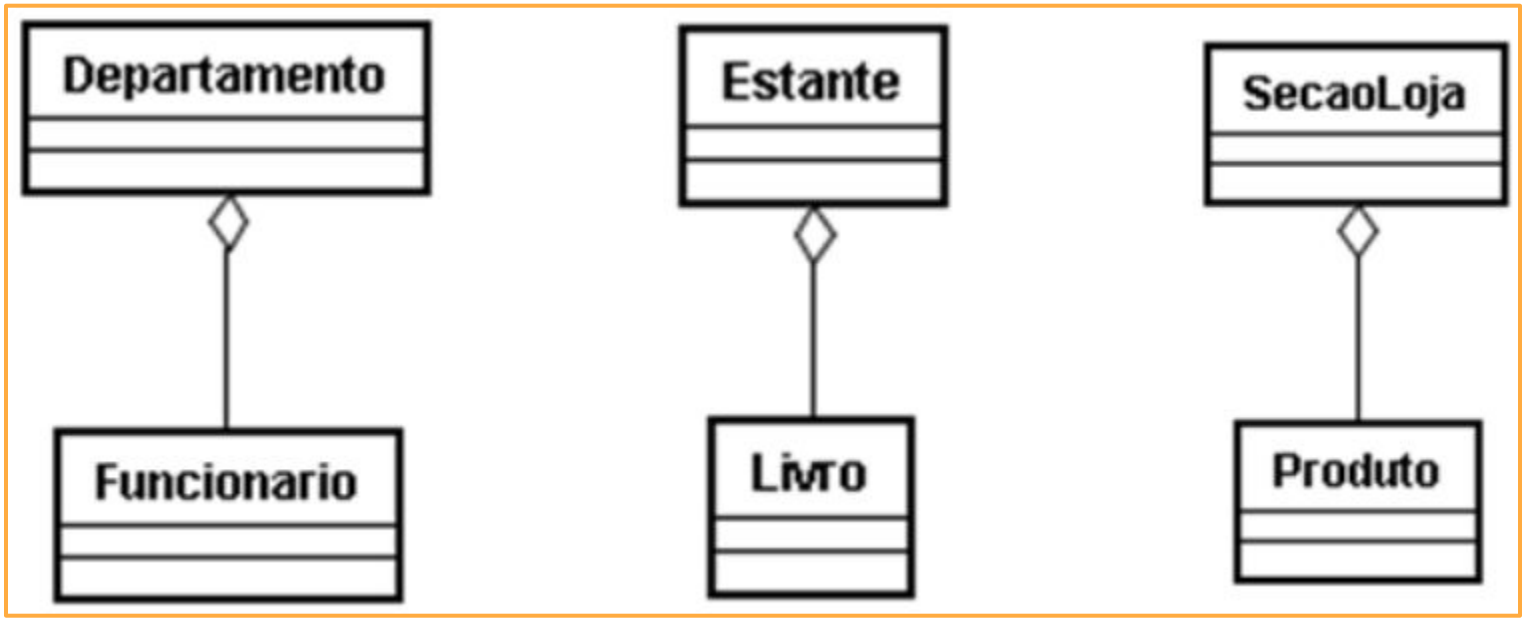


Todo depende das Partes

Associações Todo-Parte → Agregação

- Um relacionamento de agregação é uma associação que reflete a posse lógica.
- Os relacionamentos de agregação são casos particulares dos relacionamentos de associação e indicam um **agrupamento de elementos**.

Associações Todo-Parte → Agregação



Todo é independente das Partes e vice-versa

Tarefas para Próxima Aula - 13/08/2020

1. Criar a conta no Github, um repositório para o VenturaHR e informar o professor.
    - Criar uma estrutura de pastas simples, mas cuidadosa onde todos os artefatos do projeto serão publicados.
    - Migrar a Matriz de Requisitos, Diagrama de Casos de Uso e Diagrama de Classes para o Github.
2. Estudar o artigo:
   https://www.infoq.com/br/articles/onion-architecture/