UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Patricia Dousseau Cabral

FRAMEWORK PARA SISTEMAS DE VOTAÇÃO DIGITAL

Florianópolis(SC)

2014

Patricia Dousseau Cabral

FRAMEWORK PARA SISTEMAS DE VOTAÇÃO DIGITAL

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação para a obtenção do Grau de Mestre.
Orientador: Ricardo Pereira e Silva, Dr.

Florianópolis(SC)

2014

Catalogação na fonte elaborada pela biblioteca da
Universidade Federal de Santa Catarina

A ficha catalográfica é confeccionada pela Biblioteca Central.

Tamanho: 7cm x 12 cm

Fonte: Times New Roman 9,5

Maiores informações em:
http://www.bu.ufsc.br/design/Catalogacao.html

Patricia Dousseau Cabral

FRAMEWORK PARA SISTEMAS DE VOTAÇÃO DIGITAL

Esta Dissertação foi julgada aprovada para a obtenção do Título de "Mestre", e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Florianópolis(SC), 04 de abril 2014.

Prof., Ronaldo dos Santos Mello, Dr.
Coordenador

Ricardo Pereira e Silva, Dr.
Orientador

Banca Examinadora:

Ricardo Pereira e Silva
Presidente

Antonio Marinho Pilla Barcellos

Fabiane Barreto Vavassori Benitti

Ricardo Alexandre Reinaldo de Moraes

Dedico este trabalho aos meus pais, Sonia e Nelson, que sempre me apoiaram e são e sempre serão fundamentais na minha vida

AGRADECIMENTOS

Gostaria de agradecer imensamente aos meus pais por todo apoio, carinho e sabedoria, e por sempre me incentivarem naquilo que é importante pra mim.

Gostaria de agradecer ao Professor Ricardo Pereira e Silva por toda orientação, paciência e apoio durante o desenvolvimento deste trabalho. Graças a toda experiência de anos de pesquisa fui capaz de concluir e aprender muito durante o mestrado.

Ao Roberto Silvino da Cunha pelas sugestões e suporte durante a modelagem e implementação do framework, assim como por sanar minhas dúvidas sempre que eu precisei.

Ao Caio Cordeiro da Silva e ao Nelson Mariano Leite Neto por toda a dedicação e esforço despendidos neste projeto. E por terem continuado mesmo quando parecia não haver mais caminho a frente.

Todos vocês foram fundamentais para a conclusão deste trabalho e eu seria incapaz de realizar tudo isto sozinha.

After climbing a great hill, one only finds that there are many more hills to climb.

Nelson Mandela

RESUMO

Votar é um mecanismo amplamente utilizado em tomadas de decisões, sendo comumente empregado por governos e empresas. A confiança no processo de votação é fundamental para a credibilidade do resultado. Cada vez mais eleições são conduzidas através da internet devido à sua disponibilidade e facilidade de utilização. Mas esta prática traz novos desafios, tais como a credibilidade no sistema e o risco de coerção dos votantes. Apesar disto, diversos sistemas de votação online foram e continuam sendo propostos, mas implementá-los e validá-los é uma tarefa complexa e difícil. Para facilitar o desenvolvimento e a avaliação destes sistemas, assim como a idealização de novos protocolos de votação digital, foi desenvolvido um framework orientado a objetos que fornece a estrutura necessária a um sistema de votação, reduzindo o esforço exigido para o seu desenvolvimento. Com isto, é possível estender sistemas e protocolos de uma maneira simplificada e em um período de tempo menor, favorecendo focar nos pontos mais importantes da implementação, como a busca por vulnerabilidades, testes de diferentes cenários de utilização e possíveis ataques, permitindo encontrar pontos fracos que de outra maneira poderiam não ser notados. Para avaliar a adequação do framework, foram desenvolvidos três protocolos de votação digital, assim como um quarto utilizado para mostrar a importância do framework na avaliação de protocolos.

Palavras-chave: votação digital, framework orientado a objetos, protocolos de votação digital

ABSTRACT

Voting is a mechanism widely used in decision making and are commonly employed by governments and businesses. The confidence in the voting process is fundamental to the credibility of the result. Increasingly polls are conducted over the internet due to its availability and ease of use. But this practice brings new challenges, such as confidence in the system, and coercion of voters. Several online voting systems have been proposed, but implementing and evaluating them is a difficult and complex task. To facilitate the development and evaluation of these systems, as well as the idealization of new digital voting protocols, we developed an object-oriented framework that provides the necessary structure of a voting system, reducing the effort required for its development. With it you can extend systems and protocols in an easier way and in a shorter period of time, allowing focus on the most important points of development, such as searching for vulnerabilities, different usage scenarios and tests possible attacks, allowing you to find weaknesses that might otherwise not be noticed. To assess the adequacy of the framework, four digital voting protocols were developed.

Keywords: digital voting, object-oriented framework, digital voting protocols

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O processo eleitoral tem sofrido modificações ao longo dos anos, se tornando mais comum o uso de votações através da internet (VOLKAMER; HUTTER, 2004). Com isto, surgem diversos benefícios, como apurações mais rápidas e eficientes, facilidade na hora de enviar o voto, eliminação da necessidade de deslocamento até a área de votação, possibilidade de verificação do processo e redução dos custos. Mas também surgem novas ameaças, como a facilidade de coação dos votantes e novas possibilidades de fraudes. Levando em consideração estes riscos, diversos protocolos já foram propostos na literatura, tentando evitar algumas, senão todas, as ameaças que vulnerabilizam um sistema de votação digital. Além da dificuldade em idealizar estes protocolos, existe também a dificuldade em implementar um sistema completo que os utilize, para poder, desta forma, validá-los e analisá-los.

1.1 MOTIVAÇÃO

Sistemas de votação digital são normalmente complexos e com alto índice de criticidade, devido ao número de ameaças a que estão vulneráveis (WU; SANKARANARAYANA, 2002). Além disto, protocolos são estruturas bem formadas e, normalmente, uma pequena alteração na sua lógica pode comprometer a segurança do sistema. Não é uma tarefa trivial alterar parte do protocolo ou estendê-lo sem considerar o impacto em toda a sua lógica. Sendo assim, normalmente não são configuráveis, e nem os sistemas que os utilizam, uma vez que são dependentes entre si.

Além disso, devido à complexidade de idealização de protocolos e ao grande número de requisitos que eles devem idealmente atender, torna-se difícil a existência de um protocolo possível de ser utilizado em vários tipos de situações e contextos diferentes. Isto favorece a existência de protocolos com características diferentes, ideais para usos específicos. Como nenhuma das propostas é definitiva, há, com isto, uma demanda para a criação de novos sistemas de votação.

Portanto, temos sistemas pouco flexíveis e difíceis de reutilizar caso se deseje alterar sua lógica de funcionamento. Devido ao grande número de protocolos propostos e à dificuldade em validá-los (VOLKAMER; HUTTER, 2004), mecanismos que facilitem seu desenvolvimento e análise desempenham um papel importante na elaboração destes novos sistemas. Como

as soluções implementadas por eles tendem a ser mais maduras e consolidadas que as implementações recentes de sistemas de votação digital, há uma diminuição na probabilidade de se cometer erros durante o desenvolvimento. Isto também é importante na hora de propor alterações em sistemas já implementados, pois é mais fácil implementar determinadas modificações quando parte da estrutura necessária para suportá-las já está desenvolvida, ou a própria modificação em si já está implementada, sendo necessário apenas reusá-la, o que também diminui a chance de criação de novos erros no sistema.

1.2 HIPÓTESE DE PESQUISA

A disponibilidade de um framework orientado a objetos voltado a sistemas de votação digital será capaz de reduzir o esforço na implementação de sistemas baseados em propostas de protocolo votação por meio de reúso de software. Com isso, será possível avaliar a viabilidade de implementar tais protocolos, bem como suas características, por meio de teste de software.

1.3 OBJETIVOS

- **Objetivo Geral:** Projetar e desenvolver um framework para a criação de sistemas de votação digital que possibilite a implementação e análise de novos protocolos, reduzindo o esforço necessário para o seu desenvolvimento por meio do reúso de software.

- **Objetivos Específicos:**

  - Fazer um levantamento dos sistemas e dos protocolos de votação digital disponíveis na literatura;
  - Fazer a análise de domínio selecionando alguns sistemas para analisar seus pontos em comum;
  - Projetar e implementar um framework que possibilite a implementação desses sistemas;
  - Implementar pelo menos três sistemas utilizando o framework;
  - Avaliar um protocolo proposto na literatura, usando o framework desenvolvido para implementá-lo.

1.4 CONTRIBUIÇÕES

Através da utilização de um framework para sistemas de votação digital é possível reduzir os ciclos de desenvolvimento e testes, tornando o amadurecimento do software mais confiável e facilitando sua análise. Isto é importante para a criação e idealização de sistemas de votação mais seguros e flexíveis.

O framework também se torna importante na hora de avaliar implementações e modificações de sistemas de votação propostos, uma vez que é possível analisar se realmente o sistema atende aos requisitos de segurança que se propôs a atender. Uma vez tendo o sistema implementado, se torna mais fácil realizar um conjunto de testes para verificar suas vulnerabilidades e testar possíveis soluções, aprimorando estes sistemas.

1.5 MÉTODOS DE PESQUISA

A primeira parte do desenvolvimento deste trabalho se deu através do estudo de dissertações e teses sobre votação digital, verificando as potencialidades e vulnerabilidades que acometem esta área de pesquisa. Em seguida foi realizado um estudo dos sistemas de votação digital já propostos na literatura, tendo em vista analisar a viabilidade do desenvolvimento de um framework para a implementação destes sistemas. Foram buscados trabalhos correlatos à proposta de um framework para a produção de sistemas de votação digital ou avaliação de protocolos, mas não foram encontrados trabalhos similares, apenas propostas de protocolos e sistemas, sem foco em reúso. Também foram buscados trabalhos cuja proposta fosse a utilização de frameworks, componentes ou outros mecanismos que permitissem reutilizar partes do sistema para o desenvolvimento de protocolos de votação digital, sendo que foram encontrados apenas propostas de protocolos e sistemas específicos ou não implementados. Após esta etapa, foi realizado um levantamento dos pontos em comum de diversos sistemas para a modelagem do framework, sendo que este processo se deu de forma cíclica. Terminada a modelagem inicial, iniciou-se a implementação do framework, o que exigiu que a modelagem fosse constantemente modificada e atualizada, uma vez que apareceram novos requisitos. Quando o desenvolvimento foi finalizado, foram implementados três protocolos para validarem o framework mostrando sua adequação para o desenvolvimento de sistemas de votação digital. No desenvolvimento de frameworks é importante o desenvolvimento de pelo menos três sistemas para mostrar a adequação no framework em sua área de domínio (JOHNSON,

1993).

1.6 ORGANIZAÇÃO DO TRABALHO

Este trabalho foi dividido em seis capítulos, organizados da seguinte forma:
O presente capítulo apresenta a introdução do trabalho. O capítulo 2 apresenta um panorama sobre votação digital, quais os desafios encontrados, o que caracteriza um sistema de votação digital e quais as dificuldades e benefícios obtidos com a sua utilização. No capítulo 3, são apresentados os protocolos utilizados na modelagem do framework, mostrando o atual estado da arte no campo da votação digital. O capítulo 4 trata sobre o framework, explicando sua modelagem, estrutura e sua utilização. No capítulo 5 é mostrado a utilização do framework para implementar e avaliar um protocolo de votação proposto na literatura, demonstrando como o framework pode ser usado para aprimorar propostas de sistemas de votação digital. Por fim, no último capítulo, apresentamos as considerações finais e os trabalhos futuros.

2 VOTAÇÃO DIGITAL

Neste capítulo é abordado o que é votação digital e quais são suas definições e mecanismos mais correntes. São apresentados os desafios encontrados e as vantagens obtidas quando se considera sua adoção, além dos aspectos relevantes para o desenvolvimento e utilização de sistemas de votação digital.

2.1 INTRODUÇÃO

Em uma sociedade democrática, o direito ao voto é o que garante a participação do cidadão nas tomadas de decisão. Ele é essencial para garantir que o governo siga as vontades da população, permitindo às pessoas escolherem os candidatos que mais bem atendem aos seus interesses. É através do voto que as pessoas podem expressar sua voz, suas inclinações políticas, seus desejos de reforma e manutenção. O direito ao voto é um mecanismo de liberdade política quando certas condições são asseguradas, como liberdade partidária e elegibilidade política (AKANDE, 2011).

Um sistema de votação deve garantir a liberdade política, permitindo que todos os votantes votem de maneira igualitária e confiável. Como afirma o Prof. Celso Antônio Bandeira de Mello Sá (1999): ”Quem vota e em que condições se vota são algumas das questões absolutamente fundamentais para que os mandatos a serem recebidos pelos eleitos possam vir a ser reais instrumentos de representação da cidadania, isto é, para que se cumpram a função que lhes deve corresponder como instrumentos viabilizadores dos ideais democráticos.”Os sistemas de votação devem ser confiáveis para garantir que todos tenham as mesmas condições de voto, para que não haja manipulação nem adulteração, garantindo a participação da população e não a utilizando como falso instrumento de liberdade política.

Uma das grandes dificuldades é a confiança no processo eleitoral. Dependendo do método utilizado, há pouca garantia de que a eleição esteja livre de manipulação, seja através da contagem incorreta dos votos ou da alteração dos votos enviados. Países, como, por exemplo, Brasil, Austrália e Bélgica (WOLF, 2010) já aboliram a contagem de votos de forma manual, não fazendo mais uso de cédulas de papel. E outros, como a Estônia (MADISE; MARTENS, 2006) e a Suíça (BRAUN; BRäNDLI, 2006) deram a oportunidade para que os eleitores votassem através da internet. No caso da Estônia, as eleições de 2005 permitiram que 100% da população escolhesse entre votar

através da internet ou dirigir-se aos postos de votação, como forma de incentivar a participação política dos cidadãos. Muito se discutiu sobre as vantagens e as desvantagens dessa abordagem, pois as consequências poderiam ser bastante críticas, como, por exemplo, a eleição de um candidato eleito por uma minoria.

2.2 TIPOS DE ELEIÇÃO

Basicamente existem dois tipos de eleição: as do tipo plebiscito e as do tipo cargo. As eleições do tipo plebiscito possuem uma estrutura mais simples, normalmente a disputa se dando na forma de uma pergunta, seguida de suas possíveis respostas. Já as eleições do tipo cargo exigem toda uma estrutura de candidatos, podendo em alguns casos, tais como eleições políticas, contar com a existência de partidos, coligações, chapas e outras características importantes. Estas eleições tem como disputa os cargos concorridos, como, por exemplo: presidente, gerente ou zelador e tem como opções de voto os perfis dos possíveis candidatos.

2.3 REQUISITOS DE SEGURANÇA

Para avaliar um sistema de votação é útil elencar alguns requisitos de segurança que tornam uma eleição confiável, seja ela física ou digital. Samarone Araujo (2002) cita alguns requisitos importantes:

- Exatidão: garantir que apenas cédulas válidas serão contadas na apuração e que estas não podem ser alteradas ou duplicadas.
- Unicidade: garantir que apenas votantes autorizados participem da votação, cada um emitindo apenas um voto.
- Privacidade: não ser possível associar o voto ao votante (anonimato), não ser possível conhecer a opção escolhida pelo votante (não-coação) e todos os votos devem permanecer em segredo até o fim da apuração (imparcialidade).
- Verificabilidade: existem dois tipos de verificabilidade, a individual, que permite ao votante verificar que seu voto foi corretamente apurado, e a universal, que permite verificar que todos os votos foram apurados corretamente.
- Equanimidade: não deve haver distinção dos eleitores de acordo com raça, condição física, classe social, escolaridade, etnia ou localização

geográfica, além disto, ninguém deve levar vantagem sobre os outros, como saber o resultado parcial antes do término da votação (BOUGHTON, 2006).

Existe uma dificuldade inerente em atender a todas estas exigências uma vez que algumas tendem a ser mutuamente exclusivas, como, por exemplo, a dificuldade em provar que o voto do votante foi corretamente apurado e ao mesmo tempo não revelar sua opção de voto. Ou a dificuldade em permitir que apenas eleitores autorizados votem, sem associar o voto ao votante.

2.4 MECANISMOS DE VOTAÇÃO

Devido a grande variedade de mecanismos de votação, é útil distinguilos para facilitar a compreensão. No artigo "Votación electrónica basada en criptografía avanzada"os autores propõem a seguinte classificação (OLIVA et al., 2002):

- *Modelo clássico de votação* (Votação convencional): faz uso de cédulas de papel ou cartões perfurados e utilizam apuração manual dos votos. Não podem ser consideradas como um sistema de voto eletrônico, mas tem servido de referência para muitas propostas.

- *Modelo híbrido* (Votação eletrônica): se utiliza de mecanismos clássicos combinados com algum processo eletrônico, como urnas eletrônicas, cartões magnéticos, softwares de diversos tipos, leitores ópticos, etc. Segundo os autores, este é o modelo mais amplamente empregado hoje pelos governos, como os do Brasil, Costa Rica, Holanda e Japão.

- *Modelo que utiliza redes de telecomunicação* (Votação digital): neste caso existem duas divisões possíveis: os sistemas que utilizam alguma forma de comunicação privada ou pública entre as diferentes zonas de votação e os sistemas que utilizam a internet, oferecendo maior mobilidade aos usuários, por exemplo, permitindo que se emita o voto de casa.

2.5 REQUISITOS DE IMPLEMENTAÇÃO DA VOTAÇÃO DIGITAL

Idealmente, um sistema de votação digital deve atender algumas característica específicas, de forma a facilitar a interação do usuário com o sistema e aumentar sua versatilidade (WU; SANKARANARAYANA, 2002).

- Conveniência: o sistema deve ser simples, fácil e rápido de ser utilizado, tanto para os votantes quanto para os administradores.
- Flexibilidade: deve ser possível a criação de diferentes tipos de eleição, com múltiplas escolhas ou perguntas que exijam que o votante escreva uma resposta. Além disto, é desejável que o votante possa utilizar diferentes dispositivos para enviar seu voto, tais como celulares, tablets, notebooks, desktops, palmtops, etc.
- Mobilidade: os votantes devem ter a possibilidade de enviar seu voto onde estiverem, sem restrição de localização, desde que tenham acesso à internet.
- Escalabilidade: permitir número indefinido de participantes sem que isso interfira drasticamente no desempenho da votação
- Eficiência: a apuração dos votos deve ser realizada dentro de um tempo aceitável, além de não exigir o cálculo manual das cédulas.

2.6 PREOCUPAÇÕES DA VOTAÇÃO DIGITAL

Algumas questões devem ser levadas em consideração por serem pontos críticos dos sistemas de votação, podendo se tornar portas para possíveis ataques. Chuan-Kun e Ramesh Sankaranayana (WU; SANKARANARAYANA, 2002) citam alguns aspectos que devem ser levados em consideração quando se desenvolve ou se adota um mecanismo de votação digital:

- Confiança no software: conforme o software se torna mais complexo, maior a probabilidade de conter erros, que nem sempre são detectados através da realização de testes, além de não ser possível avaliar quão críticos esses erros podem ser. Apesar disso, existem muitos sistemas com alto índice de criticidade, como, por exemplo, controladores de voos e sistemas de controle de mísseis, que acabam sendo muito mais complexos que sistemas de votação digital. Isto indica que é possível implementar sistemas digitais de votação através da internet de forma confiável, seguindo determinadas práticas e padrões de desenvolvimento de software.
- Confiança na internet: a internet, por ser um sistema aberto, acaba sendo difícil de proteger e vulnerável a ataques. Alguns podem causar apenas pequenos aborrecimentos, enquanto outros podem ser bastante severos, impedindo, por exemplo, a utilização do sistema pelos votantes. Apesar disto, uma rede bem planejada pode ser bastante robusta a

ataques físicos, e um bom gerenciamento de rede pode funcionar bem. Como a internet vem sendo cada vez mais usada, os ataques também se tornam mais severos, mas as soluções também se tornam mais robustas. A vantagem da votação digital é que o seu tempo de duração não é muito extenso, ficando menos vulnerável a ataques.

- Confiança no sistema de armazenamento: certas informações, como votantes autorizados e cédulas de votação enviadas, devem ser guardadas em algum sistema de armazenamento, como um disco rígido. Estes sistemas podem ser violados, corrompidos ou danificados, perdendo informações importantes ou mesmo tendo estas informações alteradas. Estas características também estão presentes em diversos outros tipos de aplicativos, sendo importantes, por exemplo, no comércio online.

- Confidencialidade do voto eletrônico: quando o voto é enviado eletronicamente ele não deve ser legível por ninguém além da autoridade responsável pela apuração. Além disto, o sistema deve ser seguro contra ataques como *man-in-the-middle* e a replicação de cédulas. Determinadas primitivas criptográficas ajudam a assegurar estes requisitos. Um dos maiores problemas, também encontrados na votação convencional, é quão confiável determinada autoridade é, para que ela seja responsável pela manipulação dos votos. Uma solução possível é manter estas autoridades, supostamente confiáveis, sob alta supervisão e através da adoção de mecanismos tais como múltiplas autoridades e assinatura digital.

- Detecção de voto duplicado: no modelo convencional, onde o votante é obrigado a se dirigir até uma cabine de votação para emitir seu voto, é mais complicado votar mais de uma vez. Já em um sistema online, é mais difícil assegurar que o votante vote apenas uma vez, pois isso dificulta garantir o anonimato caso se identifique os votantes.

- Compra de votos: o votante pode vender seu voto ou ser coagido a votar de maneira específica. Existem mecanismos que podem dificultar estas práticas, mas é impossível impedi-las 100% em um ambiente online.

- Ataques terroristas à internet: sistemas digitais também estão vulneráveis a ataques terroristas, sejam eles sistemas de votação ou outros sistemas com alto índice de criticidade. Uma forma de minimizar os problemas decorrentes de um ataque, como, por exemplo, a falta de acesso ao sistema, seria permitir que as pessoas pudessem votar através de um sistema de votação físico, além de garantir a integridade das informações em caso de um ataque.

2.7 BENEFÍCIOS DA VOTAÇÃO ONLINE

Quando utilizamos os mecanismos convencionais, exigindo que o votante se desloque até uma central de votação, fica mais difícil garantir alguns requisitos. Por exemplo, nem todos os votantes tem condição de se deslocar até as centrais, seja por deficiência locomotora, seja pela falta de transporte até a zona de votação. Em alguns casos, por morarem em áreas rurais ou regiões com menos zonas eleitorais, alguns eleitores acabam enfrentando grandes obstáculos para poderem emitir seus votos, o que leva muitos a se absterem de votar. Além disto, longas filas de espera podem desincentivar os votantes a participarem das eleições. Ao tornar o processo de votação mais conveniente, permitindo ao eleitor emitir seu voto em casa, pode ser possível o aumento da participação e engajamento político dos eleitores.

Caso a eleição não seja governamental, a possibilidade de votar online de forma segura também beneficiaria vários outros setores, como, por exemplo, empresas com acionistas de diversos países, permitindo que pessoas em localidades diferentes votassem sobre o mesmo assunto, ao mesmo tempo, e sob as mesmas condições. Além de exigir menos recursos financeiros e uma menor demanda de tempo para conduzir todo o processo.

Um sistema de votação digital contorna alguns dos pontos críticos de um sistema de votação físico, como por exemplo facilidade de manipulação das cédulas físicas. Segundo Michael Ian Shamos Shamos (2011) ”Toda forma de cédula de papel já concebida pode e foi manipulada, em geral, com grande facilidade”. Além disto, a apuração e a recontagem de votos é bastante trabalhosa e demorada, normalmente exigindo maiores recursos e sendo mais propensa a falhas humanas. Sistemas digitais de votação permitem que a apuração se dê em um tempo muito menor, de forma automatizada e exigindo menos recursos humanos.

Também é interessante permitir que a população escolha a forma que deseja votar, como fazem alguns países, por exemplo, a Estônia, que permite aos votantes votarem tanto através da internet, quanto em centrais específicas de votação. Isto permite que os votantes enviem seu voto dias antes do término da eleição, diminuindo as filas no dia da votação presencial e evitando o congestionamento do sistema de votação online, uma vez que nem todos irão votar no mesmo dia.

2.8 RISCOS DA VOTAÇÃO ONLINE

Apesar dos benefícios que a votação online traz, é importante considerarmos os riscos envolvidos com a sua adoção, para podermos avaliar se é uma solução praticável que cumpre aos requisitos e objetivos que se propõe a atender. José Rodrigues-Filho (RODRIGUES-FILHO; ALEXANDER; BATISTA, 2006) alerta para os riscos de uma adoção da votação digital baseada mais na corrida tecnológica do que realmente nos ganhos obtidos. Questiona se a implantação da votação digital é movida por uma demanda popular e pelos benefícios que isto trará para a sociedade, como o aumento da confiança do votante no sistema político e a promoção do engajamento da população na vida política, ou se é apenas a adoção de uma nova ferramenta guiada pela sua disponibilidade e pelo receio dos países em não ficarem atrás na corrida tecnológica.

Um dos riscos é a vulnerabilidade de um sistema online, pelo fato da internet ser um canal para possíveis ataques e manipulações de informação. A impossibilidade de supervisão do sistema por supervisores durante o processo de votação facilita a compra e venda de votos e de outras formas de pressão e coação do votante. Como o votante pode emitir seu voto dentro de sua casa é impossível garantir sua privacidade, e nada garante que ele não esteja com uma arma na cabeça ou sendo manipulado de outras formas. Além disto, existe a dificuldade em garantir a identidade do votante, uma vez que não existe um fiscal para assegurar que o eleitor é realmente quem afirma ser. Justamente por isso, a venda de identidades eleitorais se torna uma prática possível.

Devido ao fato das características da votação online serem diferentes da votação física, como a dificuldade de garantir a identidade do votante, de impedir a manipulação dos votos, a duplicação de cédulas e outros problemas, muitas vezes o sistema acaba se tornando bastante complexo para se contornar estas ameaças. Para quem é leigo em votação digital, muitas vezes é bastante difícil entender o funcionamento do sistema, restando apenas confiar na sua exatidão, honestidade e segurança (MADISE; MARTENS, 2006). Isto pode acabar diminuindo a credibilidade no sistema e no resultado da eleição, além de plantar um sentimento de insegurança em alguns votantes.

Dentre as novas dificuldades encontradas na votação online, podemos citar a duplicação de cédulas, uma vez que a cédula agora pode ser eletrônica, e não mais física, muitas vezes tornando mais fácil sua replicação e assim o envio de inúmeros votos pelo mesmo votante ou a substituição de votos

existentes por outros falsos. Outra dificuldade é satisfazer alguns requisitos conflitantes, como a confidencialidade e a auditoria. A confidencialidade exige que o voto se mantenha anônimo, e a auditoria exige que tudo que seja efetuado no sistema seja gravado.

A internet é considerada uma plataforma reconhecidamente insegura, permitindo que vários tipos de ataques, tais como negação de serviço, worm, trojans, vírus, spy wares e spoofing sejam realizados para comprometer os resultados da eleição, o anonimato dos votantes ou interromper o processo de votação (MADISE; MARTENS, 2006).

Devemos considerar que algumas dessas ameaças também estão presentes mesmo em votações presenciais, onde muitas vezes o votante se vê coagido a votar em determinado candidato e comprovar seu voto através da utilização de fotos tiradas com celulares dentro da cabine. Muitas vezes a questão não se limita em impedir totalmente determinada prática, mas sim em minimizar suas possibilidades.

Para contornar alguns dos problemas descritos, existem diversas alternativas. A grande dificuldade é contornar todos os problemas e ainda garantir todos os requisitos necessários a uma eleição democrática. Devido a isto, diversos autores propõem protocolos de votação para situações específicas, ou seja, que são confiáveis em determinados contextos e atendem apenas a parte dos requisitos. Podemos citar como exemplo o caso do sistema de votação Helios que é adequado para eleições onde a votação deve ser secreta, mas onde coação não é uma grande ameaça (ADIDA, 2008). Como exemplo disso, podemos citar votações para clubes, comunidades de software e comunidades estudantis. Já o sistema descrito por Chuan-Kun Wu e Ramesh Sankaranarayana é adequado para votações onde haja a necessidade de votações livres de coação, pois o sistema contorna essa ameaça permitindo que o votante vote inúmeras vezes, dificultando a possibilidade de forçar o votante a escolher determinada opção, uma vez que ele pode alterá-la mais tarde (WU; SANKARANARAYANA, 2002).

