**Solução de Software para Sistema de Urna Eletrônica**

**Autores:** João Victor Mendes Freire, 758943
Julia Cinel Chagas, 759314

## 1. Introdução

A urna eletrônica foi inserida no Brasil em uma fase de testes nas eleições de 1996, o que possibilitou o voto tornar-se 100% eletrônico em 2000. No problema proposto trabalharemos no planejamento de uma urna eletrônica comum e, para isso, iremos elaborar a solução deste sistema levando em consideração algumas funções principais, como o recebimento de comandos do mesário para iniciar uma nova votação, de qual maneira a informação será passada para o eleitor sobre a votação e de que forma o voto do eleitor será recebido e registrado.

O objetivo do projeto é aperfeiçoar a construção de um software utilizando os princípios SOLI (sem o D, que ainda não foi estudado) ensinados no decorrer do curso, em especial as duas técnicas mais recentemente aprendidas: *Liskov Substitution Principle* e *Interface Segregation Principle.*

O Princípio da substituição de Liskov presa que uma classe derivada deve ser substituível por sua classe base, permitindo utilizar o polimorfismo com mais segurança, pois é possível chamar as classes derivadas fazendo referências à classe base sem obter resultados inesperados.

O Princípio da Segregação da Interface diz que é melhor um sistema ter interfaces mais específicas sendo criadas do que uma única interface genérica, para não forçar uma classe a implementar interfaces e métodos que não irão ser utilizados.

## 2. Solução

Considerando o documento de requisitos apresentado, chegamos ao seguinte projeto arquitetural.

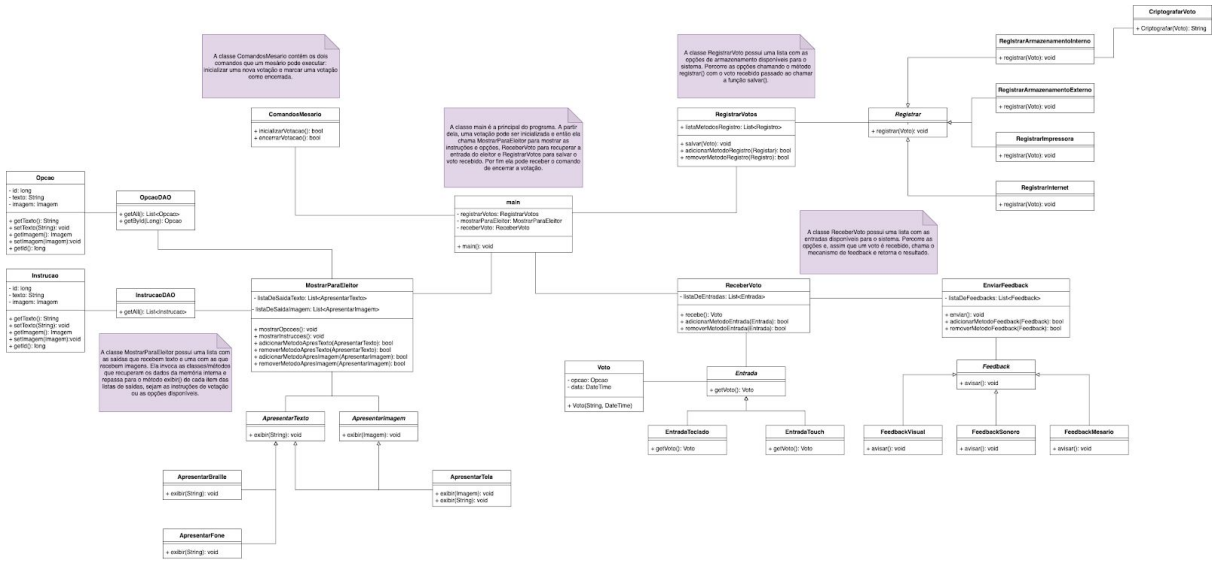


Figura 1: Visão geral do sistema.

Considerando as quatro funcionalidades principais propostas, o sistema foi dividido em quatro grandes módulos. Como pode ser observado na Figura 2, existe uma classe responsável por receber comandos do vindos do mesário (requisito F1). O fluxo da urna acontece através da classe `main`, que após receber a instrução de inicializar uma votação, fica num laço para apresentar informações ao leitor, receber a escolha e salvá-la. O laço é interrompido quando o comando de finalizar a votação for recebido.

