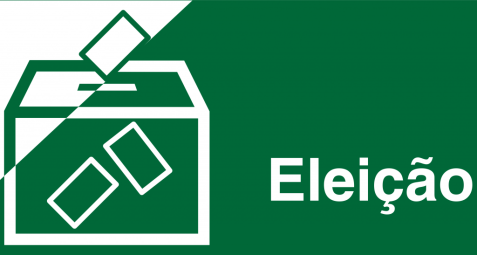


AUTARQUIA EDUCACIONAL DO VALE DO SÃO FRANCISCO – AEVSF

FACULDADE DE CIÊNCIAS APLICADAS E SOCIAIS DE PETROLINA – FACAPE

PÓS-GRADUAÇÃO EM ENGENHARIA DE SOFTWARE

PROGRAMAÇÃO ORIENTADA A OBJETOS



**SISTEMA ELEITORAL**

PETROLINA - PE

2018



AUTARQUIA EDUCACIONAL DO VALE DO SÃO FRANCISCO – AEVSF

FACULDADE DE CIÊNCIAS APLICADAS E SOCIAIS DE PETROLINA – FACAPE

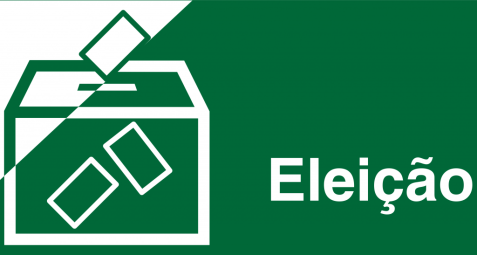
PÓS-GRADUAÇÃO EM ENGENHARIA DE SOFTWARE

PROGRAMAÇÃO ORIENTADA A OBJETOS

ANDERSON FREIRE REZENDE

DAILTON BRAGA

MARCIO ALVES RIBEIRO



**SISTEMA ELEITORAL**

PETROLINA - PE

2018

Sumário

[**1.** **INTRODUÇÃO** 4](#_Toc525047414)

[**2.** **DIAGRAMA DE CLASSES** 5](#_Toc525047415)

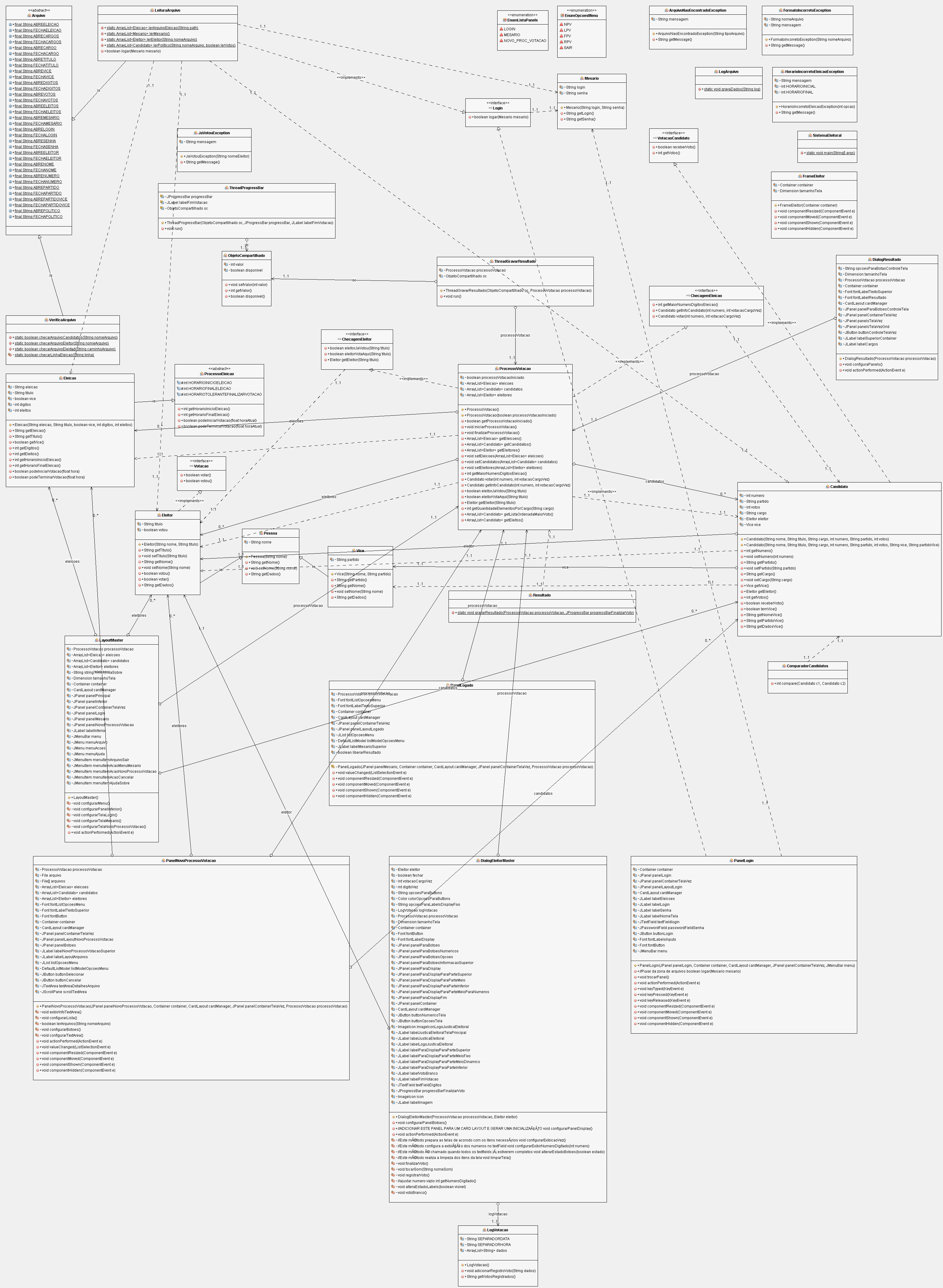
[**3.** **LISTAGEM DE ITENS OBRIGATÓRIOS** 6](#_Toc525047416)