2.9 TABELA COMPARATIVA DOS BENEFÍCIOS E RISCOS DA VOTAÇÃO ONLINE

Como forma de visualizar melhor as vantagens e os possíveis riscos da adoção de votações online foi montado um quadro comparativo (Tabela

3). Vale lembrar que algumas das ameaças presentes em votações onlines também são encontradas em votações presenciais.

2.10 CONCLUSÃO

Sistemas de votação digital devem atender a um grande número de requisitos de segurança e de implementação, o que torna o trabalho de idealizar e implementar estes sistemas uma tarefa complexa. Devido a isso, diversos autores propõem protocolos para situações específicas, de forma que sejam projetados para alcançar parte dos requisitos, considerando de antemão que serão vulneráveis a determinadas ameaças. Isto é importante pois alguns requisitos tendem a ser conflitantes, sendo difícil atendê-los ao mesmo tempo.

Benefícios	Riscos
Facilidade na hora de enviar o voto do ponto de vista do votante, permitindo que se vote em casa. Isto é importante para eleitores com dificuldades locomotoras, que moram no exterior, em zonas rurais ou afastados da zona de votação.	Dificuldade na hora de provar a identidade do votante, uma vez que não existem fiscais para assegurá-la.
Maior agilidade na hora de computar os votos, não sendo necessário uma apuração manual.	Possibilidade de efetuar uma contagem tendenciosa, que não represente a correta somatória dos votos.
Possibilidade em aumentar o engajamento político da população e a participação nas eleições, pela maior facilidade em participar das votações.	Maior possibilidade de coação dos votantes, uma vez que é difícil garantir que não existem outras pessoas ao lado do votante, influenciando sua decisão ou obrigando-o a votar em determinada opção.
Dependendo do protocolo adotado é possível que o votante verifique se o seu voto foi corretamente apurado e se o somatório dos votos está correto, aumentando a credibilidade na eleição.	Dificuldade em compreender o funcionamento do sistema, o que pode levar a sua descredibilidade.
Divulgação dos resultados mais rapidamente, com menos recursos financeiros e de forma mais confiável, por permitir práticas tais como verificação individual e universal.	Ameaças presentes no computador do votante, tais como trojans, spywares, vírus, worm, negação de serviço e spoofing.

Tabela 1: Tabela mostrando as vantagens e os riscos do uso de sistemas de votação digital

3 ANÁLISE DE DOMÍNIO: PROTOCOLOS E SISTEMAS DE VOTAÇÃO DIGITAL

Diversas propostas de protocolos já foram apresentadas e colocadas em prática, mostrando quais requisitos são atendidos e a quais ameaças estão vulneráveis. Este capítulo visa traçar um panorama do que existe de mais atual na área de votação digital, fazendo um levantamento dos protocolos e sistemas considerados mais relevantes. Descreve suas características, casos de uso, rotina do processo de uma eleição e seus pontos fortes e fracos. A partir dos méritos e das limitações das soluções estudadas, buscamos subsídios para a modelagem do framework, definindo quais características deveriam ser contempladas e quão flexível deveria ser a sua modelagem para permitir a implementação de uma gama considerável de sistemas e protocolos, de forma a ser possível obter um grande índice de reúso.

3.1 HELIOS

O Helios (ADIDA, 2008) é um sistema de votação digital desenvolvido para grupos com baixo risco de coerção, que mesmo assim ainda precisam de eleições confiáveis e secretas, tais como votações estudantis ou de pequenos grupos. Uma das principais características do Helios é que qualquer um pode auditar o processo de votação, mesmo não sendo um votante cadastrado. Isto significa que é possível auditar a cédula para verificar se ela foi corretamente cifrada, ou seja, que não foi inserida outra opção de voto que não a escolhida pelo votante. O principal artifício para permitir isto, é a separação entre gerar a cédula para o votante e enviar o voto para a urna. Uma cédula pode ser gerada e preenchida por qualquer um a qualquer momento, sem autenticação, com o votante sendo autenticado apenas na hora de enviar o voto. Isto permite que um número maior de usuários verifique a honestidade do sistema, uma vez que não precisam se autenticar para auditar o sistema. Além disto, o Helios possui outra forma de auditoria, conhecida como universal, que permite garantir que tanto o embaralhamento das cédulas durante o processo de votação, quanto a sua decifragem, deram-se de maneira honesta. Isto é necessário pois o Helios utiliza uma rede de mistura para embaralhar as cédulas e não permitir associação entre o votante e seu voto, sendo que cada cédula é recifrada várias vezes durante todo o processo.

O Helios possui uma *Bulletin Board* (BB), para onde todos os votos cifrados são enviados depois de mandados para a urna. Esta BB é acessível a

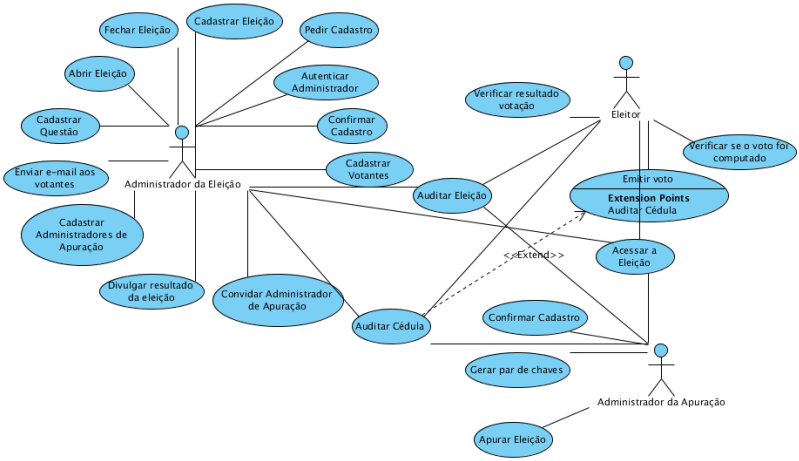

Figura 1: Casos de uso do Protocolo de votação Helios

qualquer um, permitindo que os votantes verifiquem se seu voto foi apurado. É esperado que os auditores verifiquem a integridade da *Bulletin Board* e que um número considerável de eleitores verifique se seus votos se encontram na BB. Isto é necessário para aumentar a confiança no Helios, verificando se ele apurou todos os votos de forma correta.

Para diminuir o risco de interceptação e roubo de informações sobre o voto, os dados não trafegam decifrados na rede. Uma vez que a cédula é carregada no navegador do votante, todas as opções do votante são armazenadas na memória do navegador sem nenhuma transferência de dados com o servidor, até que a cédula esteja cifrada e a cédula sem cifragem seja descartada. Ou seja, todas as operações são realizadas no navegador do votante e a cédula só é enviada ao servidor depois de cifrada.

3.1.1 Casos de uso

O sistema de votação Helios permite diversas atividades para diferentes papeis de usuários, o que pode ser visualizado através da Figura 1.

3.1.2 Problemas e limitações

O Helios é um protocolo vulnerável a coerção, pois ele não impede que outra pessoa esteja ao lado do votante e influencie seu voto, nem dispõe de mecanismos para contornar esta situação. Por isto, o autor afirma que é um sistema que prioriza a integridade (pois emite provas de correção e é universalmente auditável) sobre a privacidade (pois não é livre de coerção).

Outra característica deste protocolo é que caso o Helios fosse corrupto, ele poderia associar o votante ao seu voto e enviar isto para terceiros. Como o autor do protocolo deixa claro, o Helios não é livre de coação, inclusive demonstrando esta vulnerabilidade através de um botão chamado ”coerça-me”. Este botão aparece após o votante cifrar a cédula, e possibilita decifrá-la mostrando o valor aleatório usado na cifragem, assim como toda informação contida na cédula. Ou seja, este botão faz o mesmo que o processo de auditoria, com a diferença que não obriga o votante a recifrar a cédula, enviando-a ao sistema mesmo após mostrar o valor usado para cifrá-la. Com isto, é possível mandar a terceiros estas informações e provar qual a opção de voto escolhida pelo votante, possibilitando a compra e venda de votos, assim como de outras práticas ilícitas.

3.2 PROTOCOLO DE VOTAÇÃO MULTI-CÉDULAS

O protocolo Three Ballot (RIVEST, 2006), proposto por Ronald Rivest, não utiliza nenhum tipo de cifragem, apenas cédulas de papel. Foi proposta sua versão digital, o Protocolo Multi-cédulas (SANTIN; COSTA; MAZIERO, 2008), que leva em consideração a emissão de recibos de votação, venda e compra de votos, materialização do voto (para permitir recontagem manual), auditabilidade do processo de votação, anonimato do votante e autenticidade.

A principal característica deste protocolo é o uso de três cédulas de votação por votante, assim como a existência de três urnas por eleição, o que difere dos outros protocolos analisados. Na hora do votante emitir seu voto, ele obtém três cédulas do sistema através do console de votação. Toda a interação do usuário com o sistema de votação se dá através deste console, que utiliza uma chave específica para o votante, ajudando a garantir seu anonimato. Obtida as três cédulas, o votante tem a oportunidade de selecionar a opção que deseja votar. Para cada candidato, o sistema faz uma marcação em uma das três cédulas. Por exemplo, considerando que temos quatro candida-

tos, o sistema poderia marcar na primeira cédula o candidato C, na segunda o candidato B, na terceira o D e novamente, na primeira, o candidato A. Assim, quando o votante for selecionar seu candidato, por exemplo, o B, ele precisa simplesmente marcar esta opção em qualquer uma das cédulas, exceto a segunda, que já possui uma marcação para este candidato. Em seguida, o votante envia cada uma das cédulas para uma urna diferente. Cada urna contém um representante eleitoral responsável por seu gerenciamento, ficando sob sua responsabilidade a chave criptográfica relativa a esta urna. Sendo assim, cada cédula é cifrada com a chave relativa à urna que irá armazená-la, o que dificulta a obtenção das cédulas de forma legível por pessoas não autorizadas.

Na hora da contagem dos votos, o sistema recolhe as três cédulas já decifradas e verifica qual opção está marcada duas vezes, descobrindo qual a opção escolhida pelo votante. Dessa forma, é necessário ter acesso às três cédulas para revelar a opção escolhida pelo votante. Como cada cédula se encontra em uma urna diferente, cifrada com a chave do representante responsável por aquela urna, seria necessário invadir cada uma das urnas ou corromper cada um dos representantes para descobrir quem votou em quem. Outra vantagem da utilização de três cédulas é que elas individualmente não revelam nada sobre a opção escolhida, sendo possível usar qualquer uma das três como um recibo de votação.

3.2.1 Casos de uso

Este protocolo prevê a existência de três tipos de usuário: o votante, que deve emitir seu voto; o agente de registro, responsável por cadastrar e autenticar os votantes; autoridade eleitoral, que gerencia a eleição e, junto com o representante de cada urna, faz a apuração do resultado. Os casos de uso de cada um dos perfis podem ser visto através da Figura 2.

3.2.2 Problemas e limitações

Caso não seja um problema a compra e venda de votos e coerção ao votante, ou existam mecanismos para impedi-los, acredita-se que esse protocolo possa ser usado para eleições online.

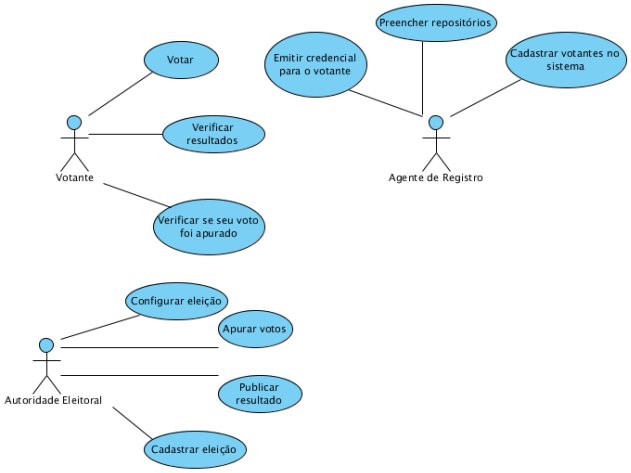

Figura 2: Casos de uso do protocolo da votação Multi-cédulas

3.3 VOTING WITH CODE SHEETS

Este protocolo propõe uma solução introduzida originalmente por David Chaum, conhecida por SureVote (CHAUM, 2001). Nesse novo modelo, conhecido como Code Sheets (HELBACH; SCHWENK, 2007), a agência de votação imprime cartelas que contêm uma lista com todas as alternativas, cada uma associada a dois números aleatórios diferentes em cada cartela. O primeiro número é chamado de TAN de Votação (Voting Transaction Number) e deve ser enviado a um formulário web para informar a opção de voto. Já o segundo número, chamado de TAN de Confirmação, é utilizado para verificar se o TAN de votação foi inserido corretamente.
Este protocolo considera que a Autoridade de Votação é confiável e que o computador do votante é inseguro. Portanto o votante não realiza operações de cifragem, pois não seria possível assegurar sua validade.

3.3.1 Protocolo

Assume-se a existência de um centro de votação confiável que emita as cartelas de votação e que é controlado por uma seleção democrática de pessoas de todos os partidos políticos. Ou seja, todas as cartelas impressas estão

corretas e são distribuídas aos votantes. Também assume-se que a contagem de votos é confiável, ou seja, a autoridade de votação não adiciona ou remove votos. E que a compra e venda de votos é proibida. Este protocolo possui três fases: configuração, votação e verificação do resultado, sendo esta última opcional.

3.3.1.1 Configuração

Nesta primeira etapa a Autoridade de Votação gera a lista com os TAN. Para cada candidato é gerado um TAN de votação e um TAN de confirmação de forma aleatória. Esses dados são impressos em uma cartela e também armazenados em um banco de dados seguro no centro de votação. O endereço dos votantes é impresso nos envelopes e as cartelas são inseridas de forma aleatória nos envelopes, que são enviados aos votantes. Dessa forma não é possível associar o votante à sua cartela, pois esta associação é feita de forma aleatória e não é registrada em nenhum lugar. Outra consideração é que os TAN são grandes o suficiente (no mínimo 15 dígitos) para impedir que um atacante os acerte por tentativa e erro. E mesmo que um atacante consiga adivinhar um TAN de votação válido, ele não será capaz de saber de qual candidato é aquele TAN, pois nem o TAN de votação, nem o TAN de confirmação dizem nada sobre o candidato.

3.3.1.2 Votação

Após o recebimento da cartela de votação, cada votante entra com o TAN de votação correspondente ao seu candidato, no site do sistema. Após entrar com este dado, o TAN de confirmação aparece na tela para que o votante verifique se entrou corretamente com o TAN de votação. Caso o TAN de votação não seja aceito ou o sistema responda com um TAN de confirmação errado, o votante deve reclamar para a Autoridade de Votação. Como é assumido que a Autoridade de Votação é honesta e que a distribuição das cartelas se deu de forma correta, pode-se concluir que o computador do votante está infectado.
Para dificultar a coação dos votantes, este protocolo permite atualizar o voto, sendo que apenas o último será considerado. Isto ajuda a minimizar as ameaças de negação de serviço, já que o votante é encorajado a votar antes, uma vez que pode posteriormente trocar seu voto. Além disso, minimiza a compra e venda de votos, já que o votante pode manter uma cópia da cartela consigo e atualizar seu voto mais tarde o que resultaria em uma corrida entre ambos,

votante e comprador.
Um ataque possível seria mostrar um TAN de confirmação errado ou mostrar uma mensagem de erro após o votante entrar com o TAN de votação. O votante provavelmente tentaria outro TAN para verificar se o mesmo erro ocorre, o que faria com o que o atacante tivesse acesso a outros TAN da cartela. Este é um ataque bastante severo e para isso é proposto um outro código TAN, um TAN de finalização que deve ser inserido para finalizar o voto depois de verificado o TAN de confirmação. Este novo TAN também deve vir na cartela de votação.
Assume-se que os votantes não entrariam com o TAN de finalização a menos que o sistema tenha apresentado um TAN de confirmação correto. Para os casos onde o votante entrou com um TAN de votação, mas não entrou com um TAN de finalização (provavelmente porque houve algum problema), esses TAN seriam publicados em uma *Bulletin Board* especial. Os votantes então deveriam verificar se seus TAN se encontram nessa *Bulletin Board*, e caso se encontrem, deveriam utilizar um outro computador para tentar reenviar o seu voto.

3.3.1.3 Verificação do resultado da eleição (opcional)

O resultado da eleição pode ser verificado publicando os TANs de votação ao lado do nome dos candidatos. Cada votante pode então verificar se seu voto foi corretamente apurado, e contar o número de votos para cada candidato. Mas, com isto, também é possível provar o voto, facilitando sua compra e venda.

3.3.2 Casos de uso

Este protocolo assume a existência de dois tipos de usuário: o votante e a autoridade de votação. O primeiro é responsável por emitir seu voto e o segundo por gerenciar as fases do processo de votação, como pode ser visto através da Figura 3.

3.3.3 Problemas e limitações

Permite a compra e venda de voto, pois é possível provar em quem se votou, caso os TAN de votação sejam publicados na *Bulletin Board*. Caso haja negação de serviço, o votante fica impossibilitado de votar, mas isso é

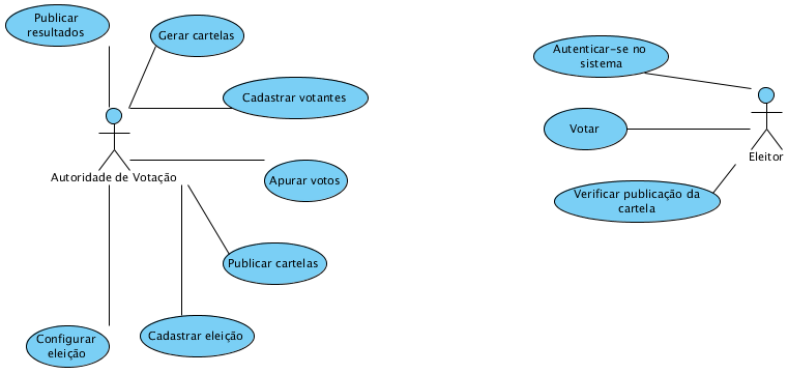

Figura 3: Casos de uso do protocolo de votação Code Sheets

um problema que atinge todos os outros protocolos analisados. Como não é utilizado nenhum mecanismo de cifragem ou canais seguros, caso o atacante conheça o IP do usuário, é possível verificar em quem ele votou interceptando a rede, mas neste caso é necessário ter a cartela do votante para saber qual TAN corresponde a qual opção.

3.4 AN ANONYMOUS ELECTRONIC VOTING PROTOCOL FOR VOTING OVER THE INTERNET

Este protocolo foi desenvolvido por Indrajit Ray et al. Ray, Ray e Narasimhamurthi (2001) para ser usado em eleições em grande escala, pois faz uso de mensagens não rastreáveis, mas ainda assim autenticadas. Não faz uso de técnicas de cifragem complexas. Faz uso de três autoridades que não precisam ser confiáveis. Caso elas conspirem para cometer uma fraude, esta pode ser facilmente detectada e provada, sendo o voto fraudulento não computado.

O protocolo assegura que: apenas votantes legítimos podem enviar votos; apenas um voto é enviado por votante; o votante é capaz de verificar se seu voto foi contado na apuração final; ninguém além do votante é capaz de associar o voto ao votante e, caso o votante não envie um voto, ninguém é capaz de enviar um voto fraudulento em seu lugar. Este protocolo não impede que haja a compra e venda de voto, pois é possível ao votante provar em quem votou. Também permite que se identifique o IP do votante, o que parece ser

um problema, apesar dos autores afirmarem que não.

Uma das características deste protocolo é que ele faz uso de três autoridades:

- Distribuidora de Cédulas: prepara as cédulas em branco e distribui uma a cada votante

- Autoridade Certificadora: certifica que a cédula que foi enviada, foi enviada por um votante registrado e que este votante enviou uma e apenas uma cédula

- Apurador de Votos: cada cédula preenchida enviada pelo votante é entregue ao Apurador de Votos. Seu trabalho é computar os votos e anunciar todas as estatísticas relevantes ao processo de votação.

Assume-se que se algum desses agentes conspirar com um ou mais votantes, eles serão capazes de enviar um único voto, correspondente a cada votante com quem eles conspiraram. Nesse caso, o votante não será capaz de enviar um segundo voto, ou seja, se o votante decidir conspirar, ele prejudicará apenas o seu voto. Esse cenário é similar a utilizar um representante (procurador ou proxy) e não é considerado uma fraude pelos autores.

Outra característica deste protocolo é que o votante precisa gerar uma marca de votação utilizando o identificador de sua cédula. Esta marca é gerada através de permutações difíceis de inverter, e como o identificador da cédula é único para cada votante, esta marca também será. Esta marca é utilizada como identificador do votante, sendo utilizada para verificar se ele já votou anteriormente sem revelar a identidade do votante e também utilizada como recibo de votação ao final do processo eleitoral.

3.4.1 Casos de Uso

Este protocolo considera a existência de três tipos de usuário: votante, responsável por votar; a autoridade de registro, responsável por cadastrar os votantes e gerar suas credenciais; e o administrador, responsável por gerenciar as fases da eleição, como pode ser visto pela Figura 4.

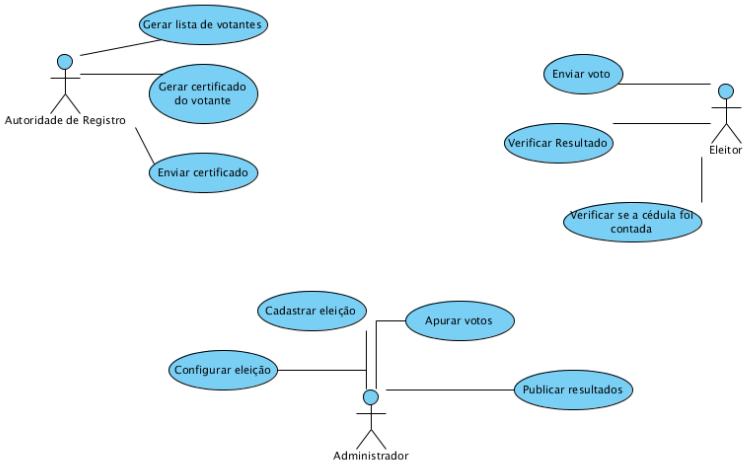

Figura 4: Casos de uso do protocolo de votação Anonymous Electronic Voting Protocol

3.4.2 Problemas e Limitações

É possível saber o IP do votante, o que deixa a suspeita de que isso possibilite sua identificação. Outro problema é que o votante pode provar como gerou a sua marca de votação e provar em quem votou. Com isso, é possível a compra e venda de votos.
Também é possível que a Autoridade Certificadora vote pelos votantes que criaram uma marca de votação, mas não enviaram o seu voto. Se ela conseguir identificar estes votantes conspirando com o Apurador de Votos, ela pode forjar o resto do processo e adicionar votos na apuração.

3.5 SENSUS

O protocolo Sensus (CRANOR; CYTRON, 1997) é uma expansão do trabalho de Fujioka, Okamoto e Ohta Fujioka, Okamoto e Ohta (1993). Ele corrige alguns problemas, como as autoridades conseguirem enviar voto pelos votantes que se abstiveram de votar, e diminui sua complexidade. O Sensus utiliza assinatura cega para assegurar o anonimato do votante, ao mesmo

tempo em que garante que cada votante emita apenas um voto.
Requer a existência de três módulos: um do votante, um do validador e um do apurador. O votante prepara uma cédula já preenchida e cifrada com uma chave privada e blinda. O votante então assina o voto e o envia ao validador, que verifica se o voto foi enviado por um votante legítimo que não enviou outro voto anteriormente. Se a cédula é válida, o validador assina a cédula e manda de volta para o votante. O votante tira o fator de blindagem e obtém uma cédula cifrada e assinada pelo validador. Ele envia esta mesma cédula para o apurador. O apurador verifica a assinatura da cédula cifrada. Caso ela seja válida, o apurador coloca esta cédula em uma lista com outros votos válidos, que serão publicados após o encerramento da eleição. O apurador assina a cédula cifrada e a retorna para o votante para ser utilizado como recibo. O votante então envia para o apurador a chave que foi utilizada para cifrar a cédula. O apurador decifra o voto, e soma com os outros.
Os três módulos são essenciais para conduzir a eleição. Fora estes, podem existir outros para reduzir a chance de erro humano, automatizar tarefas e poupar tempo, como, por exemplo, um escrivão e uma autoridade de cédulas. Para aumentar a segurança do protocolo, o apurador e o validador devem rodar em máquinas separadas.

3.5.1 Protocolo

- Escrivão: responsável por registrar os votantes antes do início da votação. Gera uma lista com o nome dos votantes ou seu identificador, chave pública e opcionalmente um e-mail. O votante precisa enviar um identificador e um token T que precisa ser secreto, ambos gerados por terceiros e enviado ao votante por correio ou outro método. O votante também precisa gerar um par de chaves e enviar a chave pública ao escrivão. O escrivão então verifica se o votante enviou o token correto, e o registra na lista de votantes, junto com sua chave e identificador. Esta lista também contém um campo para saber se o votante já validou seu voto ou ainda não.

- Módulo votante: funciona como o agente do votante, apresentando cédulas compreensíveis ao votante, coletando a resposta do votante, efetuando operações de cifragem a favor do votante, obtendo recibos e validações necessárias e enviando as cédulas para a urna. É o único componente do protocolo que o votante deve confiar completamente.

- Validador: responsável por verificar se o votante está registrado e que apenas um voto seja enviado por votante. O validador gera um cer-

tificado de validação assinando uma cédula blindada. O votante tira o fator de blindagem do certificado de validação e envia ao apurador junto com sua cédula. O validador só emite um certificado por votante, e verifica seu registro na lista emitida pelo escrivão. Depois de emitido o certificado, ele marca nesta lista que o votante já validou seu voto.

- Apurador: responsável por coletar o voto dos votantes e somá-los. Os votantes enviam as cédulas cifradas e assinado pelo validador para o apurador. O validador verifica a assinatura e se a cédula cifrada é única. Caso esteja tudo correto, ele envia um recibo assinado para o votante. O votante envia a chave para decifrar a cédula. Após o término da eleição, o apurador publica uma lista com todas as cédulas cifradas, suas chaves públicas e as cédulas decifradas, para auditoria. Ele calcula o hash de cada cédula cifrada e o utiliza para indexar as cédulas cifradas e os recibos.

3.5.2 Casos de uso

O protocolo Sensus prevê a existência de três perfis de usuário, como visto na Figura 5. O perfil de escrivão, responsável por cadastrar os votantes e pela geração dos tokens; o eleitor, responsável por votar; e administrador, responsável pela condução do processo de votação. Além destes três, é possível que qualquer usuário realize a auditoria do sistema.

3.5.3 Problemas e limitações

É necessário ter confiança no computador do votante, que pode estar infectado e, desta forma, seria possível enviar o voto para terceiros, mudar o valor do voto, etc. Isto é um grande problema, pois vários computadores poderiam ser infectados e manipulados, alterando o resultado da votação. Outro problema deste protocolo é que o votante tem a possibilidade de provar em quem votou, pois recebe um recibo que é publicado ao final da apuração. Isso permite que os votantes sejam coagidos ou que haja compra e venda de votos. É possível que o validador vote por votantes omissos. É possível descobrir quais votos foram forjados verificando a assinatura do voto e comparando com a lista de votantes registrados, mas não é possível corrigir o resultado da apuração

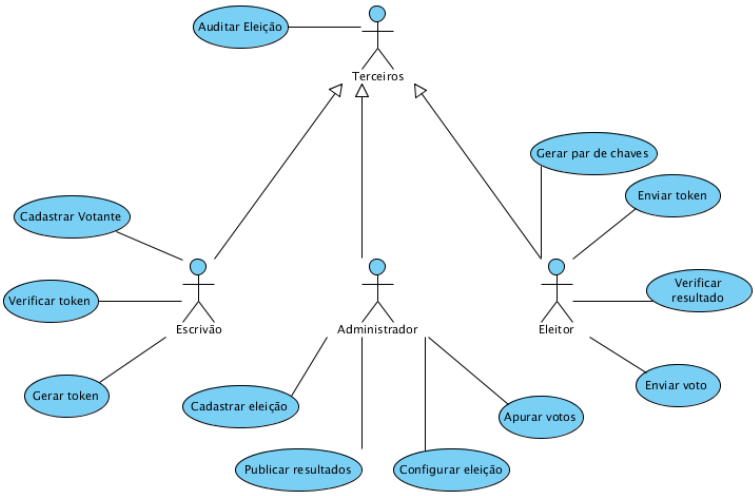

Figura 5: Casos de uso do protocolo de votação SENSUS

3.6 SEAS

O protocolo SEAS (BAIARDI et al., 2004) é baseado no Sensus (CRANOR; CYTRON, 1997), mantendo sua leveza, mas elimina um de seus problemas que é permitir que as autoridades votem no lugar dos votantes que se abstiveram de votar. Pode ser utilizado em eleições online com dezenas de milhares de eleitores.
Este protocolo assume as seguintes entidades:

- Votante
- Validador: servidor que verifica a legitimidade do votante e a unicidade do seu envio, e então valida o voto recebido
- Apurador: servidor dedicado a contar todos os votos válidos
- Escrivão: responsável por registrar os votantes que participarão do processo eleitoral

3.6.1 Protocolo

O protocolo SEAS é uma extensão do protocolo Sensus, ou seja, o protocolo continua o mesmo, com as mesmas autoridades e com as mesmas trocas de mensagens. O que é alterado é que o votante possui agora dois identificadores e dois pares de chave, diferente do Sensus, onde o votante possui apenas um identificador e um par de chaves. Outra alteração é que no Sensus existia apenas uma lista com os dados dos votantes registrados para votar, sendo que apenas o Validador tinha acesso a esta lista. No SEAS esta lista passa a ser acessível tanto ao Validador quanto ao Apurador, e uma nova lista é criada apenas para o Apurador.