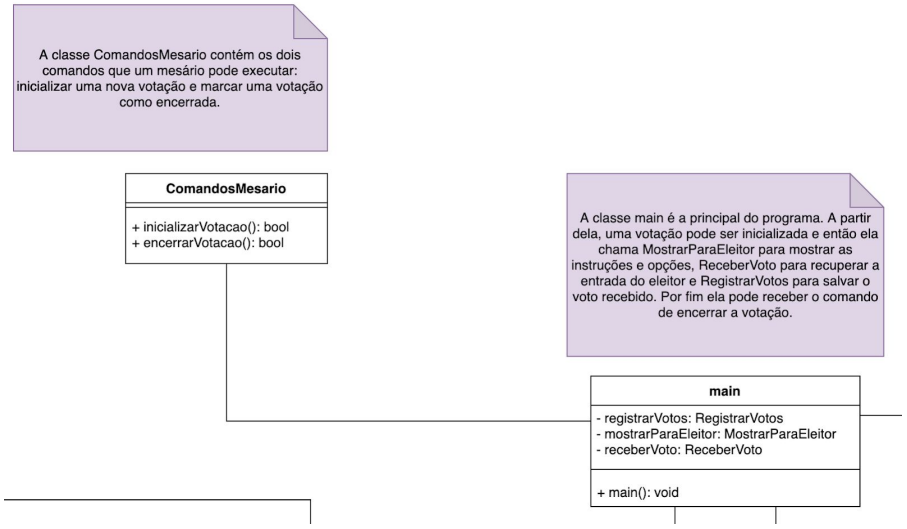


Figura 2: Opções do mesário e classe principal do sistema.

O segundo módulo engloba as classes relacionadas com a apresentação de informações ao eleitor. A Figura 3 contém uma visão geral das classes contidas no módulo.

As classes `Instrucao` e `Opcao` são as abstrações de uma instrução para o eleitor e uma das opções que pode ser escolhida, respectivamente. As classes DAO (Direct Access Objects) recuperam os dados do armazenamento secundário adequado e retornam objetos do tipo das abstrações internas, respeitando o *Single Responsibility Principle*.

O documento de requisitos contempla diversos mecanismos de saída para o eleitor. A fim de modularizar a urna e permitir expansões futuras, foram criadas as seguintes classes abstratas: `ApresentarTexto` e `ApresentarImagem`. Cada dispositivo possível tem uma implementação concreta de uma delas, sendo que ao adicionar um novo mecanismo basta criar uma nova implementação seguindo a interface adequada. A classe principal do módulo, `MostrarParaEleitor`, possui uma lista com os métodos de saída ativos na urna em questão, assim basta adicionar ou remover os objetos da lista conforme os mecanismos disponíveis no aparelho. Isso contribui para que o *Open-Closed Principle* seja respeitado, uma vez que é simples expandir o sistema e os demais módulos não tem contato com as implementações concretas.

A decisão de criar duas classes abstratas ao invés de apenas uma veio da necessidade de existirem saídas que recebem apenas texto, saídas que recebem apenas imagem e saídas que recebem os dois. Assim, para respeitar o *Interface Separation Principle*, tem-se uma interface para cada, e classes como `ApresentarTela` implementam as duas, já que os métodos têm assinaturas diferentes. Todas as classes filhas têm apenas uma implementação particular dos métodos da classe pai (respeitando saídas e entradas), então é possível intercambiar os filhos entre si, de forma que o *Liskov Substitution Principle* é respeitado.