[**4.** **CONCLUSÃO** 8](#_Toc525047417)

1. **INTRODUÇÃO**

Projeto de simulação de um sistema eleitoral desenvolvido em linguagem de programação Java como requisito de aprovação na disciplina de programação orientada a objetos do curso de pós-graduação em engenharia de software da faculdade de ciências aplicadas e sociais de Petrolina.

1. **DIAGRAMA DE CLASSES**



1. **LISTAGEM DE ITENS OBRIGATÓRIOS**
2. **Atributos:**
   1. **Encapsulamento**: Todos os atributos de classes não estáticas são do private. Exemplo: atributo “login” e “senha” da Classe “Mesario” são do tipo private para que não possam ser alterados diretamente.
   2. **String:** Atributo “eleição” da Classe “Eleicao” definido como String para que possa armazenar o nome do tipo de eleição corrente.
   3. **Enumeração:** Enum criado no pacote “enumeracao” no arquivo enum “EnumOpcoesMenu” para exibir as opções do menu.
   4. **ArrayList:** ArrayList criado na classe “ProcessoVotacao” do tipo Candidato para conter todos os candidatos do processo de eleição escolhido.
   5. **Constante:** Constante “ABRECARGO” na classe “Arquivo” criada para identificar a tag de marcação que informa ao programa em qual parte do arquivo contém o cargo do candidato. A constante em questão é utilizada na classe “LeituraArquivo”.
3. **Métodos.**
   1. **Argumentos Variados:** No método “adicionarRegistroVoto” da classe “LogVotacao” é utilizado uma lista de argumentos variados que serão utilizados para gravar o log de dados que foram computados na urna;
   2. **Método Estático:** Método estático “checarArquivoEleicao” criado na classe “VerificaArquivo” é utilizado para verificar se o arquivo de eleição está no formato correto para que o mesmo possa ser lido e convertido em objeto.
4. **Construtor:** Construtor declarado nas classes “ThreadGravarResultado”e “ThreadProgressBar” para realizar configuração iniciais nos objetos de gravação de dados e da barra de progresso ao gravar os mesmos, consecutivamente.
5. **Composição:** Composição do tipo “Eleitor” na classe “Candidato”, pois não faz sentido existir candidato que não seja também um eleitor.
6. **Agregação:** Agregação do tipo “Vice” na classe “Candidato”, pois pode existir candidato que não tenha vice.
7. **Herança:**
   1. **Atributo final em Superclasse:** Atributo final “HORARIOINICIOELEICAO” criado na superclasse **“**ProcessoEleicao**”** para checar se o horário de início das eleições está correto para que a mesma possa acontecer.
   2. **Acoplamento Dinâmico:** A classe “Vice” estende da classe “Pessoa”, na classe “Pessoa” existe o método “getNome”, este método é chamado na classe “Vice” para obter o nome do vice candidato.
8. **Polimorfismo:**
   1. **Sobrecarga:** Polimorfismo de sobrecarga criado no construtor da classe “Candidato”, pois alguns candidatos têm vice e outros não, logo um construtor possui os atributos do vice e o outro não.
   2. **Coerção Explicito:** Utilizado no método“podeIniciarVotacao” da classe “Eleicao”, este método recebe um valor flutuante que representa a fração da hora atual e converte em um inteiro que é utilizado para checar se o horário está correto para iniciar o processo de votação.
   3. **Polimorfismo de Subtipagem:** Utilizado na classe “Pessoa” no método “getDados” que é sobrescrito conforme a necessidade do objeto que é utilizado.
   4. Polimorfismo Paramétrico:
9. **Interface:**
   1. A classe “LeituraArquivo” implementa a interface “Login” que fornece a assinatura do método “logar” para a classe implementar.
   2. A classe “Eleitor” implementa a interface “Votacao” que fornece a assinatura dos métodos “votar” e “votou” que é utilizado para o eleitor realizar seu voto e checar se o mesmo já votou, respectivamente.