Em relação ao protocolo anterior, foi criada uma nova fase de registro, logo após a fase de registro anterior (onde o votante se comunica com o Escrivão). Nesta nova fase, cada votante gera um novo identificador para si e um novo par de chaves. Esta chave será usada para assinar a cédula enviada ao Apurador, permitindo que o Apurador verifique se o voto foi enviado por um votante legítimo, mas sem revelar a identidade do votante.

Agora o Apurador também identifica o votante. No protocolo Sensus, bastava mandar uma cédula válida para que o Apurador adicionasse o voto no cálculo da apuração. Agora é necessário mandar uma cédula válida por um votante legítimo, o que impede que o Validador envie votos por votantes omissos.

3.6.2 Casos de uso

Os casos de uso são idênticos ao do protocolo Sensus.

3.7 SUMÁRIO DOS SISTEMAS E PROTOCOLOS DE VOTAÇÃO DIGITAL

Abaixo segue um resumo de quais requisitos de segurança cada protocolo atende.

Sistemas	Exatidão	Unicidade	Privacidade	Verificabilidade
Helios	Possível verificar a integridade do sistema	Apenas votantes autorizados conseguem emitir seus votos	Não é livre de coerção, sendo possível ao sistema associar o voto ao votante	Verificabilidade individual e universal
Multi-cédulas	Não existem mecanismos para verificar a integridade do sistema	Apenas votantes autorizados conseguem emitir seus votos	Não é livre de coerção, sendo possível ao votante provar o seu voto	Verificabilidade individual
Code Sheets	Não existem mecanismos para verificar a integridade do sistema	É possível que pessoas não autorizadas, tendo acesso a uma cartela, possam votar	Tenta dificultar a coerção dos votantes possibilitanto a atualização dos votos	Verificabilidade individual
Anonymous Electronic Voting Protocol	Existem menicanismos para verificar a integridade do sistema	Caso o votante crie sua marca de votação e não vote, o sistema pode enviar o voto pelo votante	É possível ao votante provar em quem votou	Verificabilidade individual
Sensus	É possível verificar a integridade do sistema	É possível ao sistema votar por votantes omissos	É possível ao votante provar em quem votou	Verificabilidade individual e universal
SEAS	É possível verificar a integridade do sistema	Não é mais possível ao sistema votar por votantes omissos	É possível ao votante provar em quem votou	Verificabilidade individual e universal

Tabela 2: Tabela mostrando quais requisitos de segurança são alcançados pelos sistemas e protocolos

3.8 CONCLUSÃO

Existem diversas propostas de protocolos e sistemas de votação digital tratando sobre como estes sistemas funcionam, sua concepção, vulnerabilidades e implementação. A maioria das propostas são focadas em protocolos e implementações específicas, não havendo, normalmente, reúso algum, uma vez que os sistemas são todos implementados do zero. São poucos os artigos que descrevem o desenvolvimento de protocolos reusando partes de outros sistemas. Podemos citar a proposta de David Lundin em seu artigo "Component Based Electronic Voting Systems"(LUNDIN, 2010), onde ele propõe uma abordagem baseada em componentes para a criação de sistemas de votação digital, o que possibilitaria reusar componentes pré-existentes durante o desenvolvimento de novos sistemas. Apesar disto, o autor não cita qualquer implementação desta proposta. Durante o desenvolvimento deste trabalho, não foram encontrados artigos que tratem o desenvolvimento de sistemas de votação digital através da utilização de um framework dedicado a isto, com os artigos normalmente se restringindo a implementações específicas ou apenas à descrição do protocolo. Também não foram encontrados trabalhos focados na maximização do reúso de software na construção de sistemas de votação digital, como o proposto neste trabalho e apresentado no capítulo a seguir.

Existem algumas propostas de bibliotecas permitindo reutilizar primitivas criptográficas, muitas vezes úteis no desenvolvimento de sistemas de votação digital. Podemos citar o caso da biblioteca LibCryptoSec (LABSEC, 2010), que implementa funções criptográficas tais como criptografia assimétrica e simétrica, dentre outras. Este também é o caso da bibliteca Bouncy Castle (CASTLE, 2004), sendo esta utilizada no desenvolvimento do framework por possibilitar a geração de certificados digitais, hash, assinatura cega e outras funções importantes, como será descrito nos capítulos subsequentes. Estes mecanismos podem ser utilizados para reduzir o esforço de implementação, uma vez que podem ser reusados, apesar de não terem sido originalmente desenvolvidos para este fim.

4 FRAMEWORK ORIENTADO A OBJETOS PARA SISTEMAS DE VOTAÇÃO DIGITAL

Considerando a diversidade de protocolos e sistemas de votação existentes, assim como as diferentes características e demandas de cada eleição, é interessante um mecanismo que abarque esta vasta gama de sistemas e que possibilite sua implementação de forma simplificada. O framework foi projetado considerando os aspectos que devem ser levados em conta quando se idealiza um sistema de votação digital: suas vulnerabilidades, tecnologias utilizadas, possíveis ameaças à sua integridade e todo o gerenciamento de uma eleição. Sua estrutura consiste em diversos módulos que interagem entre si. Um módulo referente às primitivas criptográficas, outro para o gerenciamento de cédulas e opções de voto, para autenticação, para criação de protocolos, para o gerenciamento de usuários e para o gerenciamento de eleições.

4.1 PERFIS DE USUÁRIOS E RESPECTIVOS CASOS DE USO

Um sistema de votação digital deve prover certas funcionalidades básicas, tais como permitir ao administrador o cadastro de eleições e ao votante o envio de seu voto. Algumas delas devem fazer parte de todos os sistemas e outras são específicas para cada caso, podendo ou não ser encontradas em determinadas votações. Este conjunto de funcionalidades é conhecido como casos de uso do framework, estando relacionado com os perfis existentes, como mostram as Figuras 6 e 7. O framework deve ser flexível tanto para a adição de novos perfis de usuários além dos já identificados, como também para novos grupos de funcionalidades.

Foi considerado que existem no sistema os perfis de administrador, votante, escrivão e auditor, sendo cada um responsável por um aspecto do sistema. Isto aumenta a flexibilidade do framework, permitindo um maior controle sobre as ações disponíveis para cada usuário e dificultando sua utilização incorreta.
Abaixo seguem os perfis com suas principais responsabilidades.

- **Administrador:** o administrador do sistema é responsável pelo cadastro das eleições e toda informação relativa a ela, como, por exemplo, data da votação, título, opções de votos, etc. Também é de sua responsabilidade a publicação do resultado da eleição após a auditoria da apuração (caso essa etapa exista). Ao administrador também compete o cadastro de novos usuários no sistema e seus perfis, assim como en-

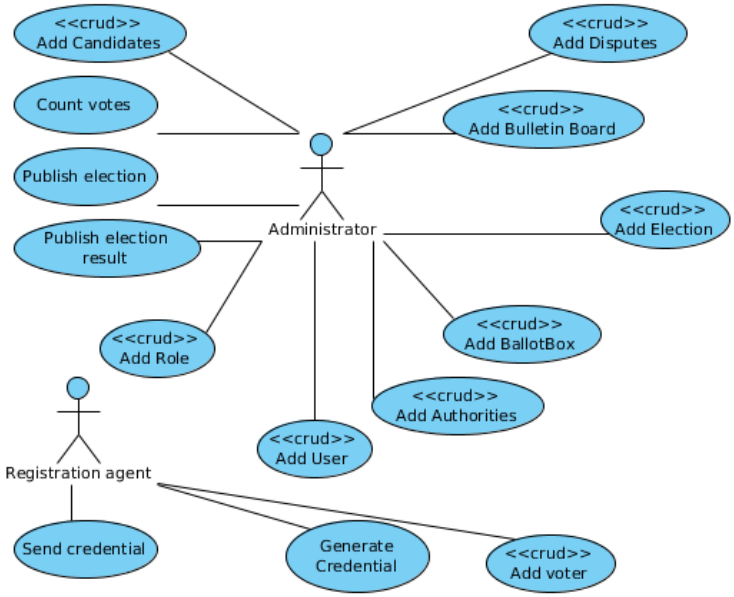

Figura 6: Casos de uso do Administrador e do Escrivão

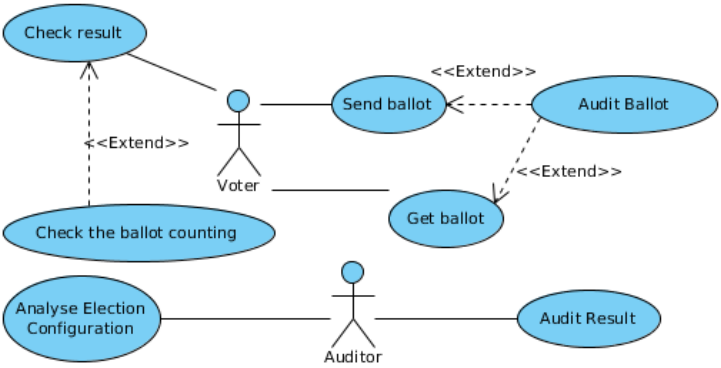

Figura 7: Casos de uso do Votante e do Auditor

tidades externas ao sistema que também fazem parte da eleição, como autoridades de votação, *bulletins boards* e urnas eleitorais que não se encontrem na mesma máquina que o sistema. A contagem dos votos e a publicação do resultado da eleição também fazem parte das obrigações do administrador. Ele também pode ser responsável por outras ações dependendo de quais perfis existam no sistema. Caso não exista o escrivão, ou o auditor, estas funcionalidades podem ser assumidos pelo administrador.

- **Votante:** são os eleitores da votação, sendo o maior grupo de usuários do sistema. As suas atribuições giram em torno da etapa de votação: em obter e enviar a cédula, verificar se sua apuração foi correta e verificar o resultado. A obtenção e o envio da cédula foram separados em duas etapas pois não necessariamente acontecem juntas, dando liberdade ao votante, caso o protocolo permita, de obter a cédula em um determinado momento, se desconectar do sistema, e só mais tarde enviá-la. Dependendo do protocolo é possível ao votante auditar sua cédula durante o processo de votação, para verificar se ela está sendo cifrada e enviada corretamente, sem manipulações.

- **Auditor:** existe no sistema a possibilidade de dois tipos de auditoria. A auditoria da configuração e a auditoria da apuração. Toda eleição pode ou não habilitar a primeira, ficando a decisão a cargo do administrador do sistema. Já a segunda é dependente do protocolo. Caso ao menos uma esteja habilitada, é necessário que ao menos um auditor esteja cadastrado na eleição. A auditoria de configuração diz respeito à análise dos dados da eleição, ou seja, se toda a informação cadastrada pelo administrador está correta: título da eleição, data de votação, opções de voto, etc. A eleição só pode ser divulgada publicamente após a aprovação de todos os auditores. Já a auditoria de resultado diz respeito à análise da contabilização dos votos ou de outras provas que a eleição possa emitir. Esta etapa é dependente do protocolo, e pode ocorrer naqueles que utilizem, por exemplo, redes de mistura, onde são emitidas provas de que o embaralhamento dos votos foi efetuado corretamente. Só após a auditoria de apuração é que o resultado pode ser divulgado. Caso esta seja rejeitada por algum problema, a eleição deve ser cancelada, pois isto prova que houve alguma falha ou fraude durante o processo.

- **Escrivão:** responsável pelo cadastro dos votantes de determinada eleição, assim como pelo gerenciamento das credenciais dos votantes, que permitem sua identificação no sistema. Estas credenciais são dependentes

do protocolo, e podem ser tokens de votação, certificados digitais, login e senhas, ou outro mecanismo de identificação, podendo ser enviadas por e-mail ou disponibilizadas de outra forma. Estas funções também podem ser atribuídas ao administrador, eliminando este papel.

Fora os casos de uso específicos de cada perfil, todos eles têm a possibilidade de se autenticar no sistema e listar todas as eleições a que tem acesso.

4.2 ANÁLISE DE DOMÍNIO

A análise de domínio para o desenvolvimento do framework começou com a identificação de similaridades entre diversos protocolos de votação descritos na literatura. Para isto, foram analisados os protocolos: Helios (ADIDA, 2008), Multi-cédulas (SANTIN; COSTA; MAZIERO, 2008), Sensus (CRANOR; CYTRON, 1997), SEAS (BAIARDI et al., 2004) e o protocolo proposto por Ray e Narasimhamurthi(RAY; RAY; NARASIMHAMURTHI, 2001). Sua escolha se deu por terem atributos pertinentes aos sistemas de votação e por constituírem uma amostra relevante nesta área. Os sistemas escolhidos tem características distintas entre si, o que permite uma visão mais abrangente do que vem a ser um sistema de votação digital. No caso do Helios, ele possui tanto verificação individual, quanto universal. Isto permite que o votante verifique se seu voto foi corretamente enviado e também permite a análise da soma dos votos. No caso do protocolo Multi-cédulas, ele foi escolhido por fazer uso de três cédulas de votação por votante e de três urnas, sendo que dentre os protocolos analisados ele é o único com estas características. Já o protocolo Sensus foi escolhido por exigir três subsistemas, um do votante, um do validador e um do apurador. O protocolo Seas é baseado no protocolo Sensus, com pequenas modificações, com a intenção de eliminar a possibilidade das autoridades votarem no lugar dos votantes omissos, problema encontrado no protocolo Sensus. E, por fim, o protocolo proposto por Ray e Narasimhamurthi por fazer uso de três autoridades, uma distribuidora de cédulas, uma autoridade certificadora e um apurador de votos.

Analisando os protocolos acima, é possível encontrar pontos convergentes entre todos eles e outros aspectos que variam. Com isto, é possível dividir um protocolo em quatro etapas:

- **Inicialização:** esta etapa é opcional, sendo executada no momento da publicação da eleição caso haja necessidade de alguma inicialização do protocolo. Por exemplo, em alguns casos é necessário a inserção

prévia de cédulas em branco na urna, o envio de cartelas aos votantes ou a criação de redes de mistura.

- **Obtenção da cédula:** etapa onde o votante obtém uma cédula e, opcionalmente, realiza a auditoria.
- **Envio da cédula:** etapa onde o votante, após obter a cédula e nela selecionar sua opção de voto, a envia ao sistema para ser contabilizada na apuração.
- **Contabilização:** etapa iniciada após o fim do período de votação, onde todos os votos são contabilizados e o resultado é divulgado. Esta etapa também pode conter a auditoria da apuração, realizada pelos auditores da eleição.

Dentre os pontos em comum podemos citar a utilização de cédulas de votação contendo as disputas e suas respectivas opções de voto e o uso de diferentes papéis que um usuário pode assumir no sistema. Outro aspecto chave é a questão das autoridades. Alguns sistemas, como o Helios, utilizam apenas uma autoridade controladora, que gerencia todo o processo de votação: autenticação dos votantes, recebimento dos votos, apuração e publicação dos resultados. Outros, como o protocolo desenvolvido por Ray e Narasimhamurthi(RAY; RAY; NARASIMHAMURTHI, 2001), distribui a inteligência do sistema entre três autoridades: uma para identificar o votante e emitir as cédulas, outra para verificar se a cédula foi enviada e garantir que cada votante envie apenas uma cédula e outra para computar os votos e divulgar o resultado.

4.3 ESTRUTURA

Para facilitar a organização do framework, sua estrutura foi projetada em módulos. Isto permite alterações em seu funcionamento sem afetar de forma significativa o resto do sistema, permitindo que o desenvolvedor se concentre nos aspectos mais importantes do que deseja implementar e não se preocupe com a parte da estrutura do framework que não diz respeito às suas necessidades. Foram criados 4 módulos, sendo cada um responsável pelo gerenciamento de um aspecto do sistema de votação.

- **Telas:** Este módulo contém a parte gráfica do sistema e a parte lógica responsável pela sua utilização. Ou seja, as telas implementadas, assim como o controle de acesso de usuários e a camada intermediária entre o core e a parte gráfica. As telas pré-existentes no framework são

aquelas que são utilizadas pelos protocolos desenvolvidos e abarcam as principais necessidades dos sistemas de votação digital. Dependendo da complexidade do protocolo não é necessário o desenvolvimento de nenhuma nova tela, sendo possível o reúso daquelas que já foram previamente testadas. Em alguns casos, onde é necessária uma interação diferente por parte do votante ou outro usuário, o desenvolvimento de novas telas, assim como das interfaces intermediárias, pode ser exigido. Estas interfaces, que fazem a ligação entre o core e a parte gráfica, são responsáveis por identificar quais métodos do core cada tela terá acesso, de forma que estas não lidem diretamente com o core. Além disto, este módulo também contém o controle de acesso de usuários, gerenciando as funcionalidades que cada perfil terá acesso em determinada tela. Isto inibe o controle indesejado por parte de alguns usuários à funcionalidades restritas apenas a outros perfis.

- **Core:** O core é a parte principal do framework, contendo toda a lógica do sistema de votação digital. Ele é o núcleo do framework e sua parte mais densa, sendo responsável pelo gerenciamento de diversos aspectos do sistema. Portanto, é aqui que se encontra a máquina de estados da eleição, responsável pelas etapas que permitem que uma votação aconteça do início ao fim. Este módulo mantém, por exemplo, o gerenciador das cédulas, as disputas e as opções de voto, os perfis de usuários, e várias outros atributos e gerenciadores necessários ao funcionamento de um sistema de votação.

- **Protocolos:** Este módulo contém as diversas implementações dos protocolos que serão utilizados pela eleição. O framework contém três protocolos predefinidos, e é possível a criação de novos através da extensão da classe pai do protocolo. Eles são o coração da eleição, sendo responsáveis, em grande parte, pela segurança do processo eleitoral. É aqui que é estabelecida a lógica da votação, como as cédulas serão recebidas, como serão armazenadas, quais e quando determinadas operações criptográficas devem ser realizadas, quais as autoridades de votação participam do processo, entre outros pontos chaves do protocolo. É necessário que cada eleição utilize um protocolo, pois ele especifica partes-chave do processo de votação que não estão definidas na própria eleição. Foi desenvolvido o protocolo mais simples possível, que não faz uso de nenhuma primitiva criptográfica, apenas recebendo a cédula do votante, a enviando para a urna e contabilizando o resultado. Isto é útil para eleições com baixo índice de criticidade e onde se deseja deixar a votação o mais simples possível.

- **Primitivas:** O framework já contém diversas primitivas criptográficas

implementadas, sendo que novas podem ser adicionadas ao sistema. São elas que ajudarão a compor um novo protocolo, sendo pontos-chave de sua implementação e responsáveis, junto com a lógica do protocolo, por garantir os requisitos de segurança. Muitas vezes as primitivas são estruturas bastante complexas cujo funcionamento é bastante crítico. Caso elas não funcionem corretamente, o protocolo também não funcionará. Um exemplo disto é a rede de mistura (ver apêndice A), que exige diversos servidores distribuídos, trabalhando de forma coordenada.

4.4 OPERAÇÃO

A máquina de estados definida na eleição possui 14 estados, que podem ser visualizados na Figura 8, sendo alguns deles são opcionais. A máquina de estados é importante para auxiliar no processo de votação, definindo com clareza quais etapas uma eleição deve passar ou não.

Toda votação deve seguir determinadas etapas para que o processo seja confiável. Cada uma delas permite apenas um conjunto limitados de ações que ficam restritas a alguns perfis de usuários. Sendo assim, toda eleição inicia com a etapa de configuração, onde são definidas as informações principais da eleição, tais como título, votantes, disputas, opções de voto, etc. Esta primeira etapa, que inicia todo o processo de votação, é de responsabilidade do administrador, que deve cadastrar todas as informações pertinentes da eleição. Em seguida vem o processo de auditoria. Esta etapa é opcional pois depende da existência de auditores. Caso haja auditores cadastrados, estes devem analisar as informações passadas e confirmar sua corretude. Caso haja alguma informação errada, o administrador deve corrigi-la e abrir novamente para auditoria. Este processo continua até que todos os auditores tenham aceito as configurações como corretas.

Finalizada esta fase, a eleição deve ser publicada, o que implica que suas informações não podem mais ser modificadas. Quando a eleição é publicada, seus dados ficam disponíveis para todos os votantes cadastrados naquela eleição. Isto permite que eles analisem informações relevantes, tais como os candidatos disponíveis e a data de início da votação.

Depois desta etapa, passa-se para o início do processo de votação. Chegada a data e hora do início de votação, a eleição fica disponível para os votantes emitirem seus votos. Esta etapa permite ao votante, basicamente, ob-

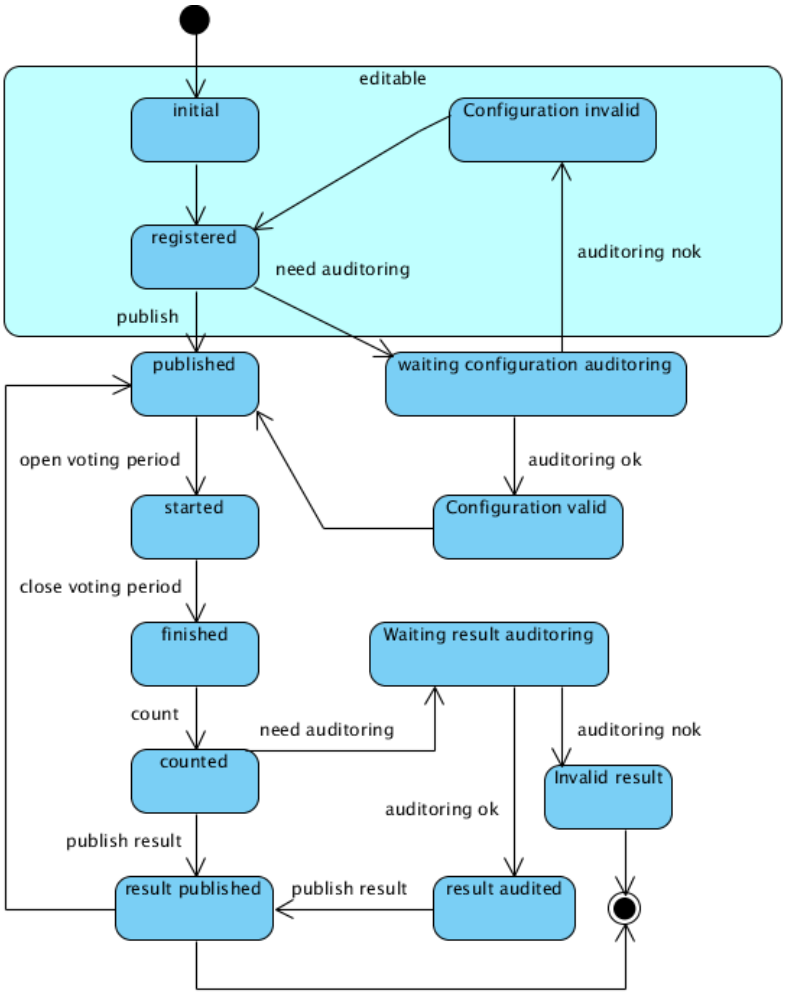

Figura 8: Máquina de estados da eleição

ter sua cédula, selecionar sua opção de voto, e enviá-la. Terminado o período de envio de votos, a eleição é encerrada e inicia o período de apuração, onde os votos são somados e o resultado é contabilizado. Aqui também é possível realizar a auditoria da apuração, fase esta que é opcional e dependente do protocolo. Nesta etapa, os auditores devem analisar as provas geradas durante a votação e averiguar se estão corretas. Cada protocolo gera provas específicas dependendo das primitivas utilizadas e do seu funcionamento. Um protocolo que faça uso de uma rede de mistura pode emitir provas de que a mistura foi honesta e de que a recifragem das cédulas foi correta. Já um protocolo que não faça uso de uma rede de mistura não pode emitir este tipo de prova. Caso exista esta etapa no processo de votação, após todos os auditores analisarem as provas, é concluído que a eleição foi honesta ou não. Caso tudo tenha ocorrido corretamente, o administrador tem permissão de divulgar o resultado da apuração. Caso conclua-se que houve alguma fraude ou erro, a eleição é cancelada, pois o resultado não é confiável. Considerando que o resultado pode ser divulgado, isto é feito através da *Bulletin Board*: um espaço de publicação confiável onde apenas pessoas autorizadas tem permissão de escrita, e onde a leitura é aberta para qualquer indivíduo. Com isto, o resultado da eleição é divulgado assegurando-se certas garantias, tais como a autenticidade da origem e do conteúdo das informações.

Passadas todas as etapas, não existem mais ações possíveis de serem executadas na eleição. Não é possível mexer em suas configurações ou abrir novamente para votação, salvo seja criada uma nova classe de eleição e sua máquina de estados seja alterada. Isto pode ser útil caso se deseje um comportamento diferente da eleição, como por exemplo, que existam diversos períodos de apuração dentro da mesma eleição.

4.5 MECANISMOS AUXILIARES DA ELEIÇÃO

O processo de votação depende de outros mecanismos, além da lógica da eleição, para auxiliar em todas as etapas da votação e garantir os requisitos de segurança. Por exemplo, é necessário garantir a identidade de votantes, pois mesmo que o protocolo de votação garanta que nenhum votante vote mais de uma vez, é necessário outro mecanismo para garantir a identidade destes usuários. São partes do sistema que devem ser confiáveis e seguir seus próprios requisitos. O mesmo sistema de autenticação poderá ser usado por diferentes protocolos, o que inclusive aumenta sua criticidade. Como mecanismos auxiliares do processo de votação podemos citar:

4.5.1 Autenticação e autorização

A autenticação é o processo que verifica a identidade dos usuários, garantido que o usuário apresente credenciais válidas e assegurando que a identidade do usuário. Já o processo de autorização garante que uma vez o usuário autenticado, ele terá acesso apenas aos recursos que estão disponíveis de acordo com o seu perfil. A autenticação pode se dar de diversas maneiras e em diversas etapas do processo de votação. O votante pode se autenticar através do uso de um login e de uma senha, através de um certificado digital válido, de um token ou de um identificador único que tenha sido informado ao usuário, entre outros. Isto permite uma vasta gama de formas de autenticação, de acordo com o que for mais conveniente para o protocolo. O framework é flexível para diferentes implementações, já disponibilizando a autenticação via login e senha.

Outra característica importante da autenticação é que ela também pode se dar em diferentes momentos da votação. Por exemplo, é possível que o sistema peça ao votante para se identificar antes de entrar no sistema, o que então permitiria ao usuário visualizar as eleições que tem acesso para só então selecionar aquela que deseja votar. Em outros casos, o sistema pode exigir que o votante se autentique apenas quando for enviar sua cédula, permitindo a qualquer pessoa obter uma cédula e selecionar uma opção de voto. Isto é utilizado no protocolo Helios, permitindo a qualquer pessoa, seja ela votante ou não, auditar a cédula da votação e comprovar a honestidade do sistema. Em outros casos, pode não ser necessário nenhum mecanismo de autenticação por parte dos votantes, ficando restrito apenas a outros perfis. Isto acontece no protocolo Code Sheets (HELBACH; SCHWENK, 2007), onde cada votante recebe uma cédula que usará na votação, e ao invés de provar sua identidade, entra apenas com o identificador da cédula recebida como forma de autenticação.