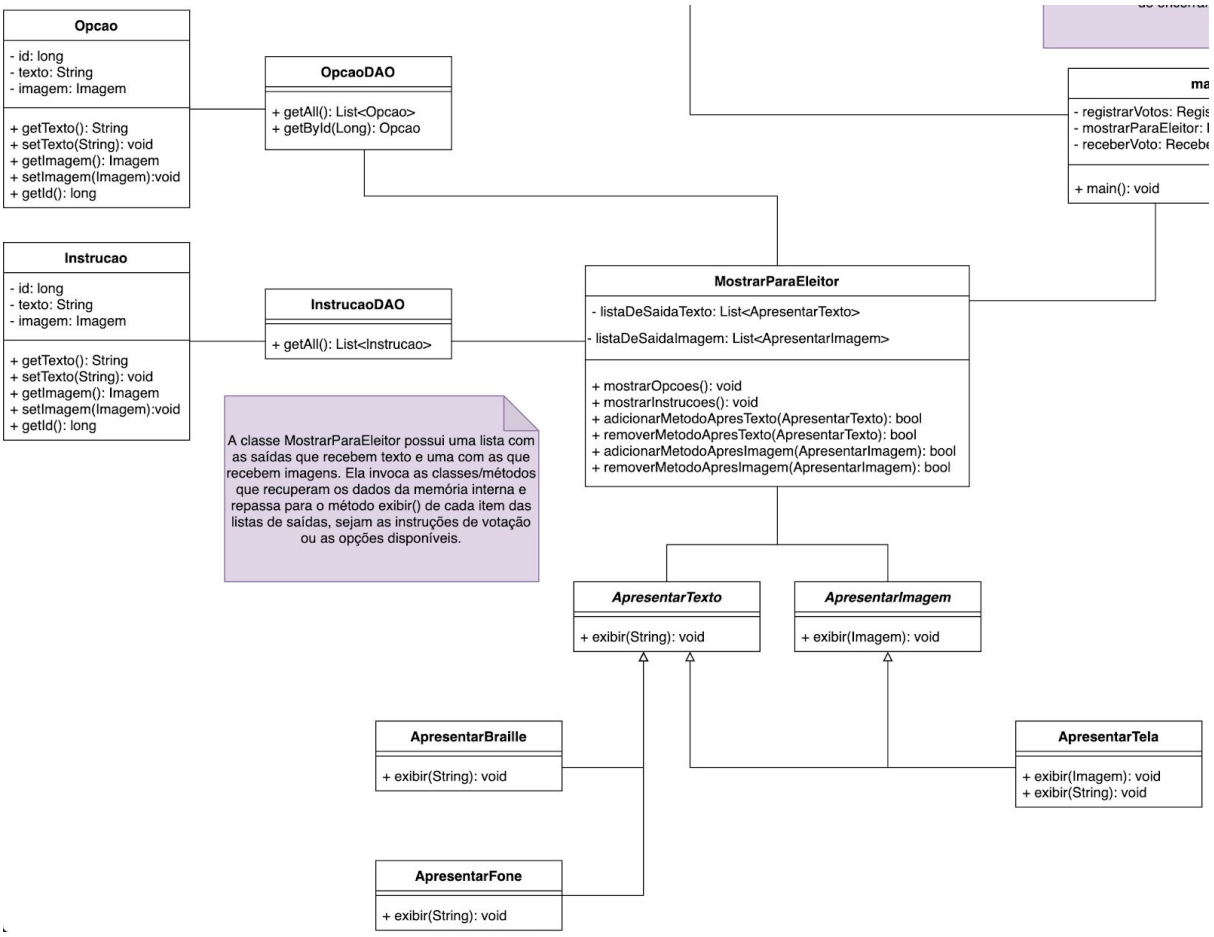


Figura 3: Módulo de exibição das informações para o eleitor.

Já o terceiro módulo é aquele responsável por receber o voto de um eleitor e retornar os feedbacks adequados. Embora o módulo tenha duas funções, o SRP não é violado, pois cada classe continua tendo um único papel. Quando a classe `main` chama a função `recebe()` da classe `ReceberVoto`, o programa percorre a lista de entradas disponíveis até que uma opção seja escolhida, então `enviar()` é chamado da classe `EnviarFeedback`.

De forma análoga ao módulo anterior, as diferentes entradas disponíveis são mantidas em uma lista na classe `ReceberVoto`. Para cada mecanismo, existe uma implementação concreta da classe abstrata `Entrada`, assim como, para cada tipo de feedback, existe uma implementação concreta da classe abstrata `Feedback`. As entradas e as saídas de feedback disponíveis podem ser adicionadas ou removidas para cada urna e novas opções podem ser adicionadas no futuro, assim o OCP é respeitado.

O LSP e o ISP são respeitados, uma vez que existe uma classe abstrata que generaliza as implementações e todas podem ser intercambiadas livremente.