* 1. A classe “Candidato” implementa a interface “VotacaoCandidato” que fornece a assinatura dos métodos “receberVoto” e “getVotos” que é utilizado para o candidato receber um voto e checar a quantidade de votos do mesmo, respectivamente.
  2. A classe “ProcessoVotacao” implementa duas interfaces, sendo elas: “ChecagemEleicao” e “ChecagemEleitor”, que fornecem a assinatura dos métodos “eleitorJaVotou”, “eleitorVotaAqui”, “getEleitor”, “getMaiorNumeroDigitosEleicao”, “getInfoCandidato”, “votar” que são utilizadas para determinar os procedimentos que deverão acontecer na classe.

1. **Pacotes:**
   1. O pacote “sistemagrafico” é referente as classes que constroem os recursos gráficos do sistema: Janelas, botões, caixas de diálogo, etc.
   2. O pacote “objetos” é referente as classes de objetos mais simples do sistema, referente aos itens básicos de uma votação, como: pessoas, candidatos, eleitores, etc.
2. **Classe Abstrata:**
   1. Classe abstrata “Arquivo” que contém os métodos e constantes necessárias para realizar o trabalho das classes manipuladoras de arquivos.
   2. Classe abstrata “ProcessoEleicao” que contém os elementos, assinaturas de métodos básicos e constantes necessárias para que o procedimento de eleição possa ocorrer.
3. **Exceções:** a classe “FormatoIncorretoException” foi criada para tratar as exceções que ocorrem quando for verificado que um determinado arquivo está no formato correto. Exemplo: arquivo de eleição com tags incompletas.
4. **Caixas de diálogo:** na classe “PanelLogin” existe uma caixa de diálogo que é exibida quando o usuário (mesário) estiver colocando as informações incorretas de login.
5. **Arquivos:** Ao término da votação de cada eleitor, quando o mesmo finalizar o último voto, os dados dos candidatos serão armazenados em um arquivo na pasta, “Resultado”, que conterá os dados dos candidatos e a quantidade de votos dos mesmos. Estes dados serão recuperados mais tarde, quando o mesário finalizar o processo de eleição, ao exibir a tela de “Resultado”.
6. **CONCLUSÃO**

O desenvolvimento deste trabalho contribuiu expressivamente para o aprendizado dos autores, uma vez que proporcionou conhecimentos específicos da linguagem de programação Java as quais ainda se tinha um conhecimento bem sucinto como também da IDE NetBeans 8.2 muito utilizada no mercado.