4.5.2 Gerenciamento de usuários

O processo eleitoral pode conter diferentes tipo de usuários, cada um com requisitos e obrigações diferentes. Isto permite uma melhor distribuição de tarefas e um melhor cumprimento de obrigações, além de dificultar a manipulação da eleição por parte de administradores ou outros perfis, pois em certos casos é necessário um consenso por parte dos envolvidos para a manipulação das informações. Por exemplo, com a existência de auditores em uma eleição, é muito mais complicado ao administrador inserir informações falsas durante o cadastro da eleição.

Os diferentes perfis contêm obrigações diferentes e, portanto, tem acesso a partes diferentes do sistema. O gerenciamento de usuários está intimamente relacionado com o mecanismo de autenticação, pois é este que controla quem terá acesso a quais páginas e analisa a identidade dos usuários, de forma que não seja possível forjar identidades, entrar no sistema sem uma credencial válida ou acessar partes a que não deveria ter acesso.

4.5.3 Gerenciamento de cédulas

Todas as eleições preveem o uso de cédulas, que são responsáveis por conter a estrutura das disputas e suas respectivas opções de voto. Isto traz inúmeras vantagens, pois permite ao votante obter estas cédulas e, em alguns casos, usá-las posteriormente ou realizar algumas operações sobre elas, tais como cifrá-las, assiná-las, blindá-las, entre outras.

A estrutura da cédula nem sempre é trivial, podendo ser bastante complexa dependendo de como são organizadas as disputas e suas opções de voto. Alguns casos, como disputas do tipo cargo, necessitam de toda uma hierarquia e relacionamento entre atributos, tais como partidos, coligações, chapas e outros dados do candidato, como número de votação, apelido, nome, etc.

Para facilitar sua criação, uma cédula é criada e as outras são copiadas da cédula original. Após a cópia, pode haver a inserção de dados específicos, como um identificador da cédula, distinguindo-a das demais.

4.5.4 Disputas e opções de voto

Toda eleição é constituída por disputas e suas respectivas opções de voto. As disputas dizem respeito a quais assuntos serão votados e quais opções de voto para cada assunto estarão disponíveis. O framework prevê disputas do tipo plebiscito e do tipo cargo, sendo as primeiras equivalentes a perguntas, tais como ”Você é a favor do desarmamento”ou ”Qual o melhor título para a matéria”. Disputas do tipo cargo são aquelas que possuem candidatos e toda a estrutura por trás deles, como partidos, coligações e chapas, além das informações próprias do candidato, tais como nome, apelido, número de votação, etc.
Fora disputas de cargo e plebiscito, o framework permite a criação de outros tipos disputas e opções de voto sem grande impacto no resto do sistema, sendo

possível reutilizar mecanismos como cédulas e urnas. O sistema também permite a criação de cédulas onde o votante tenha a oportunidade de selecionar mais de uma opção de voto por disputa, sendo possível estipular um determinado número de opções selecionáveis. Assim, o votante tem a possibilidade de selecionar não apenas a opção que julgue mais interessante, mas uma quantidade específica de opções. Também existem situações onde a eleição possibilite votos nulos ou em branco. Uma forma de tratar situações deste tipo é considerando ambos os tipos de voto como uma opção da disputa.

4.5.5 Urna

A urna é responsável por manter as cédulas recebidas de forma que apenas os responsáveis possam enviá-las e obtê-las, da mesma forma que as urnas físicas. É possível utilizar mais de uma urna por protocolo, como é o caso no protocolo Three Ballot (SANTIN; COSTA; MAZIERO, 2008) onde existem três urnas, cada uma de responsabilidade de uma pessoa diferente, dificultando a obtenção das cédulas de forma indevida. Alguns protocolos exigem que toda cédula seja cifrada com a chave pública da urna, tornando mais difícil a leitura das cédulas por pessoas não autorizadas. Além disto, o conteúdo da urna só deve ser acessível ao final da eleição, para impedir que sejam divulgados resultados parciais, o que comprometeria o andamento da eleição, influenciando os votantes.
A urna é uma estrutura a parte tendo seu próprio banco de dados, separado da eleição, podendo inclusive ficar em outro computador.

4.6 MODELAGEM

A modelagem do framework foi dividida em várias etapas. Inicialmente foi necessária uma análise de domínio do problema através da seleção e estudo dos protocolos julgados mais pertinentes, pois não se tinha certeza da possibilidade da implementação de um framework para o problema encontrado. Dentre os protocolos selecionados, foram verificados aspectos presentes em grande partes deles para definir os pontos mais relevantes dos protocolos e dos sistemas que os utilizam. Com isto foi possível ver as similaridades e os pontos essenciais que compõem um sistema de votação. Este conhecimento foi então organizado através da identificação de quais objetos, operações e relacionamentos eram importantes para a modelagem de um framework de votação.

Depois desta etapa, foi iniciada a modelagem. Foram pensados quais os aspectos principais do framework, tais como, protocolo, eleição e quais gerenciadores e interfaces seriam necessários. Foram analisadas quais classes o framework deveria possuir e como seria possível utilizá-lo de forma que não fosse necessária a criação de muitas classes novas quando se quisesse fazer uma alteração em sua estrutura. Ou seja, um framework que não fosse nem caixa branca, nem caixa preta, mas sim caixa cinza (FAYAD; SCHMIDT; JOHNSON, 1999). Isto aumentaria sua flexibilidade sem que fosse necessário muito esforço por parte do desenvolvedor. A intenção era um framework que permitisse um alto grau de reúso, sem torná-lo inflexível.

4.6.1 Diagrama de casos de uso

A modelagem iniciou com a identificação das funcionalidades do framework: o que ele deveria prover aos seus diversos usuários. Levantadas essas funcionalidades, foi possível gerar um diagrama de casos de uso identificando quais seriam os tipos de usuários do sistema e o que cada um seria responsável. Este diagrama pode ser visto nas Figuras 6 e 7. Esta etapa foi importante para o prosseguimento da modelagem, pois seria a base para todo o desenvolvimento subsequente, pois nesta etapa foi delimitado o que o framework abarcaria e o que estaria fora do escopo de desenvolvimento.

4.6.2 Diagramas de classe

Após o diagrama de casos de uso, foram modelados os diagramas de classe contendo aquelas que inicialmente julgou-se necessárias, como, por exemplo, os diagramas 15, 16, 17, 18, 19, 20, 21. As primeiras classes identificadas foram as mais centrais, tais como as classes de eleição e de cédula. Num primeiro momento foram modeladas apenas as classes, com alguns atributos, mas sem a identificação dos métodos. Através da modelagem verificou-se ser necessário tomar algumas decisões, tais como a separação da eleição e do protocolo, de forma que ambas não se tornassem dependentes, o que afetaria o funcionamento do framework, sua usabilidade e todo a modelagem subsequente.

A produção dos diagramas de classe se deu junto com a modelagem de outros diagramas, de forma que um complementasse o outro. Esta interação foi cíclica, implicando que quando um diagrama fosse alterado, os outros

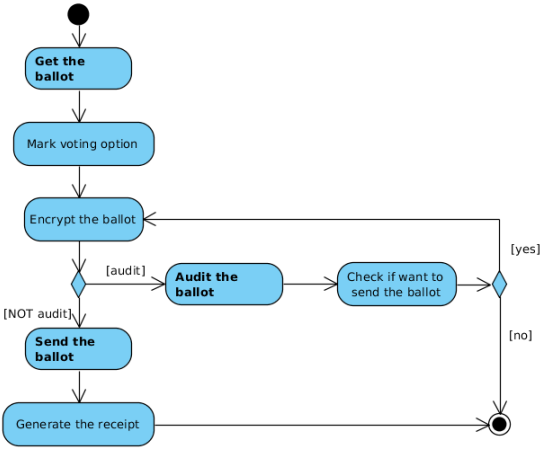

Figura 9: Diagrama de atividade relativo ao processo de votar

também sofressem modificações, levando ao amadurecimento da estrutura do framework.

4.6.3 Diagramas de atividade

Os diagramas de atividade ajudaram a identificar o fluxo de controle de determinadas atividades do sistema. Foi modelado um diagrama de atividade para cada caso de uso, possibilitando a identificação de quais classes estariam envolvidas em cada caso, sendo adicionadas aos diagramas de classe caso ainda não tivessem sido previstas. Foram úteis na compreensão das funcionalidades do sistema, permitindo descrevê-las de modo mais detalhado em comparação com os casos de uso.

Um dos diagramas de atividade centrais da modelagem é o relativo ao processo de votar, separado em duas grandes etapas: obter a cédula e enviar a cédula. O diagrama de atividade que engloba todo o processo de votação pode ser visto através da Figura 9 e as duas etapas descritas em maiores detalhes podem ser vistas através das Figuras 10 e 11, respectivamente o diagrama de atividade de obtenção da cédula, e o diagrama de atividade de envio da cédula. Outros diagramas de atividade podem ser visualizados no Apêndice C.

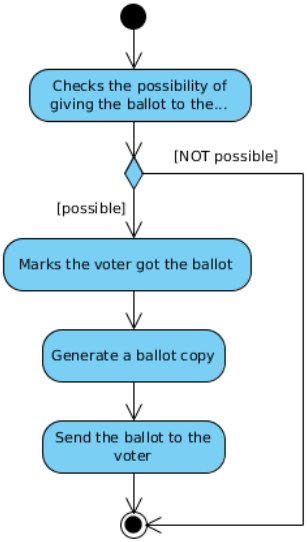

Figura 10: Diagrama de atividade relativo ao processo de obtenção da cédula

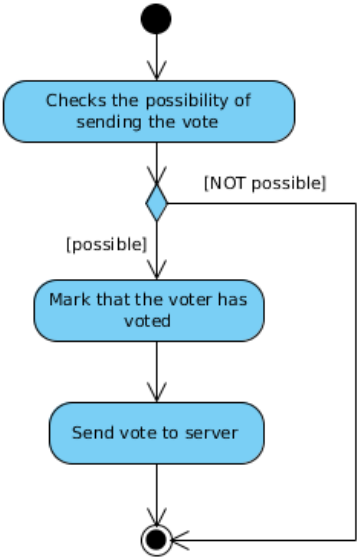

Figura 11: Diagrama de atividade relativo ao processo de envio da cédula

4.6.4 Diagramas de sequência

Os diagramas de sequência ajudaram na identificação de quais métodos cada classe deveria ter. Eles são importantes para definir a troca de mensagens entre os objetos, ajudando a refinar a modelagem em um nível mais específico de detalhamento. Foram criados diagramas de sequência para cada uma das principais atividades do sistema, como pode ser visto através da Figura 12. Este diagrama especificamente trata sobre o processo de votação, sendo mais bem detalhado em outros dois diagramas: o de obtenção da cédula (Figura 13) e de envio da cédula (Figura 14). Mais diagramas de sequência podem ser visualizados no Apêndice B.

4.7 DECISÕES REALIZADAS NA MODELAGEM

Durante a modelagem foi necessário tomar algumas decisões sobre as classes e suas lógicas de interação, pois isto afetaria os tipos de protocolos e sistemas de votação que poderiam ser desenvolvidos com o framework. As principais decisões tomadas foram as seguintes:

4.7.1 Protocolo e eleição

Resolveu-se separar o protocolo da eleição, e considerá-los duas entidades distintas, como pode ser visto através da Figura 15. A eleição contém a máquina de estados da votação, ou seja, ela que conhece as etapas do processo de votação. Ela sabe se é hora de enviar os votos, ou se é hora de realizar a apuração, mas não detém conhecimento de como essas ações são realizadas. Ela não sabe como os votos serão obtidos, nem como serão contabilizados. Isto é responsabilidade do protocolo, que detém a inteligência do processo de votação. É ele quem interage com as primitivas criptográficas, e sabe como e quando são utilizadas. Os protocolos podem ser considerados o coração da eleição, existindo inúmeras implementações possíveis sendo apenas uma utilizada por eleição. Já a classe de eleição é normalmente a mesma em todas as eleições, pois a maior parte das votações utiliza a mesma máquina de estados definida no framework.

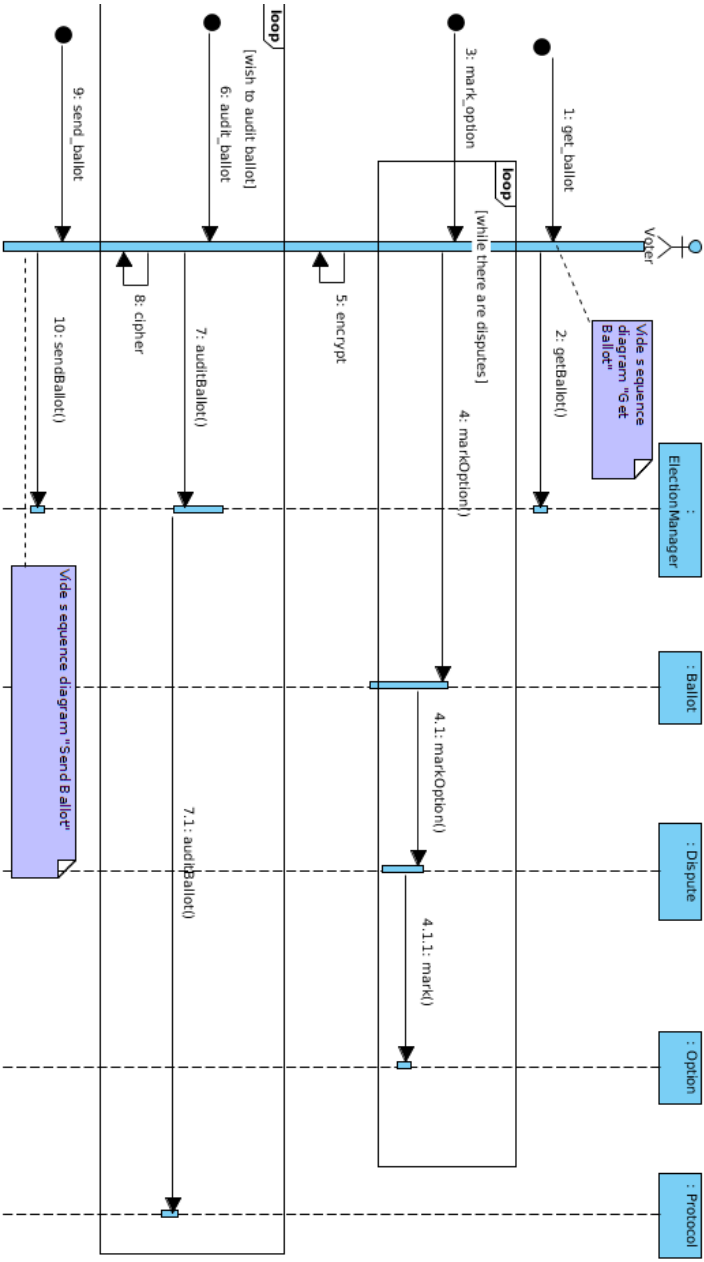

Figura 12: Diagrama de sequência relativo ao processo de votar

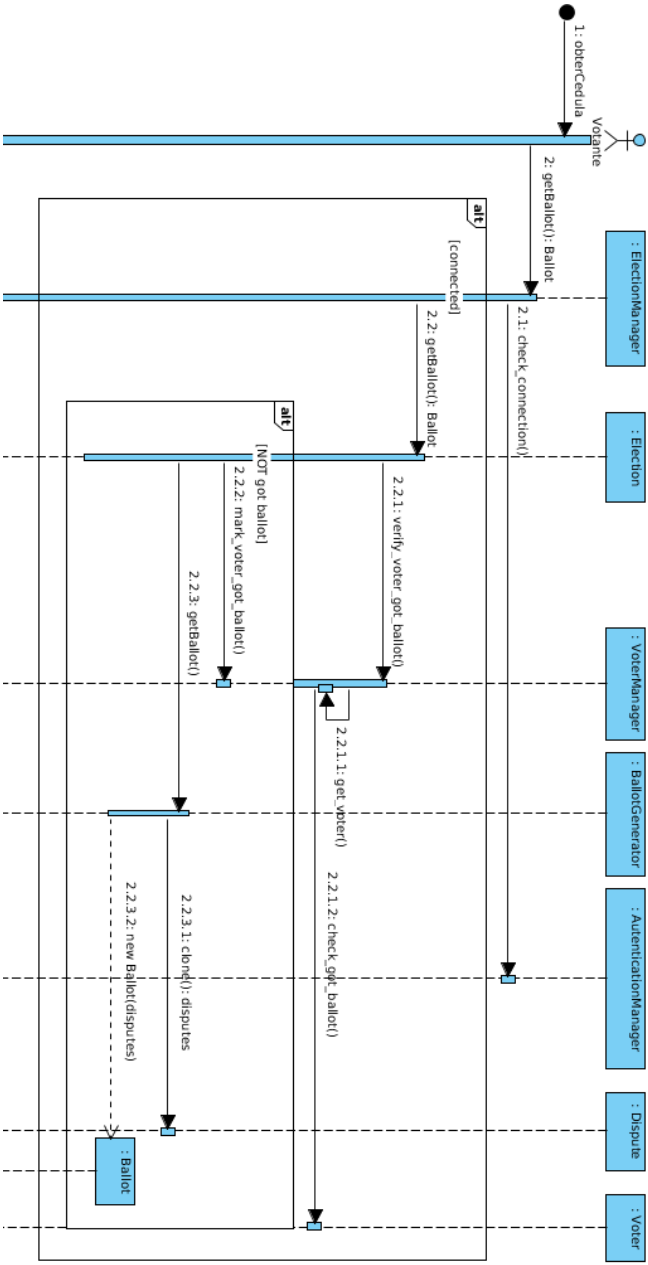

Figura 13: Diagrama de sequência relativo ao processo de obtenção da cédula

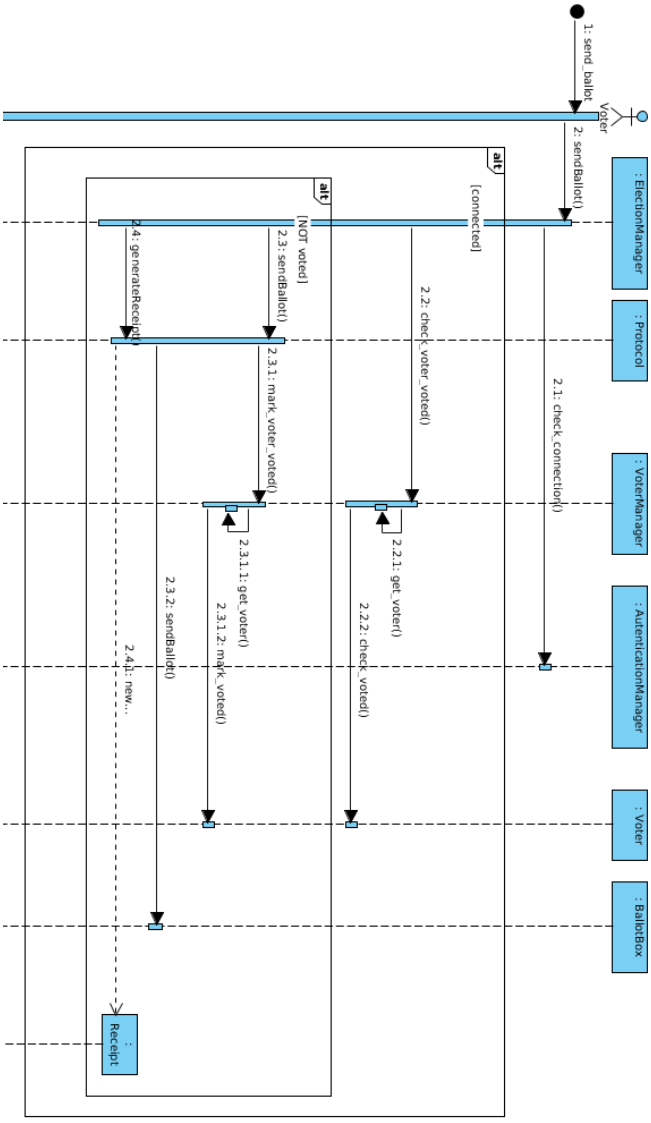

Figura 14: Diagrama de sequência relativo ao processo de envio da cédula

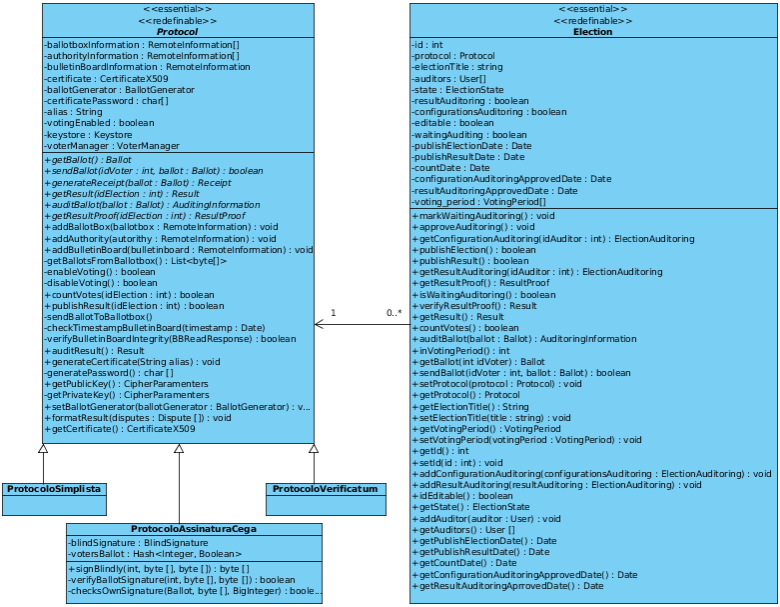

Figura 15: Diagrama de classe mostrando a relação entre a eleição e o protocolo

4.7.2 Cédulas

Constatou-se que vários protocolos fazem uso de cédulas de votação, normalmente utilizando-as para apresentar aos votantes as disputas e opções disponíveis. Apesar de nem todos a utilizarem, é possível tratar a estrutura da cédula de forma variada, permitindo que os protocolos a utilizem adequando-a como for mais conveniente. Por exemplo, o protocolo Code Sheets (HELBACH; SCHWENK, 2007) faz uso de cartelas que são enviadas aos votantes antes do início da votação, o que vai contra a lógica da maioria dos protocolos, que permitem que o votante obtenha a cédula apenas quando a eleição abre para votação. Além disto, a estrutura da cartela é relativamente diferente da estrutura das cédulas dos outros protocolos, contendo mais campos que o normal. Apesar destas diferenças é possível estender a classe cédula e implementá-la de acordo com os requisitos do protocolo Code Sheets, de forma que ela seja reaproveitada em um protocolo com características bastante diferentes.

O framework exige que todo protocolo faça uso da estrutura da cédula. Mesmo que em sua concepção ele não utilize cédulas de votação, é necessário adequar o protocolo à estrutura do framework. Outra característica importante do uso da cédula, é que existe a possibilidade de exportá-la como um arquivo XML, arquivo texto, ou utilizando outra estrutura, o que permite a realização de operações criptográficas com a cédula, tais como cifragem e assinatura. Além disso, tendo a cédula como uma estrutura exportável, é possível separar a votação em duas etapas: a de obtenção e de envio da cédula.

4.7.3 Separação entre obter a cédula e enviá-la

A primeira ideia que se tem quando se pensa em uma votação, é que ela é uma etapa monolítica, ou seja, o votante seleciona sua opção de voto e a envia, tudo ao mesmo tempo, sem divisões. Mas quando separamos isto em mais de uma etapa, ganhamos algumas vantagens como flexibilizar o processo, pois é possível ao votante obter a cédula em um momento, e só em outro enviá-la. Isto também permite ao framework um maior controle de cada etapa, permitindo, por exemplo, que o votante obtenha a cédula inúmeras vezes, mas a envie apenas uma. Ou que obtenha a cédula apenas uma vez, e que possa selecionar sua opção de voto e enviá-la ao sistema inúmeras vezes. Também permite que as duas etapas ocorram com um grande intervalo de tempo, por exemplo dias, permitindo ao votante guardar sua cédula durante este período.

4.7.4 Usuários e perfis

Devido aos diversos papéis que um usuário pode assumir nos sistemas de votação digital estudados, escolheu-se uma abordagem onde fosse possível cadastrar um usuário e associar papéis a sua conta. Dessa forma é possível a um mesmo usuário assumir diversos papéis: ser votante em uma eleição, administrador de outra e auditor de uma terceira, permitindo que em todos os casos a mesma conta seja reutilizada. Para isto, existe uma superclasse de perfis que está associada à conta do usuário, como mostra a Figura 16. É possível, assim, que um usuário não tenha nenhum perfil associado à sua conta, não podendo efetuar nenhuma ação no sistema a não ser seu login, ou que tenha vários, e possa realizar diferentes ações dependendo de qual perfil selecionar.

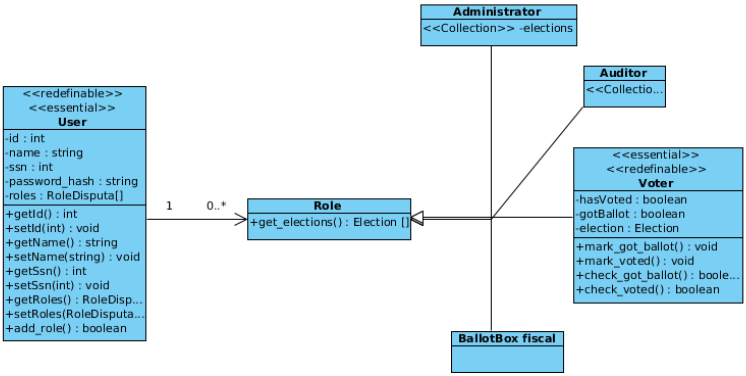

Figura 16: Diagrama de classes mostrando a relação entre a conta do usuário e seus perfis

4.7.5 Gerenciadores

Existem diversos mecanismos dentro do framework responsáveis por gerenciar partes importantes do processo de votação, como, por exemplo, a construção das cédulas e a autenticação de usuários. Para facilitar o controle de determinados atividades, foram criadas classes gerenciadores. Elas permitem concentrar a comunicação com determinadas classes em um único lugar, de forma que baste o acesso a uma única classe gerenciadora para se ter controle de toda uma gama de ações sobre algum aspecto específico do framework.

Existem no framework oito classes gerenciadoras: para eleição, para o protocolo, para as credenciais, para as cédulas, para autenticação, para os votantes, para os usuários e para os perfis (Figura 17). Cada uma delas dará controle a determinadas funcionalidades do framework. Por exemplo, a classe gerenciadora da eleição dá acesso a todas as possíveis funcionalidades de uma eleição: adicionar título, obter cédula, enviar cédula, contar votos, publicar resultado, dentre outros, como pode ser visto na Figura 18. Todas estas atividades dizem respeito à eleição, mesmo que não possam ser acessíveis a todos os perfis, e é justamente por isto que existem as interfaces.

Figura 17: Diagrama de classes mostrando todas as classes gerenciadoras e as interfaces que tem acesso a cada uma delas

4.7.6 Interfaces

Como pode ser visto na Figura 18, o gerenciador de eleição lida com atividades que pertencem a diferentes grupos de usuários. Ali se encontram as ações de responsabilidade do administrador (adicionar período de votação, configurar protocolo, contabilizar os votos, etc), as ações dos votantes (obter cédula, enviar cédula, etc) e dos demais perfis. Com isto, é necessária outra camada que restrinja esse acesso, provendo apenas os métodos que cada perfil deverá acessar. Esta camada é a camada de interface, que faz a ligação entre as telas dos usuários e os gerenciadores, que fazem parte do core.

Para cada perfil é necessário uma interface, totalizando quatro dessas classes implementadas, sendo que é possível criar novas, caso sejam criados novos perfis. Todas elas são extensões da classe UserInterface, dando opção ao perfil que dizem respeito de se autenticar no sistema e de listar quais eleições têm acesso. São as interfaces que sabem exatamente o que cada perfil tem liberdade de acessar, e são elas que conhecem as camadas mais internas do core. Isto facilita o trabalho de desenvolvimento das telas, pois permite aos desenvolvedores conhecerem apenas as interfaces, e não todo o funcionamento do framework.