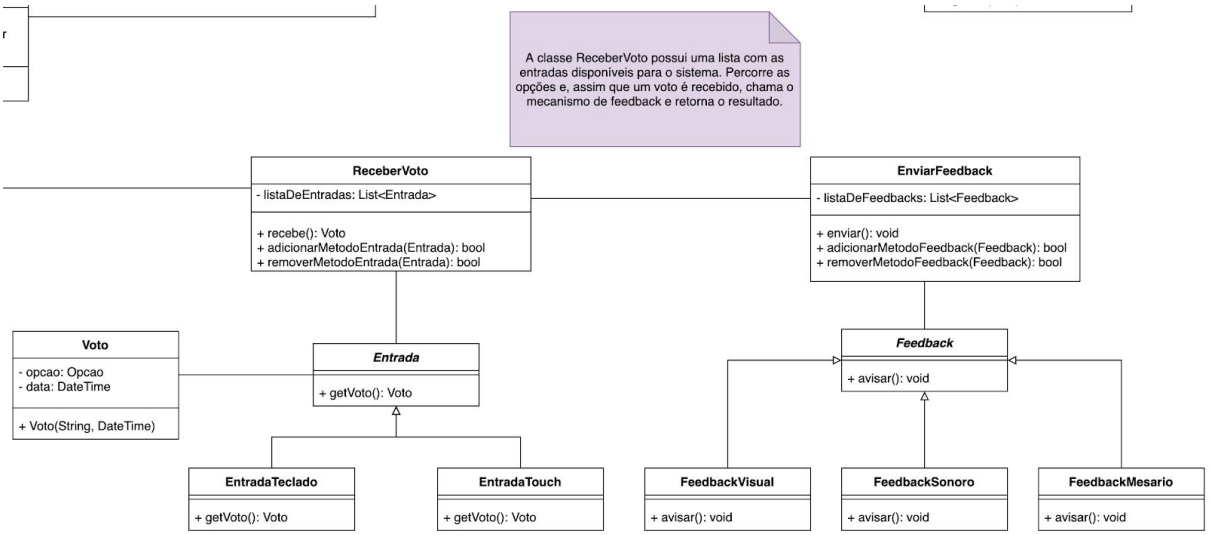


Figura 4: Módulo de recebimento da escolha do eleitor e devido *feedback.*

Tal qual os módulos anteriores, no módulo de salvamento dos votos existe uma classe abstrata `Registrar` e uma implementação concreta para cada dispositivo de armazenamento adequado. Os mecanismos podem ser adicionados ou removidos conforme necessário e novos podem ser adicionados no futuro.

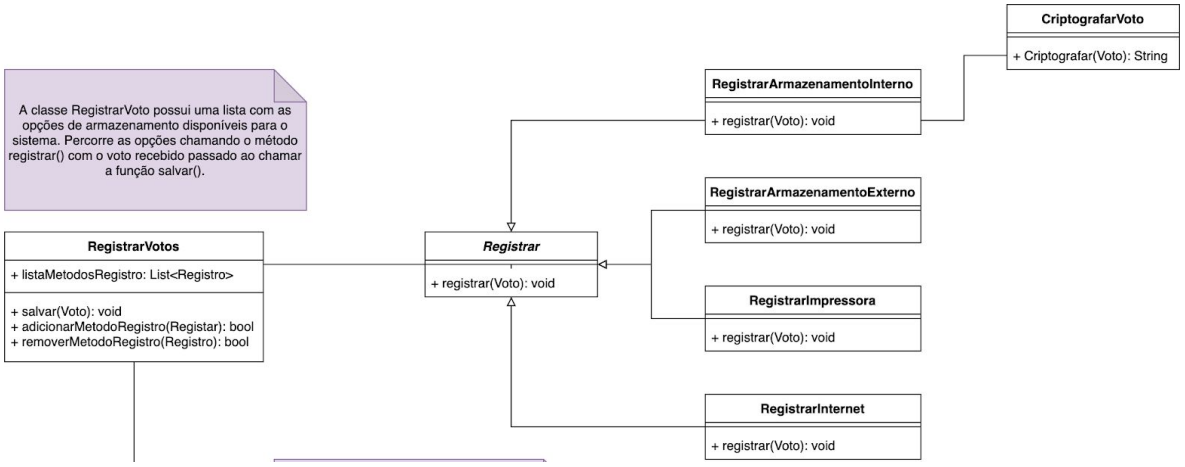


Figura 5: Módulo de salvamento dos votos.

## 3. Bibliografia

[1] LUCRÉDIO, Daniel. **Programação Orientada a Objetos Avançada:** 09 de nov. a 16 de jan. de 2020. Vídeo Aulas. Departamento de Computação, Universidade Federal de São Carlos.

SOLID: Princípios da programação orientada a objetos. Disponível em <https://medium.com/desenvolvendo-com-paixao/o-que-%C3%A9-solid-o-guia-completo-para-voc%C3%AA-entender-os-5-princ%C3%ADpios-da-poo-2b937b3fc530>. Acesso em 05/01/2021.

The Liskov Substitution Principle. Disponível em <https://web.archive.org/web/20151128004108/http://www.objectmentor.com/resources/articles/lsp.pdf>. Acesso em 02/01/2021.