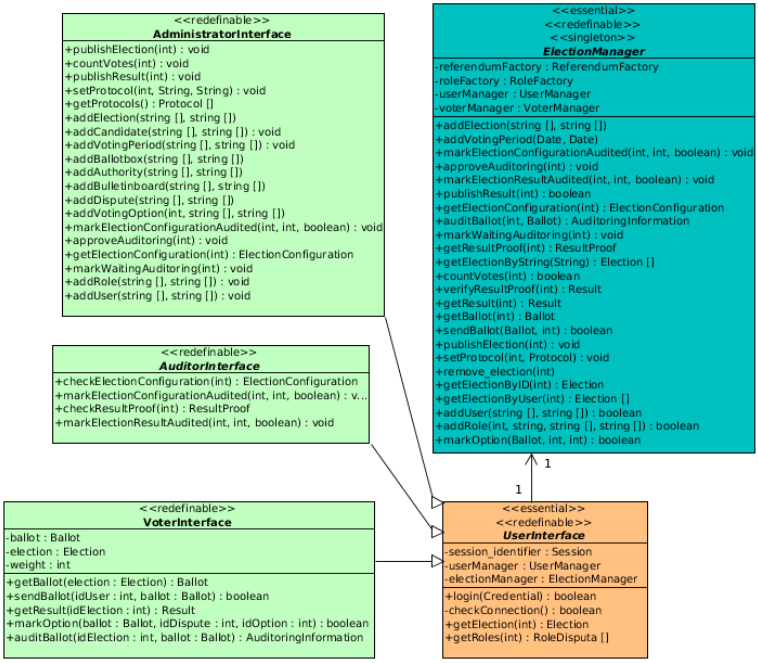

Figura 18: Diagrama de classes mostrando a classe gerenciadora da eleição

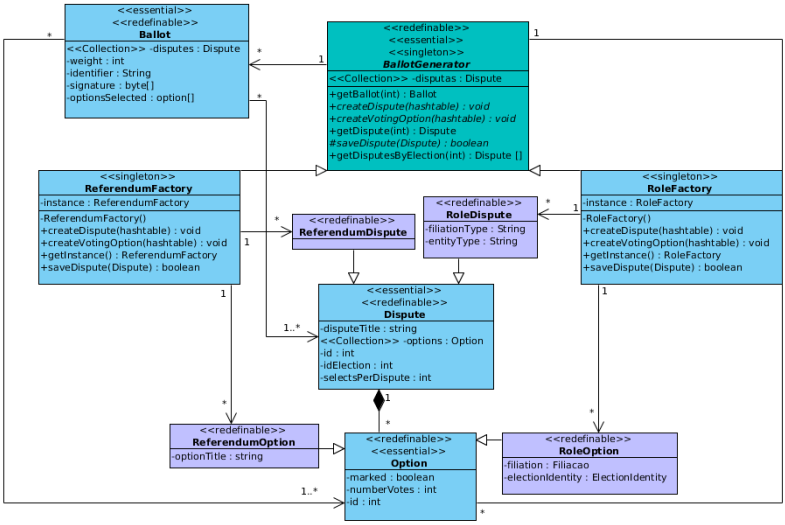

Figura 19: Diagrama de classes mostrando a utilização do padrão factory

4.7.7 Padrões de Projeto

Foram utilizados diversos padrões de projeto no framework de forma a solucionar de forma eficiente alguns problemas encontrados. Abaixo seguem os padrões de projeto empregados:

4.7.7.1 Factory

Para o gerenciamento das disputas e opções de voto foi utilizado o padrão factory, que permite delegar a instanciação das classes para as subclasses. Isto é útil na hora da geração das cédulas, pois a classe que controla sua geração não precisa saber com qual factory está lidando, sendo essa informação delegada a quem realmente define qual tipo de disputas e opções serão utilizadas na eleição.

Como podemos ver na Figura 19, a classe BallotGenerator é responsável por gerar as cédulas, que podem ser compostas por disputas do tipo Referendum ou do tipo Role. Esta classe é abstrata, tendo duas possíveis classes instanciáveis: ReferendumFactory e RoleFactory, que tem o conhecimento suficiente para gerenciar os tipos de disputas pelas quais são responsáveis.

Isto é útil pois algumas disputas e suas respectivas opções podem ser relativamente complexas, como no caso de opções do tipo cargo, que podem ter uma estrutura bastante detalhada entre chapas, partidos e candidatos, sendo que esta complexidade fica contida na fábrica responsável por sua instanciação. O benefício desta abordagem é que as classes que lidam diretamente com as fábricas não sabem especificamente com qual fábrica estão lidando, agindo de forma padronizada com elas. Isto é útil caso se deseje adicionar outros tipos de disputas e opções de voto, não sendo necessário alterar as classes que lidam com essas classes.

De forma geral, as classes BallotGenerator, ReferendumFactory e RoleFactory formam a estrutura principal do padrão de projeto Factory, responsáveis por gerar toda a estrutura de disputas. No caso, a classe que sabe com qual factory está lidando é o ElectionManager, sendo ela a responsável por instanciar a fábrica correta, que será posteriormente utilizada pelas outras classes sob a forma do BallotGenerator, e não a fábrica concreta.

4.7.7.2 Prototype

Este padrão foi utilizado na geração da cédula, pois as opções de voto podem ser estruturas complexas e consequentemente ter alto custo criá-las do zero toda vez que é preciso emitir uma nova cédula, o que ocorre frequentemente. Por isto, apenas uma dessas estruturas é criada e todas as outras são cópias da original.

4.7.7.3 Singleton

Todos os gerenciadores seguem o padrão singleton, pois é necessário que exista apenas uma instância de cada um deles. Sendo assim, toda vez que é necessário obter um gerenciador específico, a primeira coisa a verificar é se ele já está criado, em caso afirmativo, ele é simplesmente recuperado ao invés de ser novamente instanciado. Isto garante que exista apenas uma cópia por vez de cada gerenciador.

4.7.7.4 Composite

A estrutura de autenticação permite que o usuário se autentique de diferentes maneiras, através de login e senha, tokens de identificação, certifica-

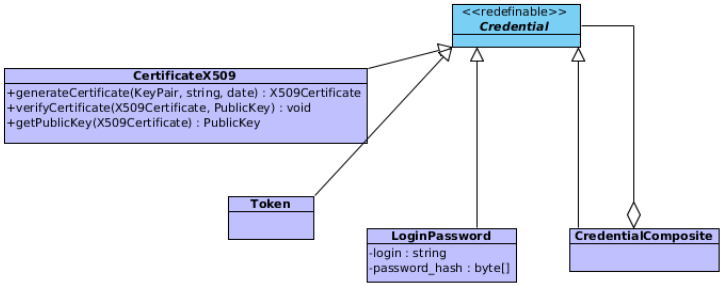

Figura 20: Diagrama de classes mostrando a utilização do padrão composite

dos digitais ou outra forma que o administrador ache desejável. Para permitir que diferentes formas de autenticação fossem tratadas de forma similar pelo framework, foi utilizado o padrão de projeto Composite, como mostrado na Figura 20. Com isto é possível utilizar simultaneamente várias mecanismos de autenticação sem aumentar a complexidade na forma de tratá-los.

4.8 IMPLEMENTAÇÃO

A linguagem utilizada para a implementação do framework foi o Java 7, por existirem várias bibliotecas úteis ao desenvolvimento, como será mostrado adiante.

A implementação do framework se deu em diferentes etapas, pois algumas partes do sistema exigiam que outras partes já estivessem implementadas. Por exemplo, para desenvolver a persistência, era necessário já ter implementado as entidades que seriam persistidas, assim como os gerenciadores que tem conhecimento de quando persistir cada entidade. Além disto, toda classe cuja lógica fosse complexa esteve associada a um teste unitário, de forma a avaliar seu funcionamento correto. Também foram realizados testes de integração para verificar se o sistema como um todo funcionava adequadamente.

4.8.1 Etapas da implementação

A implementação do framework foi iniciada pelo core, pois é ali que se concentram as principais classes do sistema, aquelas que comprometeriam severamente o seu funcionamento caso não fossem corretamente desenvolvidas. As primeiras classes do core a serem implementadas foram as entidades, classes como os perfis de usuário, disputas, opções de voto e a cédula, todas aquelas que deveriam ser persistidas no banco. Em seguida foram definidas as classes que lidavam com estas entidades: os gerenciadores. Os gerenciadores tem o conhecimento de como lidar com estas classes, quando é necessário persisti-las e quais ações são cabíveis com cada uma delas, como, por exemplo, como montar a cédula, como cadastrar uma eleição ou como adicionar um usuário. Em seguida foram desenvolvidas as classes responsáveis pela persistência, permitindo persistir, atualizar e remover as entidades do banco de dados.

Após o desenvolvimento do core, foram implementados três protocolos de votação para analisar quão adequado o framework se mostrava para a implementação de cada um deles. Com isto, foi possível verificar quais alterações seriam desejáveis no framework, partindo das necessidades dos próprios protocolos. Dessa forma foi possível analisar com maior clareza como deveria ser a modelagem das cédulas, o que deveria ser alterado na máquina de estados da eleição e a melhor maneira de modelar a classe de protocolos.

Após o desenvolvimento das entidades, dos gerenciadores, dos protocolos e das classes responsáveis pela persistência, foram desenvolvidas as interfaces. Estas classes são as que definem quais métodos serão acessíveis para cada usuário em cada tela, e fazem a comunicação entre a parte gráfica do sistema e o core. Finalmente foram implementadas as telas, permitindo testar o sistema através de uma interface gráfica e verificar a adequabilidade das interfaces. E por último, foi implementado o mecanismo de autenticação, que permite aos diferentes perfis se identificarem e terem acesso ao sistema. Com isto foi possível simular eleições como as que ocorreriam em situações reais.

4.8.2 Persistência

Para implementar a persistência escolhemos utilizar o framework Hibernate (COMMUNITY, 2013) por sua facilidade de utilização, ampla docu-

mentação e uma comunidade bastante ativa, que oferece suporte e acesso a muita informação e exemplos de uso. Outro fator importante foi o fato do Hibernate suportar a linguagem Java que foi a utilizada no framework.

Como gerenciador de banco de dados utilizamos o PostgreSQL (POSTGRESQL, 2013) por também ter uma comunidade bastante ativa, onde é possível obter suporte para problemas e dúvidas surgidas durante o desenvolvimento, e também por sua facilidade de utilização, sendo ambas as tecnologias facilmente integradas.

Para a implementação da persistência foi necessário adequar as classes de entidade de acordo com o Hibernate, colocando as anotações adequadas e definindo quais atributos seriam persistidos.

4.8.3 Primitivas

Diversos protocolos utilizam primitivas criptográficas em sua lógica de funcionamento para garantir que certos requisitos de segurança sejam cumpridos. Estas primitivas podem ser bastante complexas exigindo um alto rigor técnico no seu desenvolvimento, uma vez que são partes críticas do sistema.

No desenvolvimento do framework foram utilizadas bibliotecas que disponibilizam algumas das principais primitivas criptográficas encontradas nos protocolos analisados. Todas as primitivas possuem uma interface desenvolvida de acordo com as exigências do framework, possibilitando que caso o utilizador queira trocar as bibliotecas, os protocolos continuem funcionando normalmente, sem qualquer alteração na forma de utilizar estas primitivas, como pode ser visto através da Figura 21.

As bibliotecas utilizadas foram o Bouncy Castle (CASTLE, 2004) para a geração dos certificados digitais, função hash, criptografia simétrica e assimétrica, assinatura cega e elGamal. Também foi utilizada a biblioteca Thep (THEP, 2013) para a utilização do paillier. Além disto, para o desenvolvimento da rede de mistura foi utilizado o Verificatum (WIKSTRöM, 2013), assim como também o Bouncy Castle. Para uma leitura mais detalhada sobre cada primitiva, vide Apêndice A.

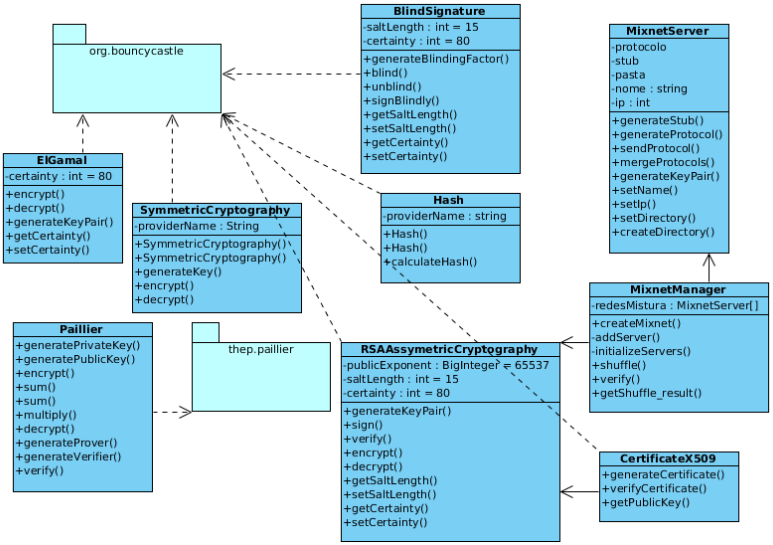

Figura 21: Diagrama de classes mostrando as primitivas e as bibliotecas utilizadas

4.8.4 Autenticação

Para a autenticação foi utilizado o Apache Tomcat 7 (TOMCAT, 2013), que permite gerenciar o acesso de usuário através do JDBCRealm. Com isto é possível configurar quais perfis de usuários terão acesso a quais páginas do sistema, compartilhando o mesmo banco de dados utilizado pelo framework.

Para configurar esta forma de autenticação foi necessário criar uma tabela contendo o nome do usuário e o seu perfil. Além da configuração necessária no banco de dados, também foi necessário configurar o arquivo web.xml dentro do framework, especificando quais páginas seriam acessíveis para cada um dos perfis e quais páginas seriam acessíveis sem autenticação. Uma das grandes facilidades dessa forma de autenticação pelo Tomcat é que ele gerencia todo o controle de acesso, sendo necessário apenas implementar uma página de autenticação segundo as especificações exigidas pelo Tomcat.

4.8.5 Web Service

Foi necessário utilizar a tecnologia de web services para fazer a comunicação entre algumas partes do sistema, como, por exemplo, entre os vários servidores de uma rede de mistura, fazendo com que cada nodo da rede rodasse em seu próprio web service. Para isto foi utilizado o Apache Axis (AXIS, 2006), que é um framework escrito em Java que auxilia na construção de web services no padrão SOAP.

4.8.6 Servidores

Alguns sistemas de votação digital utilizam mais de uma autoridade por eleição, para distribuir a inteligência e assim dificultar que o sistema seja corrompido. É desejável que estas autoridades fiquem em computadores separados, pois caso um seja atacado, não necessariamente os outros serão. Dessa forma, o framework prevê a existência de diversas autoridades em máquinas diferentes se comunicando entre si.

4.9 TIPOS DE ELEIÇÃO

O framework provê a estrutura para dois tipos de eleição: do tipo cargo e do tipo plebiscito. Uma eleição é configurada para ser apenas de um tipo, exigindo que suas disputas e opções de voto também o sejam. Por exemplo, uma eleição de cargo contém apenas disputas do tipo cargo e consequentemente suas opções de voto seguem o mesmo tipo.

A classe disputa é abstrata e contém como atributos: o título da disputa, as opções relativas a esta disputa, o número de opções que podem ser selecionadas e a eleição a que ela pertence, pois cada disputa é específica de uma determinada eleição. O número de opção selecionáveis é importante em situações onde uma determinada pergunta permite múltiplas respostas, permitindo ao votante selecionar mais de uma opção por vez. Como a classe Disputa é abstrata, ela necessita ser estendida. No caso, existem duas classes concretas que a estendem: a ReferendumDisputa e a RoleDispute, ambas podem ser vistas através da Figura 22. Cada uma das classes está associada a uma fábrica, responsável pela sua criação. Como o padrão de projeto recomenda, cada uma dessas fábricas contém uma única instância, seguindo o padrão de projeto Singleton (AL., 1994). Estas duas fábricas tem o conhecimento necessário para construir as disputas pelas quais são responsáveis, de

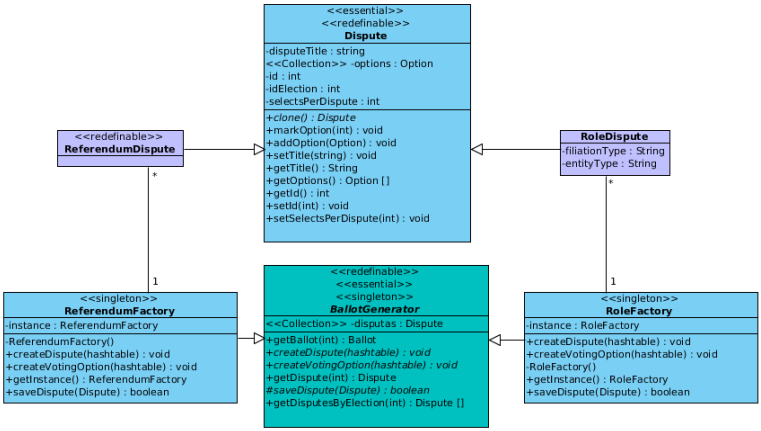

Figura 22: Estrutura e relacionamento das disputas

forma a diminuir a inteligência exigida no processo de criação de disputas.

A classe Option também é abstrata, sendo estendida pelas classes ReferendumOption e RoleOption. Como atributos ela contém apenas um sinalizador avisando se está ou não selecionada, para marcar se ela foi selecionada pelo votante e um atributo chamado de numberVotes importante na hora de contabilizar os votos.

Disputas do tipo plebiscito são bastante simples, não exigindo muitos atributos novos, como pode ser visto na Figura 23. A classe ReferendumDispute apenas sobrescreve o método clone exigido pelo padrão Prototype e a classe ReferendumOption cria o atributo para o título da opção, que no caso é o nome da opção de voto. Disputas do tipo cargo são bem mais complexas, exigindo uma estrutura mais sofisticada que o plebiscito. Elas contêm as seguintes classes:

- Candidato (Candidate): esta classe trata sobre o candidato e seus dados pessoais, aqueles que não variam de acordo com a eleição: seu nome, CPF (representado pelo campo SSN - Social Security number) e sua foto.

- Identidade eleitoral (ElectionIdentity): diz respeito à identidade do candidato, se ele está concorrendo sozinho ou se faz parte de uma chapa.

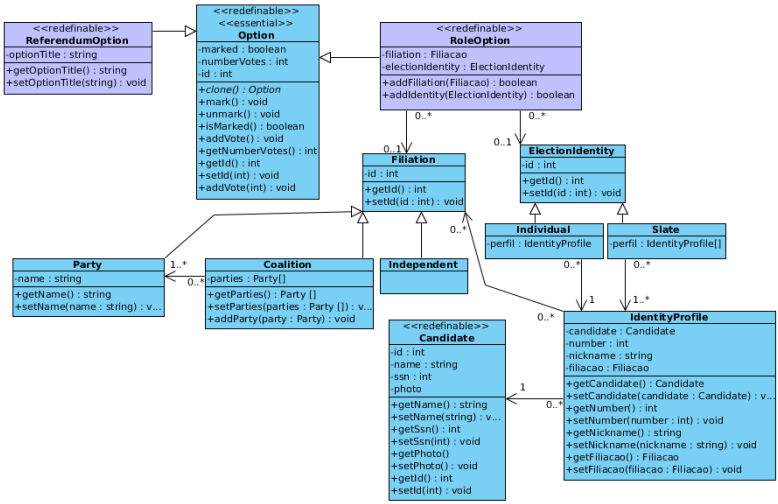

Figura 23: Estrutura e relacionamento das opções

Esta classe é abstrata e tem como classes concretas as classes Individual e Slate (chapa), sendo que a primeira contém apenas um candidato, no caso o IdentityProfile do candidato, e a segunda contém um conjunto destas classes.

- Perfil de identidade (IdentityProfile): esta classe é responsável pelo perfil do candidato. Uma mesma pessoa pode concorrer em eleições diferentes com perfis diferentes, ou seja, concorrendo por outros partidos, conhecido por outros apelidos ou utilizando números de votação diferentes. Estas informações que são transitórias são configuradas nesta classe, que também contém uma ligação com a classe candidato.

- Filiação (filiation): a filiação trata sobre a relação do votante com o partido. Se existe uma coligação (Coalition) entre determinados partidos, se ele está concorrendo de forma independente (Independent) ou se a relação dele é com apenas um partido (Party).

Toda esta estrutura de cargos é necessária para sabermos quem é o candidato, sua vinculação com o partido e a forma que está concorrendo. Por ser uma estrutura complexa, foi utilizado, como dito anteriormente, o padrão prototype, exigindo a criação do método clone na classe Option. Este método trata sobre como fazer cópias de toda esta estrutura.

4.10 TELAS

As telas são canais de comunicação importantes entre os usuários e o sistema. O framework contém a implementação das principais telas que serão usadas nos sistemas de votação desenvolvidos, além de permitir aos desenvolvedores implementar suas próprias telas quando necessário. As telas estão contidas dentro de um módulo de forma a manter um maior desacoplamento dos sistemas, permitindo que sejam alteradas sem que isso impacte consideravelmente o resto do framework.

Para fazer a ligação entre as telas e os sistemas de votação, existe uma camada intermediária responsável por fazer a comunicação entre os métodos do framework e a parte gráfica do sistema. Esta camada conhece e restringe quais métodos cada tela terá acesso, auxiliando na comunicação e no controle de acesso dos usuários.

As telas mostradas a seguir são as utilizadas pelos três protocolos implementados. Caso o protocolo utilizado exija um comportamento diferente do usuário, como será mostrado no capítulo seguinte, é possível implementar um conjunto diferente de telas e adicioná-las ao sistema.

4.10.1 Tela de autenticação

Para o usuário ter acesso ao sistema ele precisa primeiro se autenticar via login e senha. A página inicial do sistema pode ser vista na Figura 24 e é usada por todos os perfis do sistema, não havendo distinção entre a página de autenticação do administrador, do votante, ou dos outros perfis.

4.10.2 Tela principal do sistema

Assim que o usuário se autentica no sistema, ele é redirecionado para a tela principal, que possibilita visualizar todas as eleições que ele tem acesso. Dependendo do perfil selecionado, o usuário terá acesso a diferentes funcionalidades. Através da Figura 25 é possível visualizar a tela principal de um usuário cujo perfil é de votante, o que possibilita que ele verifique o resultado de uma eleição passada, e que vote em uma eleição que está atualmente aberta. Caso fosse o perfil de um administrador logado, este usuário teria a opção de cadastrar eleições, publicá-las e outras ações específicas deste perfil.

Figura 24: Tela de autenticação dos usuários

4.10.3 Cadastro de eleições

Uma das telas principais do sistema é a responsável pelo cadastro das eleições, utilizada pelo administrador. Ela contém todas as informações necessárias para o cadastro de uma eleição completa, separada em diferentes etapas, como pode ser visto na Figura 26.

No primeiro passo, o administrador entra com as informações básicas da eleição, tais como seu título, data de início e término da eleição, o protocolo utilizado e se é necessário auditoria das configurações. Após esta etapa, é possível cadastrar as disputas e suas opções de voto, outros administradores, votantes e, caso hajam, auditores. Na Figura 27 podemos ver a tela que permite ao administrador escolher entre adicionar novas opções de determinada disputa ou editar as já existentes. Quando todas as informações estiverem cadastradas, é mostrada uma tela com todas as configurações da eleição, e um botão de confirmação. Caso a eleição necessite que suas configurações sejam auditadas é necessário esperar a aprovação de todos os auditores cadastrados para que suas informações sejam disponibilizadas aos votantes. Caso não haja necessidade, o administrador pode disponibilizadas imediatamente publicando a eleição. Quando a eleição é publicada nenhum de seus dados pode mais ser alterado.

Figura 25: Tela principal do sistema

4.10.4 Auditoria

Para eleições configuradas para passarem pelo processo de auditoria, é necessário que cada um dos auditores verifique se as informações cadastradas da eleição estão corretas, como visto na Figura 28, e caso haja algum erro, devem informar isso ao responsável pelo seu cadastro através da própria página de auditoria, como visto na Figura 29.

4.10.5 Processo de votação

Quando a eleição se encontra dentro do período de votação, ela é disponibilizada aos votantes permitindo que estes emitam seus votos. A primeira tela da votação apresenta todas as disputas e suas respectivas opções de voto, possibilitando ao votante escolher as opções que julgar mais adequadas, como pode ser visto na Figura 30.

Após o votante clicar no botão Próxima, ele é redirecionado para um tela que mostra as opções selecionadas de cada disputa, como visto na Figura 31. Esta tela é útil pois permite ao votante verificar se ele selecionou corretamente todas as opções antes de enviar seu voto. Estas duas telas encerram o processo de votação, e após o votante clicar no botão Confirmar, ele é redirecionado para a tela principal do sistema, que confirma se o seu voto foi corretamente enviado.

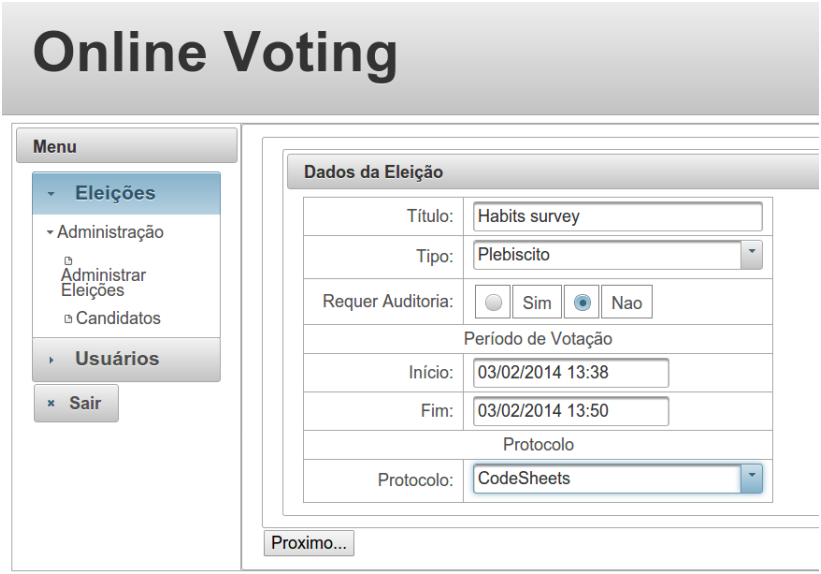

Figura 26: Tela de cadastro da eleição

4.10.6 Verificação do resultado

Após a eleição ter sido encerrada, o administrador pode contabilizar os votos e publicar o resultado. Uma vez que o resultado se encontre publicado, ele é disponibilizado aos votantes, que podem verificar a apuração geral dos resultados, como visto na Figura 32.

4.11 PROTOCOLOS IMPLEMENTADOS PARA AVALIAR O FRAMEWORK

Para verificar se o framework diminui o esforço no desenvolvimento de novos sistemas de votação digital, e se tem a capacidade de suportar diferentes tipos de sistemas, foram implementados três protocolos de votação digital e sistemas para utilizá-los. Por meio deles, foi possível verificar o que precisava ser alterado no framework, o que faltava e o que era necessário acrescentar na sua modelagem por não ter sido anteriormente previsto. Também foi possível verificar o grau de reúso do framework e sua utilidade no desenvolvimento de novos protocolos.

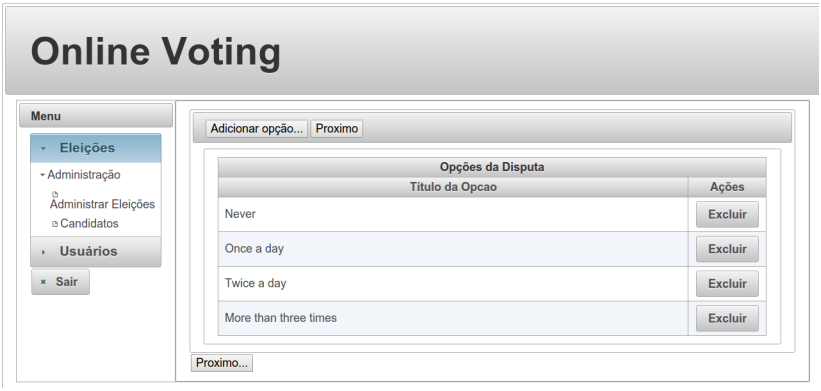

Figura 27: Tela de cadastro de opções de uma determinada disputa

Figura 28: Tela de auditoria da eleição

4.11.1 Protocolo simplista

O protocolo simplista é talvez o mais simples de todos os protocolos (ARAUJO, 2002), não fazendo uso de nenhuma primitiva criptográfica e não tem nenhum mecanismo de segurança específico, sendo necessário a confiança nos votantes, nos demais participantes e também no sistema de votação. Este protocolo funciona da seguinte maneira:

1. O votante se autentica no sistema através de algum mecanismo de autenticação, como login e senha ou através do uso de certificados digitais.

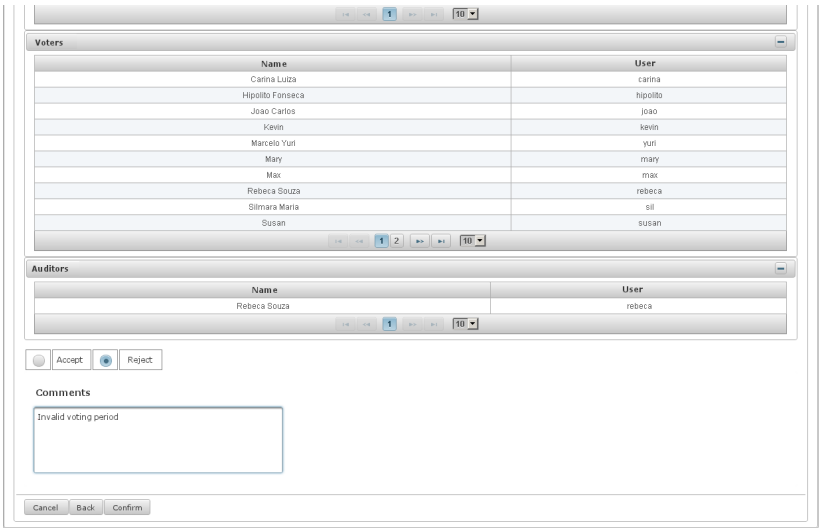

Figura 29: Tela de auditoria da eleição, sendo que o auditor informou que os dados da eleição contêm erros

Figura 30: Tela inicial de votação, aguardando a seleção do votante

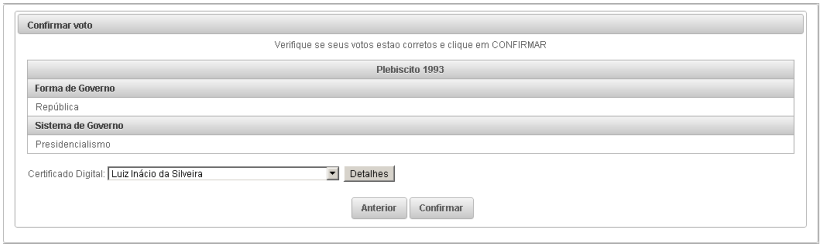

Figura 31: Tela de confirmação das opções selecionadas

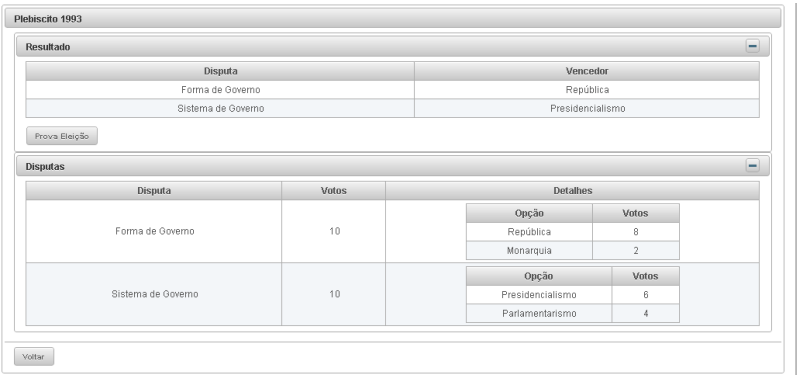

Figura 32: Tela de resultados da eleição

2. O votante envia o seu voto para o sistema de forma livre, sem fazer uso de assinatura ou cifragem.

3. O sistema registra o voto do votante.

4. Quando a votação é encerrada, o sistema faz a contagem dos votos e publica o resultado.

Este protocolo assume que todas as suas entidades são confiáveis, ou seja, considera-se que o votante não é desonesto, que o canal de comunicação entre o votante e o sistema é confiável e que o sistema de votação não é corrupto. Caso contrário nenhum requisito de segurança é satisfeito, pois não existem mecanismos para garantir que estas entidades não tenham como ser corrompidas, como o caso do canal de comunicação, onde as informações trafegam sem nenhum mecanismo de segurança, podendo ser interceptadas. Este protocolo também não é verificável, pois o votante não tem como verificar que seu voto foi apurado corretamente, nem que a somatória dos votos se deu de maneira honesta.

4.11.2 Protocolo de votação com assinatura cega

O protocolo anterior exige confiança no sistema de votação e no ambiente onde o sistema é utilizado, pois caso eles não sejam confiáveis, é possível manipular a eleição de diversas maneiras (ARAUJO, 2002):

- associar o voto ao votante, ou seja, divulgar quem votou em quem.

- adicionar votos inválidos na contagem do resultado.
- alterar o voto dos votantes, alterando com isto, o resultado da eleição.
- remover votos válidos.
- fazer a contagem dos votos de forma errada.
- divulgar um resultado errado da eleição.

É importante garantir que mesmo que as entidades do sistema tenham um comportamento malicioso, não será possível corromper a eleição. Por exemplo, mesmo que um atacante tente manipular o resultado da eleição, seja alterando os votos de outros votantes ou através de outra técnica, isto não será possível ou será consideravelmente mais complicado, devido a restrições impostas pelo protocolo. Uma das possíveis técnicas adotadas é o uso da assinatura cega utilizada para evitar a associação do votante com o seu voto, garantindo assim o anonimato.

Este protocolo funciona da seguinte maneira:

1. A primeira etapa deste protocolo exige que o votante gere 10 conjuntos de cédulas. Cada conjunto contém todas as possibilidades de voto, por exemplo, se existem 3 alternativas, o conjunto conterá 3 cédulas, uma com a primeira opção selecionada, a segunda com a segunda opção, e a terceira com a última opção selecionada. Cada cédula também deverá conter um número serial único e aleatório, grande o suficiente para evitar duplicação.
2. Em seguida o votante blinda cada cédula com um fator de blindagem diferente e assina cada um dos 10 conjuntos, enviando-os então para o sistema.
3. O sistema verifica se o votante já enviou algum outro conjunto de cédulas através da verificação da assinatura. Caso essa seja a primeira vez, ele pede o fator de blindagem de 9 dos 10 conjuntos. Ele abre os 9 conjuntos e verifica se estão todos corretamente formados e assina cegamente o 10^{o} conjunto, aquele que não foi desblindado. O sistema assina cada cédula individualmente, sem desblindá-las, e devolve o conjunto assinado para o votante e registra que ele já solicitou sua assinatura.
4. O votante remove o fator de blindagem do conjunto recebido e escolhe sua opção de voto, ou seja, a cédula que contém a opção que ele deseja. Ele cifra essa cédula com a chave pública do sistema e a envia de forma anônima.

5. O sistema decifra a cédula recebida com a sua chave privada, verifica a assinatura com a sua chave pública para garantir que foi ele quem assinou aquela cédula anteriormente (através da assinatura cega), salva o número serial e registra o voto.

6. Ao final do período de votação o sistema faz a contagem das cédulas recebidas e publica o resultado, assim como o número serial de cada cédula associada a cada voto.

Este protocolo tenta resolver o problema do anonimato, garantindo que mesmo que o sistema queira descobrir em quem cada votante votou, ele não conseguirá. Isto acontece pois os votantes enviam o seu voto de forma anônima, sendo que cada voto foi anteriormente validado pelo sistema, garantindo sua validade. Considera-se que nesse caso o sistema não teria como mapear o endereço de rede dos computadores de onde se originam os votos, pois assim seria possível descobrir de onde os votos se originaram.

Outro problema encontrado nesse protocolo é que nada impede que caso o sistema seja desonesto, ele envie votos em nome dos votantes que se abstiveram de votar, forjando cédulas assinadas por eles. Outro problema é que este protocolo não é livre de coação devido à existência de recibo, permitindo ao votante provar em quem votou.

4.11.3 Protocolo com rede de mistura

Este protocolo tenta garantir o anonimato dos votantes através do uso de uma rede de mistura. Ela é útil em impedir a associação entre as mensagens que chegam e seus remetentes, garantindo que não será possível relacionar os votos que chegam à urna com os votantes que acabaram de votar.

O funcionamento deste protocolo se dá da seguinte forma:

1. O votante se autentica no sistema através de algum mecanismo de autenticação.

2. O votante cifra sua cédula com a opção de voto já selecionada utilizando a chave pública da urna.

3. Ele envia seu voto cifrado para a urna através da rede de mistura.

4. O sistema busca as cédulas na urna, que as decifra, e faz a contagem dos votos publicando em seguida o resultado.

Este protocolo é bastante simplificado e tem por objetivo esconder a ordem de chegada dos votos até a urna, de forma que mesmo que o sistema seja corruptível ou a rede esteja sendo analisada, não seja possível descobrir quem emitiu qual cédula. Este protocolo, como os anteriores, apresenta algumas vulnerabilidades, tais como a falta de verificação individual e universal, não permitindo que os votantes verifiquem se seus votos foram contados corretamente e que o resultado da eleição se deu de maneira honesta.

4.12 ADEQUAÇÃO DO FRAMEWORK

A boa prática no desenvolvimento de frameworks recomenda que sejam implementados ao menos três sistemas utilizando o framework em questão, para verificar sua adequabilidade no desenvolvimento de sistemas em sua área de domínio. A intenção com o desenvolvimento dos três protocolos citados acima foi justamente analisar quão adequado estava a estrutura do framework para a implementação de sistemas e protocolos de votação digital. Com isto, foi possível verificar quais estruturas deveriam ser incluídas ou modificados e como determinadas classes contidas no framework poderiam ser melhor modeladas.

4.13 CONCLUSÃO

O framework foi projetado e desenvolvido de forma a ser flexível o suficiente para implementar uma grande gama de protocolos com características distintas entre si. Ele permite que o desenvolvedor crie sistemas em um período de tempo menor justamente pelas facilidades que o framework oferece, como estrutura de autenticação, controle de acesso, gerenciamento de cédulas, máquina de estado da eleição, dentre outros. Devido a isto, é possível não só implementar estes sistemas de forma mais rápida, como, também, de forma mais confiável, uma vez que o framework oferece mecanismos que já foram testados em diversos projetos, tornando os sistemas que os utilizam menos suscetíveis a erro e mais maduros.

5 USO DO FRAMEWORK PARA A AVALIAÇÃO DE UM PROTOCOLO DE VOTAÇÃO DIGITAL

Os protocolos descritos na seção anterior foram importantes no sentido de mostrar que o framework provê a infraestrutura básica para desenvolver sistemas de votação digital, e portanto reduz o esforço durante o desenvolvimento. Mas estes protocolos são relativamente simples, não sendo possível utilizá-los em eleições reais por não atenderem vários requisitos de segurança. Escolheu-se implementar um protocolo descrito na literatura, mais complexo e que promete garantir vários requisitos de segurança, podendo ser usado em diversos tipos de eleições. O protocolo escolhido foi o Code Sheets (HELBACH; SCHWENK, 2007), baseado em um protocolo de cédulas de papel proposto por David Chaum.

Um dos objetivos com a implementação do Code Sheets é mostrar que o framework também pode ser utilizado na avaliação de propostas de protocolos, assim como mostrar sua viabilidade em implementar sistemas que podem ser utilizados em situações reais. Como o framework reduz o esforço necessário para a implementação destes protocolos, se torna mais fácil testar sistemas de votação que os utilizem, permitindo desenvolver protocolos que foram apenas idealizados e tirá-los do papel, o que permite comprovar sua viabilidade de forma prática. Com isto, podemos testar inúmeros cenários de utilização, simular ataques e realizar uma busca por vulnerabilidades, o que em alguns casos não seria possível sem um sistema implementado. Assim podemos ver que o framework não apenas fornece a estrutura para o desenvolvimento de sistemas de votação, como também permite testar e verificar a viabilidade e a resistência a ataques destes mesmos sistemas.

5.1 DESCRIÇÃO DO PROTOCOLO CODESHEETS

Este protocolo possui um comportamento bastante diferenciado em relação aos outros protocolos implementados. Uma de suas particularidades é que ele permite a atualização do voto e permite ao usuário votar sem a necessidade de autenticação. Outra característica importante é que ele não faz uso de nenhuma primitiva criptográfica, contando com outros tipos de mecanismo para garantir a segurança do protocolo.

A cédula utilizada por este protocolo também difere das demais, sendo utilizada uma estrutura chamada de cartelas code sheet. Estas cartelas contém

CANDIDATE	VOTING TAN	CONFIRMATION TAN
David Scott	1287457877	2376554221
Jason Harris	1238900012	6364100273
Kevin Moore	8985200233	7283666319
Ronald Taylor	2348762349	2389472993

CodeSheet Identification: 9762209221

Figura 33: Cartela típica do protocolo CodeSheets

um identificador único, responsável por identificá-las durante o processo de votação. Elas contém também todas as disputas e suas respectivas opções de voto, cada uma delas associada a dois números, como pode ser visto através da Figura 33. O primeiro número é conhecido como TAN de votação, utilizado para identificar a opção do votante, e o segundo é TAN de confirmação, utilizado para conferir a opção selecionada. A proposta original deste protocolo prevê o envio das cartelas através dos correios. Para a implementação utilizando o framework, decidiu-se enviar as cartelas via e-mail de forma a facilitar as simulações.

Este protocolo contém apenas uma autoridade, que é responsável por gerenciar o processo de votação. Na primeira etapa ela é responsável por gerar cartelas únicas em número suficiente de forma que cada eleitor receba uma delas. Elas são então colocadas aleatoriamente dentro de envelopes com o endereço dos votantes, de forma que não seja possível conhecer de antemão o destino de cada cartela e, em seguida, elas são enviadas pelos correios.

Após esta etapa de inicialização, vem a parte da eleição propriamente dita. Cada votante entra no sistema sem se identificar, informa o número de sua cartela para identificá-la no sistema e em seguida vota. O processo de votação se dá da seguinte maneira: para cada disputa o votante informa o TAN de votação do candidato escolhido. Este TAN é um número aleatório grande o suficiente para não haver repetição e para não ser descoberto através de força bruta. O sistema então retorna o TAN de confirmação para cada TAN de votação, permitindo ao votante ter certeza que sua opção foi reconhecida pelo sistema, uma vez que cada TAN de confirmação está relacionado a apenas um TAN de votação.

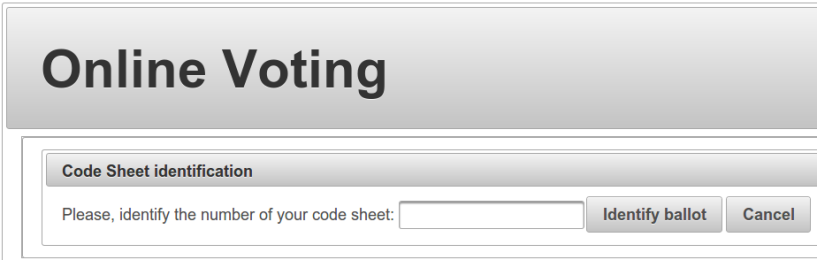

Figura 34: Tela onde o usuário entra com o identificador da cartela

5.2 UTILIZAÇÃO DO PROTOCOLO IMPLEMENTADO

Foi necessário criar um conjunto de telas específicas para o Protocolo CodeSheet, uma vez que as telas existentes funcionam de maneira diferenciada. Nos outros protocolos é necessário que a primeira coisa que o votante faça seja se autenticar no sistema, para só então ter acesso às eleições onde está cadastrado, e votar naquelas que estão publicadas e dentro do período de votação. Eleições que utilizam o protocolo Code Sheets funcionam de maneira levemente diferente, exigindo que o votante não esteja autenticado durante a votação. Para ter acesso a página inicial da votação, ele deve entrar com a URL específica da eleição, que também foi enviada por e-mail juntamente com a sua cartela. Tendo acesso a essa página, a primeira coisa que o votante deve informar é o identificador da eleição.

Com o código informado o sistema sabe com qual cartela está lidando e conhece quais códigos de votação pode esperar e quais código de confirmação deve retornar. Esta é a primeira tela acessada pelo votante podendo ser vista pela Figura 34. Após informar o código e o sistema conseguir localizar esta cartela no banco de dados, o usuário é redirecionado para a tela com as opções de voto. Diferente dos outros protocolos, aqui não são mostradas as opções de voto de forma selecionável, como dentro de um radio group, mas apenas o nome de cada opção e um campo para informar o identificador da opção desejada. Figura 35.

O votante deve localizar na cartela recebida por e-mail qual o código TAN da opção desejada e informá-la na caixa correspondente para cada uma das disputas. Feito isto, o sistema busca qual o TAN de confirmação para cada TAN de votação informado pelo usuário, como pode ser visto através da Figura 36. Desta maneira é possível ao votante saber que o código entrado foi realmente reconhecido pelo sistema e que a sua opção foi corretamente seleci-

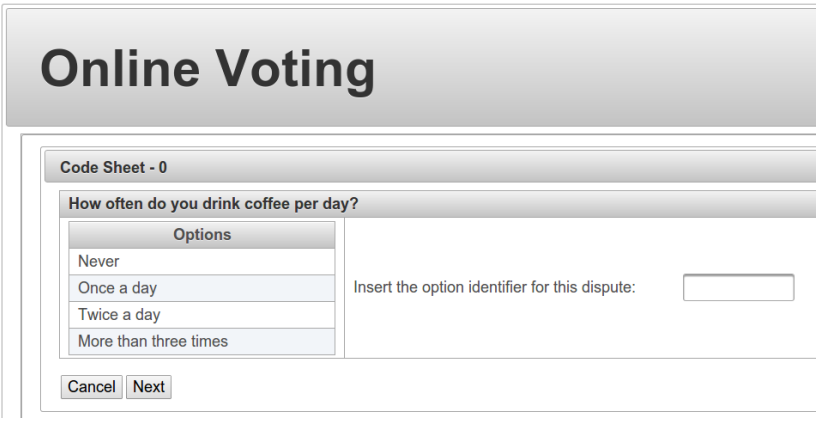

Figura 35: Tela onde o usuário deve informar o TAN de votação da opção desejada

onada. Após verificar todos os códigos de confirmação retornados, o votante pode submeter suas opções ao sistema e caso tudo ocorra corretamente, ele será redirecionado para a tela avisando se seu voto foi corretamente enviado ou não.

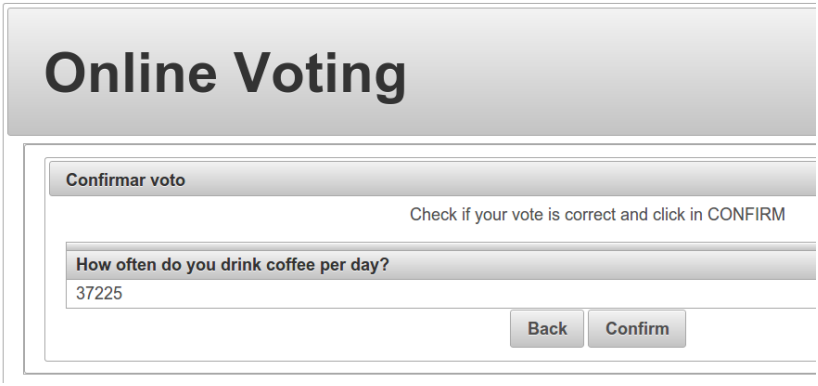

Figura 36: Tela mostrando o TAN de confirmação da opção selecionada

Tendo conhecimento de como este protocolo funciona e como os anteriormente explicados também funcionam, é possível ver que o comportamento exigido do votante é diferente, exigindo que as telas fossem todas

desenvolvidas novamente. A grande vantagem é que por utilizarmos o framework para este trabalho, apenas as telas foram reescritas, sendo que as classes que fazem a comunicação entre o core do framework e a parte gráfica precisaram apenas de alguns novos métodos implementados, e o core praticamente se manteve intacto. Mas isso será explicado em mais detalhes na próxima seção.

5.3 IMPLEMENTAÇÃO

A proposta original do protocolo prevê o envio das cartelas através dos correios, o que dificultaria a simulação de testes sem agregar vantagens. Isto foi alterado para que as cartelas fossem enviadas via e-mail, ainda de forma anônima, mantendo ainda as características almejadas pelo protocolo.
Devido ao desenvolvimento com o framework, foi necessário implementar apenas partes dos sistema que são específicas deste protocolo. Como exemplo podemos citar o uso das cartelas, cuja lógica e utilização difere dos outros protocolos.

5.3.1 Classes e métodos implementados

Para o desenvolvimento do protocolo CodeSheet foi necessário implementar algumas novas classes, sendo as outras reutilizadas do framework. A principal classe desenvolvida foi a relativa ao protocolo, chamada de ProtocoloCodeSheet. Esta classe herda a classe Protocol, e é responsável por toda a lógica de funcionamento do protocolo Code Sheets. Dentre seus métodos, três deles foram sobrescritos:

- Initialize: esta etapa é executada quando a eleição é publicada, ou seja, todas as informações sobre a eleição já estão definidas e não podem mais ser alteradas. Portanto, quando esta etapa é executada, já é conhecido o número de eleitores. Neste protocolo esta fase é responsável por gerar todas as cartelas para cada um dos votantes, e enviá-las por e-mail.

- GetBallot: o procedimento para obter a cédula é diferente da dos outros protocolos. Enquanto nos protocolos anteriores obtinha-se uma cópia da cédula e a configurava na hora de enviá-la ao usuário, neste protocolo a cartela já está previamente formada, e deve ser recuperada para cada votante. Como o usuário já recebeu a cartela via e-mail, na hora de votar ele deve informar o identificador da cartela, e esta é recuperada

do banco de dados.

- SendBallot: O envio da cédula se dá de maneira semelhante aos outros protocolos, a cartela é transformada novamente em uma cédula e tem as opções escolhidas pelos votantes marcadas na cédula, e então ela é enviada à urna.

Todos os outros métodos desta classe se mantiveram intactos, inclusive o relativo à contagem de votos que continuou utilizando a abordagem *default*, onde são coletadas todas as cédulas da urna e cada uma delas é somada para a contagem final. O mesmo acontece em relação à publicação e verificação do resultado, ambas mantendo-se da forma originalmente prevista no framework.

Outra alteração exigida foi a criação da classe relativa às cartelas, chamada de CodeSheet, como pode ser visto na Figura 37. Esta classe herda a classe Ballot, mantendo boa parte de sua estrutura e métodos. Além dos métodos já implementados na classe pai, foi preciso implementar aquele responsável por gerar todos os TAN para cada opção de voto e métodos responsáveis por fazer downcasting e upcasting entre as classes CodeSheet e Ballot. Também foi necessário implementar um método para selecionar as opções de voto, uma vez que agora a opção é selecionada buscando o código TAN dela, ao invés de se passar o id da opção, como anteriormente. Devido a isto, foi também necessário implementar mais duas classes relativas à cédula: CodeSheetTan e ListCodeSheetTan. A primeira diz respeito ao código TAN propriamente dito, reunindo na mesma classe o código de votação, código de confirmação e o identificador da opção. A segunda classe é uma lista de CodeSheetTan, relativa a cada disputa da cartela.

Para o processo de votação, é necessário que o votante acesse o sistema de forma anônima. Para isto, é criada uma URL diferente para cada votação que utilize este protocolo, de forma que ela seja disponibilizada aos votantes. Após o acesso a esta página, o votante deve informar o identificador da cartela que ele possui, e para cada disputa, deve informar o TAN de votação da opção por ele selecionada. O sistema retorna para cada opção o TAN de confirmação, sem identificar qual foi a opção escolhida.

Além das classes descritas, foi necessário estender outras classes. Foi preciso implementar um método para salvar e recuperar as cartelas, uma vez que é necessário saber os números de votação e de confirmação de cada cartela na hora que o votante a identifica e envia sua opção. Isto foi necessário pois as cartelas contém atributos diferentes da cédula, sendo necessário per-

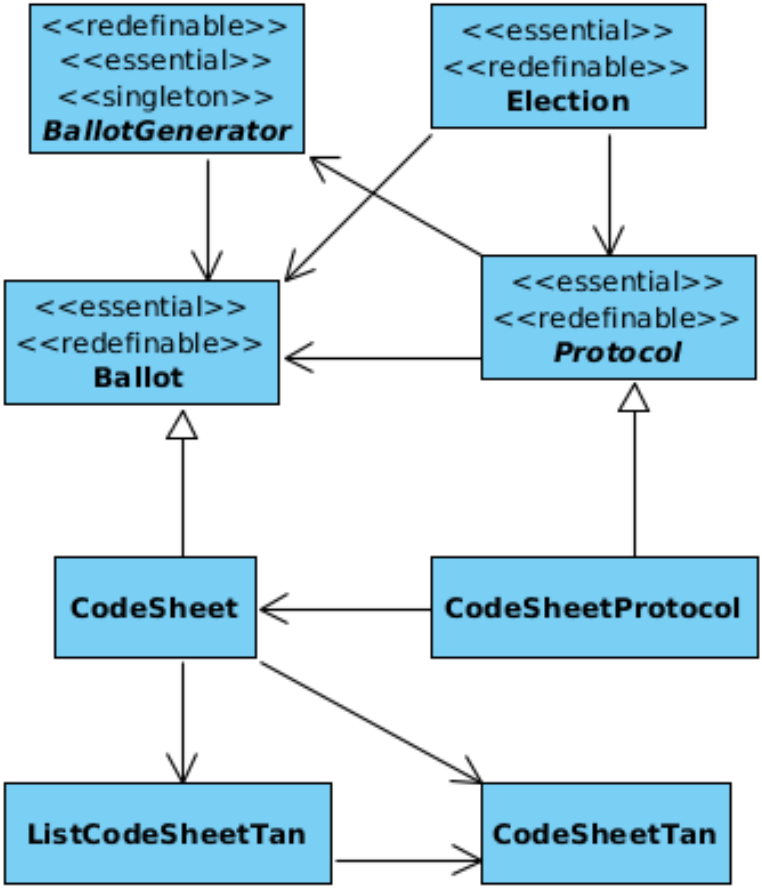

Figura 37: Classes implementadas

sistir estes atributos. Com isto, foi necessário sobrescrever o gerenciador de Eleição, pois na hora da publicação da eleição ele passou também a salvar as cartelas recém criadas. Também foi necessário estender as classes que fazem a comunicação com o banco de dados, para salvar as cartelas e suas classes dependentes. Assim como foi preciso alterar as classes que fazem o relacionamento com as telas, como será explicado na sessão seguinte.

Além das subclasses criadas, diversas outras foram reutilizadas do framework, sem qualquer alteração. A classe relativa a eleição foi reutilizada exatamente como encontrada no framework, não sofrendo qualquer modificação em sua estrutura, assim como seu gerenciador. Também foram reutilizadas as classes que gerenciam os perfis de usuário e sua autenticação, uma vez que os administradores e demais perfis de controle ainda se autenticam para gerenciar a eleição. A autenticação de votantes não é necessária, uma vez que eles provam sua legitimidade através do identificador da cartela que recebem. Como cada cartela é única e confidencial, toda pessoa que possuir um identificador válido para aquela eleição específica deve ser um votante legítimo, a menos que tenha obtido a cartela por meios desonestos. Mesmo que o usuário obtenha um identificar válido por força bruta, ele não terá como votar em determinada opção, pois teria também que adivinhar por tentativa e erro um TAN de votação válido, e assim que este fosse identificado, não seria possível saber a qual opção ele diz respeito, pois tudo que o sistema retorna é um TAN de confirmação, que não revela nada sobre a opção em si.

O sistema de gerenciamento de cédulas também foi reutilizado. Ao invés de gerar cédulas do tipo Ballot ele passou a gerar cartelas do tipo CodeSheet, mantendo ainda a estrutura de disputas e opções de voto. Em algumas classes foi necessário fazer pequenas alterações para lidar com as classes recém criadas. Foi preciso implementar um método para salvar e recuperar as cartelas, uma vez que é necessário saber os números de votação e de confirmação de cada cartela na hora que o votante a identifica e envia sua opção. Isto foi necessário, pois as cartelas contêm atributos diferentes da cédula, sendo necessário persistir estes atributos. Também foi necessário alterar as classes que fazem o relacionamento com as telas, como descrito na sessão seguinte.

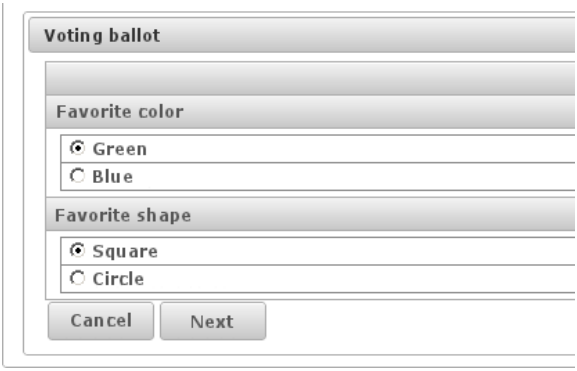

Figura 38: Tela de votação dos protocolos anteriores

5.3.2 Telas

Como explicado anteriormente, foi necessário desenvolver um novo conjunto de telas para este protocolo, e consequentemente alterar a classe que faz o relacionamento entre a tela e o core do framework. A responsável por isto foi a classe VotingMB, que lida com o controle das telas relacionadas com o processo de votação. Foram implementados dois métodos, um para recuperar a cartela e outro para recuperar o TAN de confirmação relativo ao TAN de votação recebido. O método para o envio da cédula se manteve igual, posteriormente sendo alterado para suportar um mecanismo de segurança, explicado adiante.

Em relação ao método de obtenção da cartela foi necessário criar um método para obter a cartela do banco de dados para retorná-la a classe VotingMB, que tem o conhecimento necessário para manipular a cartela.

5.3.3 Reúso

Para a implementação deste protocolo através do framework foi possível reusar aproximadamente 85% do código existente. O critério utilizado levou em conta a contagem de classes. O framework contém mais de 250 classes, sendo apenas sete classes novas implementadas (quatro classes e três telas) e mais de 50 reutilizadas, o que permitiu que a maior parte do sistema fosse implementado reusando a estrutura do framework. Como toda a lógica do protocolo fica contida na classe relativa ao protocolo, não é necessário a

criação de muitas classes novas. Além disso, o framework prevê que novas classes implementadas possam ser facilmente utilizadas através de herança, sem a necessidade de reconstruir partes do sistema já previstas no framework.

5.4 ANÁLISE DO PROTOCOLO CODE SHEETS

Um dos objetivos da implementação do protocolo Code Sheets é mostrar que o framework é capaz de implementar sistemas de votação digital para serem usados em situações reais. Mas além disto, mostrar que o framework, justamente por reduzir o esforço necessário no desenvolvimento, permite avaliar se estes sistemas são realmente confiáveis através da realização de testes e pela busca de vulnerabilidades. A implementação dos protocolos permite que seja feita uma análise prática, pois é possível testar diferentes cenários e ataques, como será mostrado mais adiante.

5.4.1 Interceptação do canal de comunicação

Na implementação feita não foi utilizado nenhum mecanismo de cifragem para os dados trafegados na rede, da mesma forma que descrito pelos autores. Como as únicas informações que trafegam pela rede são os TAN de votação e o identificador da cédula por parte do votante, e o TAN de confirmação por parte do sistema, não é possível identificar qual a opção de voto que está sendo selecionada, uma vez que estes números não dizem nada a respeito disso. Mesmo que se identifique através do endereço IP qual máquina está sendo utilizada, e possivelmente qual o usuário que a está utilizando, mesmo assim não será possível identificar seu voto.

5.4.2 Negação de serviço

O ataque de negação de serviço é uma forma de tornar um sistema indisponível para seus utilizadores. Existem diversas formas de praticar este ataque, sendo as principais através da sobrecarga do sistema (KIM et al., 2004). Isto pode ser feito eliminando todos os recursos disponíveis, como memória ou processamento, de forma que ele não possa mais oferecer seus serviços. Também é comum o bloqueio do canal de comunicação entre o sistema e seus utilizadores.

Considerando o protocolo, seria possível causar uma sobrecarga no

sistema com diversas requisições ao mesmo instante, provavelmente utilizando a mesma cartela para isto, uma vez que é difícil obter várias delas. Uma maneira possível de se obter uma cartela, caso o atacante não fosse um votante legítimo e não tenha a opção de usar a sua própria para o ataque, seria através da interceptação e roubo da cartela de outro usuário. Este é um cenário possível quando se utiliza o envio via e-mail, como também quando se utiliza o envio via correio convencional. A intenção do atacante por trás desta prática não seria simplesmente votar inúmeras vezes, uma vez que é possível atualizar seu voto, mas sobrecarregar o servidor com uma quantidade abusiva de requisições, inviabilizando o funcionamento do sistema.

Uma solução possível seria limitar o número de requisições que uma mesma cartela pode enviar em determinado período de tempo. Por exemplo, uma vez que o votante terminou de selecionar todas as opções, caso ele deseje atualizar novamente seu voto, deve esperar certo período de tempo de forma que não seja possível enviar requisições intermitentemente. Apesar dessa solução, ainda seria possível ao atacante utilizar IPs diferentes na hora de enviar o voto, podendo fazer uso, por exemplo, de proxies para utilizar o sistema. Sendo assim, seria necessário bloquear o acesso não apenas pelo número da cartela, mas também pelo IP. Um terceiro cenário seria aquele onde o usuário poderia enviar números TAN inválidos, fazendo uso ou não de proxies. Para evitar que isto sobrecarregue o sistema, o número TAN pode conter números de validação, de forma que seja possível validá-lo antes de buscá-lo no sistema ou de verificar se ele não foi enviado recentemente. Estas formas de ataque, como também as soluções propostas, não são previstas pelos autores do protocolo.

5.4.3 Coerção dos votantes

Uma das principais ameaças em sistemas de votação digital é justamente a coerção dos votantes, onde um ataque obriga ou influencia o usuário a votar em determinada opção. Este é um problema presente em diversos protocolos online devido a dificuldade em garantir que o votante estará sozinho na hora de enviar seu voto, sem nenhum atacante o ameaçando a votar em determinado candidato ou mesmo emitindo seu voto em seu lugar. Os autores propõem como alternativa a atualização dos votos. Isto permite que o votante altere a opção escolhida futuramente, sendo apenas a última atualização contabilizada no sistema. Isto dificulta a compra de voto e a coerção dos votantes, uma vez que o comprador ou o atacante não sabem se o votante atualizou mais

tarde seu voto, sobrescrevendo a opção anteriormente selecionada.

5.5 SIMULAÇÃO

Com a implementação de um sistema de votação que utilize o protocolo desenvolvido, foi possível realizar um conjunto de testes para verificar sua eficácia contra possíveis ataques, como, por exemplo, a interceptação e roubo de cartelas. O sistema implementado envia as cartelas via e-mail, diferente da proposta original onde elas eram enviadas pelos correios. O roubo de cartelas daquela forma é uma ameaça possível, pois alguns servidores de e-mail não criptografam o tráfego de mensagens, sendo possível interceptá-las. Também é possível a invasão de servidores de e-mail e o roubo de seu conteúdo. Uma solução possível para isso envolveria o uso de algumas primitivas criptográficas, como a cifragem do conteúdo da mensagem, impedindo que mesmo que fossem interceptadas, suas informações não estariam disponíveis ao atacante. Já no caso das cartelas serem enviadas por correio, é mais difícil roubá-las de forma escalável.

Outra simulação é o ataque de negação de serviço, através do envio de requisições de forma abusiva ao servidor de aplicação. A solução proposta por nós é limitar o número de requisições dentro de um determinado intervalo de tempo, como por exemplo, restringindo o votante a enviar seu voto apenas uma vez a cada minuto, sendo obrigado a aguardar este tempo caso queira sobrescrever sua opção anterior. Isto dificultaria situações onde o atacante envie inúmeras requisições de forma ininterrupta para tentar sobrecarregar o servidor.

5.5.1 Temporizador de votos

Para assegurar que uma cartela específica seja utilizada apenas uma vez dentro de um determinado intervalo de tempo, foi desenvolvido um mecanismo chamado de Temporizador de votos. Para garantir que o votante não envie uma grande quantidade de votos de uma única vez, é feita uma checagem antes da obtenção da cédula. Quando o usuário informa o identificador da cartela, a primeira coisa que o sistema faz é analisar se aquela cartela foi enviada há menos de um minuto. Caso ela tenha sido enviada, uma flag é setada avisando que os próximos passos devem também ser bloqueados ao usuário, e o sistema não chega nem a buscar a cartela no banco de dados. Com isto, foi possível reduzir de três acessos ao banco de dados para

zero, pois anteriormente seria necessário buscar a cartela do usuário, buscar os TANs de confirmação relativos aos TANs de votação recebidos e enviar a cartela. Ao invés disso, é armazenada em memória a informação do último acesso de cada cartela, otimizando o processo.

Da mesma forma que é realizada a verificação por identificador da cartela, também é efetuada a validação por IP do usuário, de forma que mesmo que ele envie identificadores diferentes, é possível bloquear requisições de um IP específico durante determinado período de tempo. Nesse cenário o atacante utilizaria possivelmente identificadores inválidos de forma a sobrecarregar o sistema. Um solução para isto seria validar o identificador da cartela, como explicado em seguida.

5.5.2 Validação do identificador da cartela

Um cenário de ataque possível seria aquele onde o usuário utiliza conexões diferentes, fazendo uso, por exemplo, de *proxies* e envia vários identificadores diferentes, utilizando IPs diferentes, de forma a sobrecarregar o sistema. Possivelmente os identificadores enviados pelo atacante seriam inválidos, uma vez que é difícil conseguir identificadores válidos de outros usuários, exigindo ao atacante gerar estes números de forma independente. Neste caso, bloquear o acesso ao sistema via IP ou via identificador da cartela não seria suficiente para bloquear o atacante, e o sistema verificaria para cada requisição se aquela cartela existe no banco de dados. Uma solução para isto é gerar identificadores de cartela com dígitos de validação, de forma que caso o atacante envie identificadores inválidos, o sistema não precisa efetuar nenhuma consulta ao banco para saber que aquele número não corresponde a nenhuma cartela. Para isto foi implementado um mecanismo para gerar e validar os identificadores. Essa validação é realizada antes da verificação do temporizador de votos e permite que aquela requisição seja descartada como inválida antes de qualquer outra verificação do sistema. A geração dos identificadores ocorre de forma semelhante à geração de CPFs. No caso, são gerados 5 números de forma aleatória e cada um dos dígitos é multiplicado por uma constante específica. O primeiro dígito é multiplicado por 1, o segundo por 2, e assim sequencialmente, até o último dígito. O resultado de cada uma das multiplicações é somado, e o total é dividido por 11. Caso o resto da divisão seja menor que 2, o primeiro dígito de validação se torna 0, caso contrário, subtrai-se o valor encontrado de 11, sendo o resultado o dígito de validação. Este primeiro número encontrado é anexado ao final do número gerado aleatoriamente. O mesmo processo se repete dessa vez considerando também o primeiro número de validação encontrado, para dessa forma ob-

ter o segundo dígito. Assim, quando o usuário entrar com o identificador da cartela, é possível verificar se os dois últimos dígitos estão corretos, para só então continuar o processo de votação.

5.5.3 Ataque de negação de serviço

Para avaliar como determinado sistema se comporta em termos de capacidade de resposta e estabilidade sob uma determinada carga de trabalho, podem ser utilizados testes de performance. Nossa intenção não é simplesmente medir o desempenho do sistema e comprovar uma possível melhora em sua performance, mas verificar que através da solução implementada conseguimos dificultar tentativas de negação de serviço, impedindo a sobrecarga do sistema por apenas um usuário.

Para a realização dos testes foi utilizada a ferramenta Apache Jmeter (JMETER, 2014) que permite a execução de testes de estresse, possibilitando a simulação de um grande número de usuários no sistema realizando determinada ação. Utilizamos um servidor com 512MB de memória, 8GB de HD e 1 processador, utilizando o sistema operacional Ubuntu. Foram simuladas tentativas de envio da mesma cartela para simular o caso onde um atacante tenha tido acesso a cartela de algum usuário, ou ele próprio seja um votante legítimo, e tente sobrecarregar o sistema votando inúmeras vezes. Este cenário é possível uma vez que o protocolo permite ao mesmo votante sobrescrever seu voto, sendo apenas o último contabilizado. Apesar disto, esta situação não foi prevista pelos autores do protocolo, não sendo considerada uma solução possível para este problema.

Considerado este cenário, e sem a implementação de nenhum mecanismo adicional que impeça a sobrecarga do sistema, foi simulado que um atacante enviou 300 requisições simultâneas em 1 minuto, o que foi o suficiente para sobrecarregar o banco de dados e impedi-lo de aceitar o recebimento de novos votos, apesar do sistema aparentemente funcionar, e ser possível navegar entre as páginas, na hora de enviar o voto, o banco simplesmente o recusava.

Feito este teste, foi aplicado o mecanismo temporizador, que impede que a mesma cartela seja utilizada mais de uma vez em menos de um minuto. Isto foi feito salvando a data do último acesso relativo a determinada cartela, e verificando esta informação em toda tentativa de envio da cartela, não sendo necessário, para isto, acessar o banco de dados. Foi possível redu-

Without the mechanism

Error %	Throughput	KB/sec	Avg. Bytes
11,68%	23,3/min	4,85	12804,9
11,68%	1,6/sec	9,11	6006,9
11,75%	1,6/sec	4,48	2954,9
11,72%	1,2/sec	7,62	6698,2
11,81%	46,6/min	2,58	3404,4
11,81%	23,3/min	0,85	2235,5
11,73%	5,8/sec	29,48	5186,1

With the mechanism

Error %	Throughput	KB/sec	Avg. Bytes
0,00%	16,6/sec	2442,28	150528,6
0,00%	16,6/sec	49,97	3077,0
0,00%	12,5/sec	74,99	6150,0
0,00%	8,3/sec	35,21	4321,3
0,00%	4,2/sec	9,22	2262,0
0,00%	4,2/sec	19,50	4780,5
0,00%	74,7/sec	4964,34	68055,4

Figura 39: Trecho das tabelas geradas pelo Jmeter

zir o número de transações de 3 para 0 durante a tentativa de envio de voto ainda dentro do intervalo de um minuto. Foi então realizado outro teste considerando o mesmo cenário, enviando 300 requisições por segundo, mas com o mecanismo habilitado. Desta vez o banco não foi sobrecarregando, permitindo ao sistema continuar funcionando normalmente após o ataque.

5.5.4 Análise do resultado da simulação

Para uma análise um pouco mais detalhada foi realizado um segundo cenário de testes simulando o envio de 500 requisições em um intervalo de 2 minutos, para poder analisar a resposta do servidor dentro de um período de tempo mais extenso. É possível verificar a diferença na resposta do servidor através da Figura 39. A última linha diz respeito ao total de cada co-

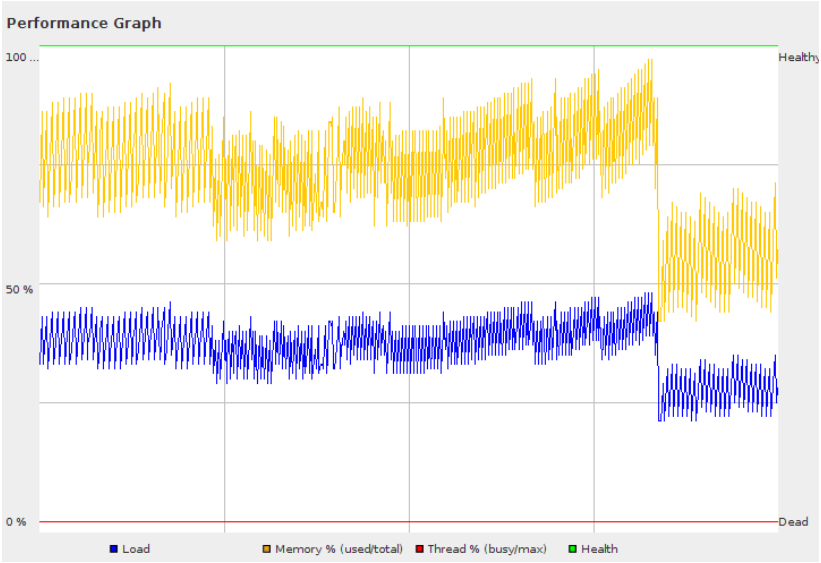

Figura 40: Situação do servidor sem a utilização do mecanismo

luna, sendo a média ou a somatória dependendo do item. Sem o mecanismo houve uma porcentagem de erro de aproximadamente 11% em cada uma das páginas, o que não aconteceu com a sua utilização, não sendo lançada nenhuma exceção durante a execução dos testes. Também podemos verificar que o throughput do sistema foi melhor com o mecanismo do que sem ele, o que significa que o servidor passou a atender mais requisições por segundo.

Podemos visualizar a sobrecarga do servidor pelas figuras 40 e 41. A linha amarela (superior) diz respeito a memória utilizada pelo servidor, quanto mais perto dos 100%, menor a quantidade de memória disponível. A linha azul (inferior) indica a carga no servidor, quanto mais perto de 100% mais carregado o servidor se encontra. A primeira mostra o servidor sem o mecanismo implementado, e é possível ver que depois de determinado tempo o servidor para de atender as requisições recebidas tendo uma queda de perfomance. O servidor não sai do ar, mas o banco de dados não consegue mais suportar tanta carga.

Com o mecanismo, o servidor se mantém constante, como pode ser visto na Figura 41, não sofrendo alterações significativas com o passar do tempo. Podemos perceber que a memória utilizada pelo servidor utilizando o mecanismo cresce em uma velocidade bastante inferior de quando não utilizado. O mesmo podemos dizer sobre a sua sobrecarga.

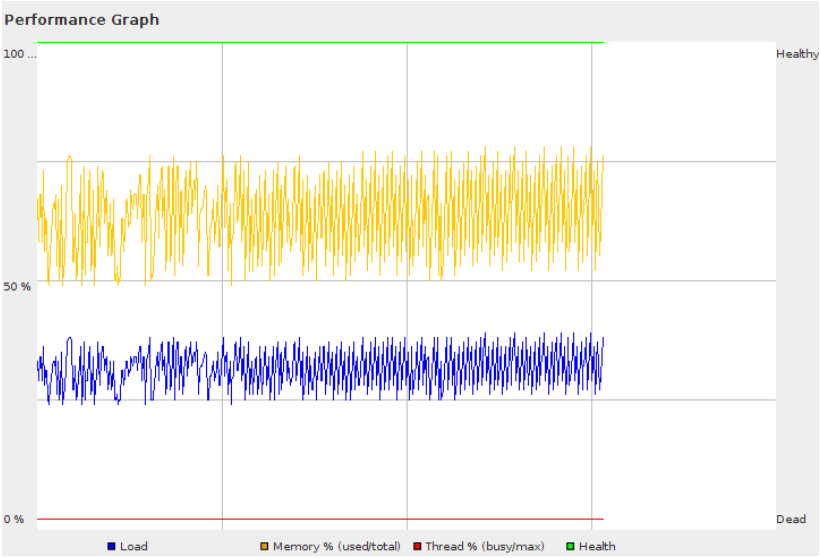

Figura 41: Situação do servidor com a utilização do mecanismo

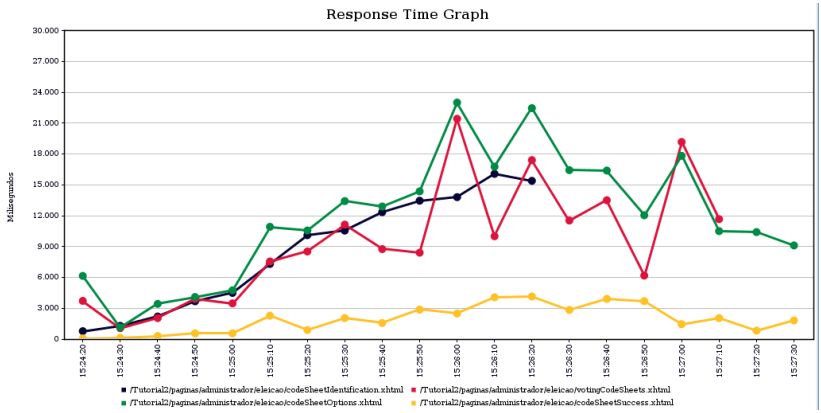

Figura 42: Tempo de resposta do servidor sem a utilização do mecanismo

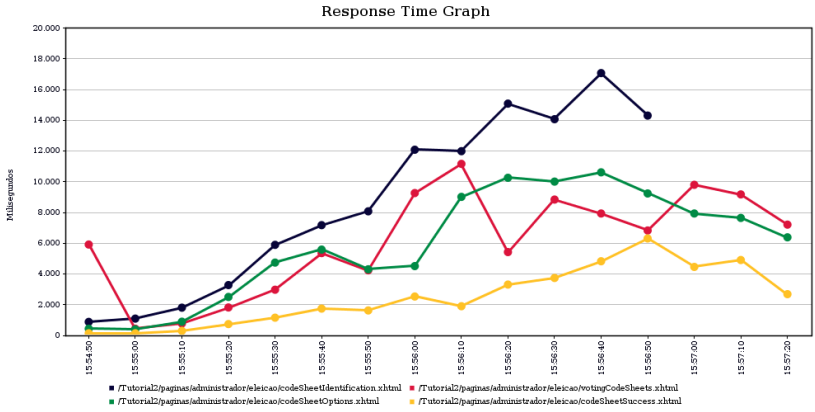

Figura 43: Tempo de resposta do servidor com a utilização do mecanismo

Portanto foi possível verificar que o novo mecanismo utilizado dificulta a queda do sistema por diminuir a sobrecarga do banco de dados e permitir que este continue funcionando corretamente mesmo após o ataque. Outra métrica obtida foi o tempo de resposta do sistema. Como o servidor resolve de maneira mais simplificada as requisições dos votantes, uma vez que não precisa mais fazer três acessos no banco de dado caso aquela cédula já tenha sido utilizada, a resposta se dá em um intervalo de tempo menor.

A Figura 42 mostra o tempo de resposta do servidor sem a utilização do mecanismo durante 2 minutos. O eixo y do gráfico mostra o tempo de resposta, e neste caso varia de 0 à 30,000 milissegundos.

A figura 43 mostra o mesmo experimento com o mecanismo habilitado, sendo que o eixo y do gráfico varia entre 0 a 20,000 milissegundos. O tempo de resposta de algumas páginas caiu pela metade, e a oscilação também diminuiu, se mantendo relativamente mais constante.

5.5.5 Testes de performance

Em ”The Art of Application Performance Testing”(MOLYNEAUX, 2009), Ian Molyneaux cita seis tipos de testes de performance, que podem ser utilizados dependendo da natureza da aplicação e do tempo disponível:

- Baseline testing: geralmente utilizado para medir o tempo de resposta de uma transação. Este teste é normalmente realizado para uma única transação, que é executada por um determinado período de tempo ou

por um determinado número de ciclos. Idealmente deve ser conduzido sem nenhuma outra atividade do sistema.

- Load testing: o modelo mais próximo do uso real do servidor, simulando a interação do usuário com o sistema, considerando inclusive os atrasos e pausas na troca de informação entre o sistema e o usuário. Considera-se uma quantidade de carga específica durante um determinado tempo, sendo um dos testes mais utilizados.

- Stress testing: o objetivo deste teste é determinar o limite de carga da aplicação, tendo por objetivo verificar quando o servidor ou parte da aplicação falha. Com isto, o teste é conduzido até que a aplicação ou parte de sua estrutura falhe: se torne impossível logar no sistema, a aplicação se torna indisponível ou o tempo de resposta se torna inaceitável.

- Soak testing: a intenção com este teste é identificar problemas que só apareceriam depois de um longo período de tempo, como, por exemplo, uma limitação não percebida no número de vezes que uma transação pode ser executada.

- Smoke testing: este teste tem por objetivo avaliar apenas as partes do sistema que sofreram alterações em sua estrutura.

- Isolation testing: utilizado em um aspecto específico da aplicação que se mostrou problemático.

Alguns dos testes citados acima são dependentes da aplicação, não sendo necessário utilizar todos eles quando se deseja avaliar um sistema. A intenção em utilizar testes de performance não era simplesmente medir o desempenho do sistema e comprovar uma possível melhora em sua performance, mas verificar que através da solução implementada foi possível dificultar tentativas de negação de serviço, impedindo a sobrecarga do sistema por apenas um usuário. Com isto, foram realizados testes com a intenção de alcançar o limite de carga da aplicação, fazendo ela entrar em colapso. Foi possível subir este limite após a implementação do mecanismo que impede que um usuário emita o voto inúmeras vezes dentro de um curto intervalo de tempo, o que dificultou a sobrecarga no sistema. Os testes descritos acima não foram utilizados para avaliar o protocolo implementado no presente trabalho. Eles são importantes para verificar aspectos do sistema que não foram testados, como, por exemplo, eleições conduzidas com um grande número de usuários, pois nesse caso, se torna necessário verificar o desempenho e a capacidade do sistema com uma grande carga de requisições.

5.6 REQUISITOS ALCANÇADOS

Realizada a implementação do protocolo Code Sheets, foi possível verificar de forma prática quais requisitos de segurança são atendidos como mostra a Tabela 3.

Além destes requisitos, foi possível verificar com a implementação do protocolo Code Sheets algumas possibilidades de ataque do tipo negação de serviço, permitindo que o sistema fique indisponível para os votantes.

5.7 CONCLUSÃO

Com a implementação do protocolo Code Sheets através do framework, foi possível verificar a viabilidade do protocolo, uma vez que não foi encontrado na literatura qualquer referência a sua implementação e utilização em votações reais. Isto foi facilitado porque o framework reduz o esforço necessário para o desenvolvimento de novos sistemas e protocolos de votação digital através de um alto grau de reúso.

Com o desenvolvimento de um sistema utilizando o protocolo Code Sheets, foi possível simular determinados cenários e ataques e propor soluções para ameaças e verificar sua eficácia. Apesar de não ter sido realizado testes neste sentido, o framework também é útil na hora de verificar o desempenho de protocolos através da simulação de votações reais, verificando se há um tempo de latência muito alto na comunicação com o servidor, o que pode acontecer com protocolos que utilizam muitas primitivas criptográficas. O framework, pelo reúso proporcionado e pela possibilidade de produzir sistemas de votação operacionais, mostrou-se uma ferramenta adequada de apoio à avaliação de protocolos de votação, comprovando a hipótese de pesquisa estabelecida.

Requisito	Conclusão
Exatidão	**Atendido:** apenas cédulas válidas são contadas ao final da apuração. É necessário ter acesso a uma cartela válida para ser possível emitir o voto, sendo que existe o mesmo número de cartelas e de votantes.
Unicidade	**Parcialmente atendido:** o sistema garante que cada cartela terá apenas um voto contabilizado. Mesmo que o votante possa atualizar o seu voto, apenas o último será apurado. O que o sistema não garante é a identidade de quem vota, pois ele assegura que a cartela é válida, mas não garante a identidade de quem faz uso da cartela.
Privacidade	**Parcialmente atendido:** é possível ao votante provar em quem votou devido à publicação dos TANs de votação, permitindo que o votante seja coagido. Para diminuir esta ameaça, os votantes tem a possibilidade de atualizar seu votos. Não é possível associar o voto ao votante a menos que se tenha acesso a cartela do votante. Além disso, não é possível contabilizar os votos antes do final da apuração.
Verificabilidade	**Parcialmente atendido:** existe a opção de verificabilidade individual caso sejam publicados os TANs de cada votante. Não existe verificabilidade universal.

Tabela 3: Tabela de avaliação do protocolo Code Sheets

6 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Existem diversos desafios que os sistemas de votação online devem encarar para poderem ser considerados seguros e utilizados em votações reais, o que deixa muitos receosos quanto à sua utilização em eleições governamentais ou com alto índice de criticidade. Muitas são as propostas que tentam contornar as principais ameaças quando se trata de votações conduzidas pela internet, mas também são grandes as dificuldades em desenvolver, testar e analisar estes sistemas, uma vez que existem diversas variáveis envolvidas durante a sua implementação, tais como a utilização de primitivas criptográficas, que podem se mostrar bastante complexas de serem implementadas e integradas ao sistemas.

O presente trabalho apresentou alguns sistemas de votação digital propostos na literatura que serviram de fundamento para a modelagem e desenvolvimento de um framework que auxiliasse na implementação de sistemas e protocolos de votação. Foram selecionados aqueles julgados mais relevantes, sendo posteriormente realizada a análise de domínio desses protocolos. Concluída esta etapa, foi iniciada a modelagem do framework seguida de sua implementação. A intenção do framework é reduzir o esforço necessário para o desenvolvimento de sistemas completos de votação digital, permitindo obter um elevado grau de reúso durante a implementação. Através da implementação desses sistemas, é possível analisá-los e verificar se atendem aos requisitos de segurança que prometem atender e buscar vulnerabilidades em sua concepção e implementação. Para verificar se o framework cumpre a função proposta, foram implementados três protocolos com diferentes requisitos de segurança, sendo possível observar que o framework foi adequado para o desenvolvimento de cada um deles, comprovando o baixo custo de desenvolvimento destes sistemas quando utilizado o framework. Através do presente trabalho, conseguimos constatar a sua importância por permitir a implementação desses sistemas a um custo reduzido, mostrando que ele consegue abarcar protocolos com características diferentes.

Outro objetivo era mostrar o uso do framework na avaliação de protocolos e mostrar sua viabilidade na implementação de sistemas reais. Foi implementado o protocolo Code Sheets proposto em (HELBACH; SCHWENK, 2007), onde os autores afirmam ser este adequado para votações tais como eleições de acionistas (shareholders). Após a sua implementação, foi realizado um conjunto de testes de forma a verificar possíveis falhas que pudessem ser exploradas por um atacante, tornando, por exemplo, o sistema inacessível

aos usuários. Conseguimos isto sobrecarregando o servidor com uma quantidade abusiva de requisições utilizando a mesma cartela de votação, o que sobrecarregou o banco de dados. A solução encontrada foi criar um mecanismo que impedisse que o votante enviasse mais de um voto dentro de um determinado período de tempo, restringindo o número de requisições por cartela. Implementamos esta solução e reconduzimos o mesmo conjunto de testes, verificando que dessa forma o sistema continuava funcionando normalmente.

Conseguimos verificar que o framework atendeu todos os objetivos almejados, mostrando-se adequado para o desenvolvimento de sistemas de votação reais, não apenas os mais simples que não atendem boa parte dos requisitos de segurança, mas também aqueles que se propõem a ser usados em votações onde o resultado deve ser confiável. O intuito com o framework não é apenas reduzir o esforço no desenvolvimento, mas também permitir que estes sistemas propostos possam ser mais facilmente testados e analisados, buscando vulnerabilidades e soluções para os problemas encontrados. Isto é possível pois o framework reduz o trabalho necessário para o desenvolvimento destes sistemas, graças ao alto grau de reúso oferecido. Uma vez tendo o sistema desenvolvido é possível conduzir um conjunto de testes para verificar quão confiável é o sistema, se ele atende os requisitos que propõe atender, analisar sua usabilidade e outros quesitos que sejam interessantes verificar. Dessa forma, podemos encontrar quais pontos devem ser melhorados e quais, talvez, não tenham sido bem planejados durante sua idealização, precisando ser revistos.

Através do conclusão deste trabalho, do desenvolvimento do framework, da implementação dos protocolos e de sua utilização para avaliar um sistema proposto na literatura, foi possível a publicação dos artigos (CABRAL; SILVA; CUNHA, 2013a) e (CABRAL; SILVA; CUNHA, 2013b).

6.0.1 Trabalhos futuros

O universo de sistemas de votação digital é bastante extenso, sendo uma tarefa árdua, senão impossível, abranger toda a gama de protocolos possíveis. Existem diversos aspectos que podem ser aprimorados na atual implementação do framework, de forma que este fique mais maduro e adequado ao desenvolvimento de sistemas de votação digital. Sendo assim, podemos citar:

- Testar protocolos de votação que utilizem mais de uma autoridade, idealmente localizadas em diferentes servidores. O framework atualmente

prevê a existência de várias autoridades, mas isto não foi efetivamente testado

- Implementar mecanismos de autenticação diversos, pois atualmente foi utilizada apenas a autenticação via tomcat, não fazendo uso de mecanismos tais como token e certificação digital

- Testar primitivas e recursos do framework que não foram utilizados por nenhum dos quatro protocolos implementados, tais como diferentes perfis de usuário, protocolos com mais de uma cédula por votante, auditoria universal, mais de uma urna por votação, entre outros.

- Elaborar métodos sistemáticos para avaliação de protocolos utilizando o framework como suporte, como, por exemplo, um roteiro de testes possíveis de serem conduzidos para verificar quais requisitos de segurança o sistema implementado atende.

- Avaliar o desempenho da implementação do protocolo Code Sheets quando utilizado por diferentes números de usuários verificando seu limite de sobrecarga, além da realização de testes de performances que não foram contemplados no presente trabalho

REFERÊNCIAS BIBLIOGRÁFICAS

ADIDA, B. Helios: Web-based open-audit voting. In: *Proceedings of the 17th conference on Security symposium*. Berkeley, CA, USA: [s.n.], 2008. v. 1, p. 335–348.

AKANDE, A. T. *The Importance of Voting in a Truly Democratic Society*. 2011. <http://www.globalpolitician.com/default.asp?26729-democracy-voting-elections>. Acessado em 12/06/2013.

AL., E. G. et. *Design Patterns: Elements of Reusable Object-Oriented Software*. [S.l.]: Addison Wesley, 1994. 395 p.

ARAUJO, R. S. dos S. *Protocolos Criptográficos para Votação Digital*. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, 2002.

AXIS, A. *Axis* . 2006. <https://axis.apache.org/axis/>. Acessado em 07/02/2014.

BAIARDI, F. et al. Seas: A secure e-voting applet system. Springer Berlin Heidelberg, v. 3233, p. 318–329, 2004.

BOUGHTON, C. Maintaining democratic values in e-voting with evacs. In: *Electronic Voting*. Castle Hofen, Bregenz, Austria: [s.n.], 2006. v. 1, p. 181–190.

BRAUN, N.; BRäNDLI, D. Swiss e-voting pilot projects: Evaluation, situation analysis and how to proceed. In: *Electronic Voting*. Castle Hofen, Bregenz, Austria: [s.n.], 2006. v. 1, p. 27–36.

CABRAL, P. D.; SILVA, R. P. e.; CUNHA, R. S. d. Framework for digital voting systems. In: *The 25th International Conference on Software Engineering & Knowledge Engineering*. Boston,USA: [s.n.], 2013. p. 715–720.

CABRAL, P. D.; SILVA, R. P. e.; CUNHA, R. S. d. An object-oriented framework for digital voting. In: *SERP'13 - The 2013 International Conference on Software Engineering Research and Practice*. Las Vegas, USA: [s.n.], 2013. p. 189–196.

CASTLE, B. *The Legion of the Bouncy Castle* . 2004. <https://www.bouncycastle.org/>. Acessado em 07/03/2014.

CHAUM, D. *SureVote*. 2001. <http://www.surevote.com/>. Acessado em 22/06/2013.

CHAUM, D. L. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, ACM, New York, NY, USA, v. 24, n. 2, p. 84–90, fev. 1981. ISSN 0001-0782. <http://doi.acm.org/10.1145/358549.358563>.

CHEN, J. *Verifiable Mixnets Techniques and Prototype Implementation*. Tese (Doutorado) — Darmstadt University of Technology, Darmstadt, Alemanha, 2007.

COMMUNITY, H. *Hibernate* . 2013. <http://www.hibernate.org/>. Acessado em 07/09/2013.

CRANOR, L. F.; CYTRON, R. K. Sensus: A security-conscious electronic polling system for the internet. IEEE Computer Society, Washington, DC, USA, p. 561–, 1997. <http://dl.acm.org/citation.cfm?id=938435.938629>.

ELGAMAL, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *Information Theory, IEEE Transactions on*, v. 31, n. 4, p. 469–472, Jul 1985. ISSN 0018-9448.

FAYAD, M. E.; SCHMIDT, D. C.; JOHNSON, R. E. *Building Application Frameworks*. [S.l.: s.n.], 1999.

FIAT, A.; SHAMIR, A. How to prove yourself: Practical solutions to identification and signature problems. In: ODLYZKO, A. (Ed.). *Advances in Cryptology - CRYPTO' 86*. [S.l.]: Springer Berlin Heidelberg, 1987, (Lecture Notes in Computer Science, v. 263). p. 186–194. ISBN 978-3-540-18047-0.

FUJIOKA, A.; OKAMOTO, T.; OHTA, K. A practical secret voting scheme for large scale elections. In: SEBERRY, J.; ZHENG, Y. (Ed.). *Advances in Cryptology - AUSCRYPT 92*. [S.l.]: Springer Berlin Heidelberg, 1993, (Lecture Notes in Computer Science, v. 718). p. 244–251.

GOLDWASSER, S.; JARECKI, S.; LYSYANSKAYA, A. *Cryptography and Information Security Group Research Project: Threshold Cryptology*. <http://groups.csail.mit.edu/cis/cis-threshold.html>. Acessado em 16/08/2013.

GOLDWASSER, S.; MICALI, S.; RACKOFF, C. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, v. 18, n. 1, p. 186–208, fev. 1989. ISSN 0097-5397. <http://dx.doi.org/10.1137/0218012>.

HEATHER, J.; LUNDIN, D. The append-only web bulletin board. In: *Formal Aspects in Security and Trust*. Guildford, Surrey, UK: [s.n.], 2008. v. 1, p. 242–256.

HELBACH, J.; SCHWENK, J. Secure internet voting with code sheets. In: ALKASSAR, A.; VOLKAMER, M. (Ed.). *E-Voting and Identity*. [S.l.]: Springer Berlin Heidelberg, 2007, (Lecture Notes in Computer Science, v. 4896). p. 166–177.

JMETER, A. *JMeter* . 2014. <https://jmeter.apache.org/>. Acessado em 07/02/2014.

JOHNSON, R. E. *How to Design Frameworks* . [S.l.]: OOPSLA - Object-Oriented Programming, Systems, Languages & Applications, 1993.

KIM, Y. et al. Packetscore: statistics-based overload control against distributed denial-of-service attacks. In: *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*. [S.l.: s.n.], 2004. v. 4, p. 2594–2604 vol.4. ISSN 0743-166X.

KRUMMENACHER, R. Implementation of a web bulletin board for e-voting applications.

LABSEC. *LibCryptoSec* . 2010. <https://projetos.labsec.ufsc.br/libcryptosec>. Acessado em 07/03/2014.

LUNDIN, D. Towards trustworthy elections. In: CHAUM, D. et al. (Ed.). Berlin, Heidelberg: Springer-Verlag, 2010. cap. Component Based Electronic Voting Systems, p. 260–273. ISBN 3-642-12979-X, 978-3-642-12979-7. <http://dl.acm.org/citation.cfm?id=2167913.2167929>.

MADISE, U.; MARTENS, T. E-voting in estonia 2005: The first practice of country-wide binding internet voting in the world. In: *Electronic Voting*. Castle Hofen, Bregenz, Austria: [s.n.], 2006. v. 1, p. 15–26.

MOLYNEAUX, I. *The Art of Application Performance Testing*. [S.l.]: O'Reilly Media, Inc., 2009. 158 p.

OLIVA, A. G. et al. Votación electrónica basada en criptografía avanzada (proyecto votescript). Mérida, Venezuela, 2002.

PAILLIER, P. Public-key cryptosystems based on composite degree residuosity classes. In: STERN, J. (Ed.). *Advances in Cryptology - EUROCRYPT '99*. [S.l.]: Springer Berlin Heidelberg, 1999, (Lecture Notes in Computer Science, v. 1592). p. 223–238. ISBN 978-3-540-65889-4.

POSTGRESQL. *PostgreSQL* . 2013. <http://www.postgresql.org/>. Acessado em 07/09/2013.

QUISQUATER, J.-J. et al. How to explain zero-knowledge protocols to your children. In: BRASSARD, G. (Ed.). *Advances in Cryptology - CRYPTO' 89 Proceedings*. [S.l.]: Springer New York, 1990, (Lecture Notes in Computer Science, v. 435). p. 628–631. ISBN 978-0-387-97317-3.

RAY, I.; RAY, I.; NARASIMHAMURTHI, N. An anonymous electronic voting protocol for voting over the internet. In: *Advanced Issues of E-Commerce and Web-Based Information Systems, WECWIS 2001, Third International Workshop on.* [S.l.: s.n.], 2001. p. 188–190.

REITER, M. K.; WANG, X. *Fragile Mixing*. 2004.

RIVEST, R. *Lecture 18: Mix-net Voting Systems*. April 2004. <http://courses.csail.mit.edu/6.897/spring04/L18.pdf>.

RIVEST, R. The threeballot voting system. 2006.

RODRIGUES-FILHO, J.; ALEXANDER, C.; BATISTA, L. E-voting in brazil - the risks to democracy. In: *Electronic Voting*. Castle Hofen, Bregenz, Austria: [s.n.], 2006. v. 1.

SANTIN, A.; COSTA, R.; MAZIERO, C. A three-ballot-based secure electronic voting system. *Security Privacy, IEEE*, v. 6, n. 3, p. 14–21, 2008.

SHAMOS, M. I. *Paper v. Electronic Voting Records - An Assessment*. 2011. <http://euro.ecom.cmu.edu/people/faculty/mshamos/paper.htm>. Acessado em 04/2004.

Sá, C. R. F. de. Voto - direito de ser cidadão. In: *Revista Paraná Eleitoral*. Brazil: [s.n.], 1999. v. 32.

THEP. *Thep* . 2013. <https://code.google.com/p/thep/>. Acessado em 07/03/2014.

TOMCAT, A. *Apache Tomcat 7* . 2013. <http://tomcat.apache.org/>. Acessado em 07/09/2013.

VOLKAMER, M.; HUTTER, D. From legal principles to an internet voting system. In: *Electronic Voting*. Lake of Constance, Austria: [s.n.], 2004. v. 1, p. 111–120.

WIKSTRöM, D. *Verificatum* . 2013. <http://tomcat.apache.org/>. Acessado em 07/03/2014.

WOLF, P. *Countries with e-voting projects* . 2010. <http://aceproject.org/ace-en/focus/e-voting/countries>. Acessado em 29/07/2013.

WU, C.-K.; SANKARANARAYANA, R. Internet voting: concerns and solutions. *First International Symposium on Cyber Worlds*, v. 1, p. 261–266, 2002.

APÊNDICE A – Primitivas criptográficas e outras funções

Para garantir a exatidão e a honestidade em um sistema de votação digital, é comum o uso de mecanismos conhecidos como primitivas criptográficas. Elas são responsáveis por ajudar a garantir a privacidade dos votantes, anonimato das cédulas, integridade das informações e dificultar a manipulação e a adulteração do resultados, cédulas e outras partes sensíveis do sistema. Abaixo segue a explicação das principais primitivas utilizadas nos sistemas de votação estudados.

A.1 BULLETIN BOARD

Muitas vezes durante o processo de votação é necessário divulgar alguns dados para os participantes, tais como cédulas recebidas, resultado final da eleição, provas de operações criptográficas, etc. Como muitas dessas informações são sensíveis, ou seja, deve-se ter confiança em sua origem e seu conteúdo, é necessário um mecanismo que garanta estes requisitos. A Bulletin Board (BB) tem justamente este propósito, sendo um espaço público para divulgar todo tipo de informação que seja relevante para o processo de votação. Qualquer participante é capaz de ler as mensagens publicadas, mas apenas entidades autorizadas são capazes de escrever na BB.(CHEN, 2007)

Existem várias propostas de Bulletin Board, algumas são distribuídas e outras fazem uso de apenas uma autoridade que centraliza toda a informação. (KRUMMENACHER,) A implementação utilizada no framework foi baseada no trabalho de Heather e Lundin (HEATHER; LUNDIN, 2008), onde eles identificam os requisitos para uma Bulletin Board centralizada. No modelo implementado existem três entidades: a bulletin board, os leitores e os escritores autorizados. Toda mensagem só pode ser publicada na BB pelos escritores, que devem assiná-la para assegurar sua identidade. Toda mensagem também deve ser anexada ao final das mensagens anteriores, contendo o hash de todas as publicações passadas, de forma que se mantenha um histórico consistente. Isto também impede que se altere ou exclua mensagens já divulgadas e que não seja possível inserir novas mensagens no meio do histórico sem que seja possível detectar a inconsistência. A vantagem da Bulletin Board é que ela garante a autenticidade das informações publicadas e a consistência das mensagens. Ela não assegura que as informações publicadas estejam corretas, mas garante que o que foi publicado foi feito por uma entidade autorizada.

A.2 CRIPTOGRAFIA SIMÉTRICA

A criptografia simétrica foi uma das primeiras técnicas criptográficas desenvolvidas, sendo ainda hoje bastante conhecida e empregada. Sua utilização depende da existência de uma chave única, utilizada tanto para cifrar quando para decifrar as informações. Isto implica que qualquer um que tenha acesso à chave conseguirá obter qualquer mensagem que foi cifrada através dela. Como esta chave tem que ser compartilhada entre os interessados, ou seja, aqueles que devem ter acesso à informação sigilosa, uma das grandes dificuldades da criptografia simétrica reside justamente na forma de distribuição da chave. Esta dificuldade advém dos canais nem sempre serem seguros, podendo ser possível interceptar a chave e obter a informação cifrada. Outra dificuldade é na hora de armazená-la de forma segura, sem que pessoas não autorizadas tenham acesso a ela. Esses dois aspectos são chaves na criptografia simétrica, residindo aí a sua segurança: como compartilhar a chave entre quem cifrou a mensagem e entre quem deve ter acesso a ela, e como armazenar esta chave de forma confiável, sem que terceiros tenham acesso.

A.3 CRIPTOGRAFIA ASSIMÉTRICA

A criptografia assimétrica surge como uma tentativa de contornar um dos problemas encontrados na criptografia simétrica: a distribuição da chave. Diferente da criptografia simétrica, onde é utilizada apenas uma chave, na assimétrica é necessário o uso de duas, sendo que elas tem uma relação matemática única entre si. Isto significa que o que uma cifra apenas a outra decifra e vice-versa. Uma das duas é escolhida para ser a chave privada, e a outra a pública. A privada deve permanecer em segredo, sendo acessível apenas ao seu dono. Já a chave pública pode ser amplamente divulgada, se tornando acessível a quem desejar, eliminando o problema de compartilhamento de chaves. Outra vantagem deste modelo é que ele permite não só a cifragem de documentos, mas também a sua assinatura. Caso o documento seja cifrado com a chave pública, apenas quem tiver posse da chave privada (ou seja, seu dono) terá acesso. Isso funciona para manter o sigilo dos dados, pois temos certeza que apenas quem detém a chave privada terá a possibilidade de ler a mensagem. Caso o documento seja cifrado com a chave privada, todos que tenham acesso a chave pública conseguirão decifrar a mensagem. Como a chave pública é, como o próprio nome diz, pública, todos podem decifrar a mensagem. Com isto temos o que é conhecido por assinatura digital, pois sabemos que a única pessoa que poderia cifrar aquele documento é justamente o dono da chave privada, o que garante sua identidade. Ou seja, a criptogra-

fia assimétrica permite garantir tanto a confidencialidade e não repúdio das mensagens, quanto sua autenticidade.

A.4 REDE DE MISTURA

As redes de mistura foram idealizadas por David Chaum em 1981, como uma proposta para impedir a associação dos remetentes e destinatários no sistema de correio eletrônico (CHAUM, 1981). Hoje são comumente utilizadas em sistemas de votação digital, para impedir que se associe a cédula ao votante. Para isto, a ordem de chegada das mensagens é escondida através do seu encaminhamento em pacotes de dados do mesmo tamanho e em ordem diferente da recebida.

As redes de mistura geralmente consistem em vários servidores para aumentar a robustez do sistema (RIVEST, 2004) e fazem uso de criptografia assimétrica, normalmente utilizando ElGamal no caso das redes de recifragem. Antes de iniciar o processo, a mensagem é cifrada com a chave pública do destinatário, de forma que apenas ele consiga obter seu conteúdo. Também é anexada à mensagem o endereço de destino e tudo é cifrado com a chave pública do próximo servidor. Cada servidor da rede recebe as mensagens e as embaralha, de forma que a ligação entre o destinatário e a mensagem seja quebrada. Em alguns casos eles também recifram as mensagens utilizando ElGamal. Em seguida as mensagens misturadas são enviadas para o próximo servidor da rede até que estas cheguem ao destino.

Existem diversos tipos de redes de mistura, podendo ser classificadas como rede de mistura de decifragem ou de recifragem (RIVEST, 2004). Também existem as redes de mistura verificáveis, o que é desejável em sistemas de votação digital, pois elas permitem verificar a corretude do trabalho de cada misturador. (CHEN, 2007)

A.5 ELGAMAL

O ElGamal é um algoritmo de criptografia assimétrico baseado na troca de chaves de Diffie-Helman, desenvolvido pelo estudioso egípcio Taher Elgamal em 1984. Sua segurança reside na dificuldade de se computar logaritmos discretos (ELGAMAL, 1985). Sua importância em sistemas de votação digital é por possuir a propriedade de homomorfismo, importante para o desenvolvimento da rede de misturadores.

A criptografia ElGamal é probabilística, ou seja, um único texto pode resultar em diversos textos cifrados diferentes. Outra característica é que o tamanho do texto cifrado aumenta na proporção de 2:1, sendo utilizado apenas para cifrar coisas pequenas, como chaves privadas. O Elgamal faz uso de três componentes: um gerador, um algoritmo para cifrar e um para decifrar. Cada assinatura ou encriptação requer um valor randômico (nounce, ou chave de sessão) k. O conhecimento de pelo menos duas mensagens ou assinaturas como o mesmo k permite a dedução da chave privada.

A.6 CRIPTOGRAFIA LIMIAR

A criptografia limiar é usada para proteger informações compartilhando a chave entre diferentes computadores, de forma que um número determinado precise colaborar para decifrar a mensagem. A intenção é criar um sistema tolerante a falhas, de forma que atendam a dois requisitos: que não seja possível a um grupo de computadores corruptos obter a informação caso o tamanho do grupo seja menor que o limiar estipulado. E que seja necessário a colaboração de um número igual ou superior ao limiar estipulado para obter a informação secreta. Ou seja, é necessário um balanço entre o valor do limiar, de forma que ele não seja baixo demais permitindo que um pequeno número de computadores corruptos consigam obter a informação, nem alto demais, de forma que, caso algum computador não queira cooperar, por ser corrupto, seja impossível obter a informação secreta. (GOLDWASSER; JARECKI; LYSYANSKAYA,)

A.7 PROVA DE CONHECIMENTO ZERO

A prova de conhecimento zero foi introduzida por Goldwasser (GOLDWASSER; MICALI; RACKOFF, 1989) em 1985. Seu funcionamento se dá normalmente entre dois participantes, um verificador e um provador. O primeiro deseja ter certeza que o outro tem conhecimento de determinada informação, sem revelá-la. Para isto, o verificador desafia o provador com algumas provas, checando suas respostas. Após um determinado número de tentativas, o verificador conclui se o provador tem ou não a informação que afirma ter. Um exemplo bastante referenciado da prova de conhecimento zero foi formulado por Jean-Jacques Quisquater (QUISQUATER et al., 1990), e mostra uma pessoa que tenta provar a outra que conhece a palavra secreta

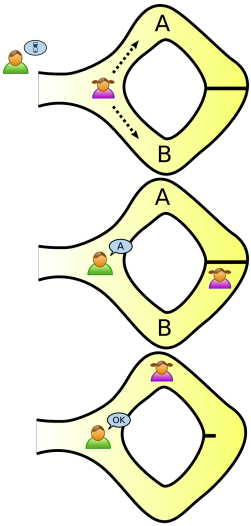

Figura 44: Exemplo da prova de conhecimento zero (QUISQUATER et al., 1990).

para abrir uma porta. No exemplo existe um túnel como o da Figura 44, com uma porta no meio que impede a passagem de um lado para o outro e que só é aberta quando a palavra correta é dita. Emprestando os nomes utilizados por Quisquater, Peggy é quem tenta provar que tem conhecimento da palavra secreta, e Victor verifica a autenticidade da afirmação. Peggy entra por um dos lados do túnel, A ou B, sem que Victor veja por onde ela entrou. Ele então pede para que ela sai por um dos lados escolhido aleatoriamente por Victor. Caso Peggy tenha conhecimento da palavra secreta, ela sempre conseguirá sair pelo lado requisitado. E quanto maior o número de tentativas, maior a certeza que Victor terá a respeito disso. Caso contrário, quanto maior for o número de desafios, maior a probabilidade de descobrir que Peggy na verdade não conhece a palavra secreta.

Dessa forma é possível que o verificador, no caso Victor, tenha uma boa garantia de que o provador, Peggy, conhece a palavra secreta, sem que seja preciso que ela a revele. As provas de conhecimento zero normalmente seguem este esquema de desafio e resposta, onde ao final o verificador aceita ou não as provas. A prova de conhecimento zero normalmente tem que atender a dois requisitos: o verificador não obtém nada além da comprovação de que o provador detém as informações que diz ter, não sendo possível para

ele executar a prova de conhecimento zero com outros participantes. E segundo, terceiros não podem obter informações dos participantes que possam ser futuramente exploradas.

Segundo Reiter e Wan (REITER; WANG, 2004) a prova de conhecimento zero deve satisfazer três propriedades:

- **Completude:** permite provar com grande segurança que o provador é honesto

- **Robustez:** é robusto se permite provar uma informação falsa com uma probabilidade insignificante

- **Conhecimento zero:** o verificador não pode obter nada além da confirmação de que o provador tem conhecimento da informação

Ou seja, o verificador deve ter confiança através do protocolo de prova de conhecimento zero que o provador não consiga provar desonestamente algo, e que caso algo seja provado, que seja com um grande índice de confiança. Além disto, não deve haver vazamento de informação, de forma que seja possível ao verificador e a terceiros descobrir a informação ou dados suficientes para simular que eles a conhecem. A utilidade da prova de conhecimento zero também é utilizada para provar a identidade de alguém, pois permite que a entidade prove ter uma informação que apenas ela poderia ter.

Além das provas de conhecimento zero interativas, como as descritas acima, existem as não-interativas. Este tipo de protocolo permite que não seja necessário que os dois participantes estejam online ao mesmo tempo, o que diminui a interação entre ambos. A heurística Fiat-Shamir (FIAT; SHAMIR, 1987) permite transformar um protocolo interativo em uma prova não interativa.

A.8 FUNÇÃO HASH

Uma função hash é um algoritmo responsável por mapear dados de tamanho variado em um conjunto de dados de tamanho fixo, funcionando como uma referência ao valor original. A função hash é conhecida por ser referencialmente transparente, ou seja, estável. A entrada determinará a saída, e esta será sempre igual para a mesma entrada, independente do número de vezes que se execute o algoritmo. Em alguns casos pode haver colisões, o que significa que entradas diferentes resultarão em saídas idênticas. Isto acontece pois é impossível mapear uma grande quantidade de dados para uma quantidade

menor sem que haja resultados idênticos.

Funções hash também são conhecidas por serem destrutivas, o que significa que a informação original é perdida quando se gera seu hash. Isto impede que se descubra a entrada original apenas conhecendo sua saída. Isto é útil para guardar senhas de usuários, permitindo armazenar apenas o valor hash, e não a senha inteira, o que possibilitaria seu conhecimento em caso de invasão no banco de dados.

Em relação a votação digital sua importância se dá de diversas maneiras, como, por exemplo, identificar a cédula do votante sem demonstrar em quem ele votou, além de ser bastante útil na hora de identificar valores que não se deseja conhecer o conteúdo.

A.9 PAILLIER

O Paillier foi inventado em 1999 por Pascal Paillier (PAILLIER, 1999), sendo um algoritmo assimétrico para criptografia de chave pública. Uma de suas características é sua propriedade homomórfica, bastante utilizada em protocolos de votação digital por permitir a soma dos votos ainda cifrados.

APÊNDICE B – Diagramas complementares

Este capítulo apresenta alguns dos diagramas desenvolvidos durante a modelagem do framework, representando casos de uso específicos do sistema (Figuras 45 e 46).

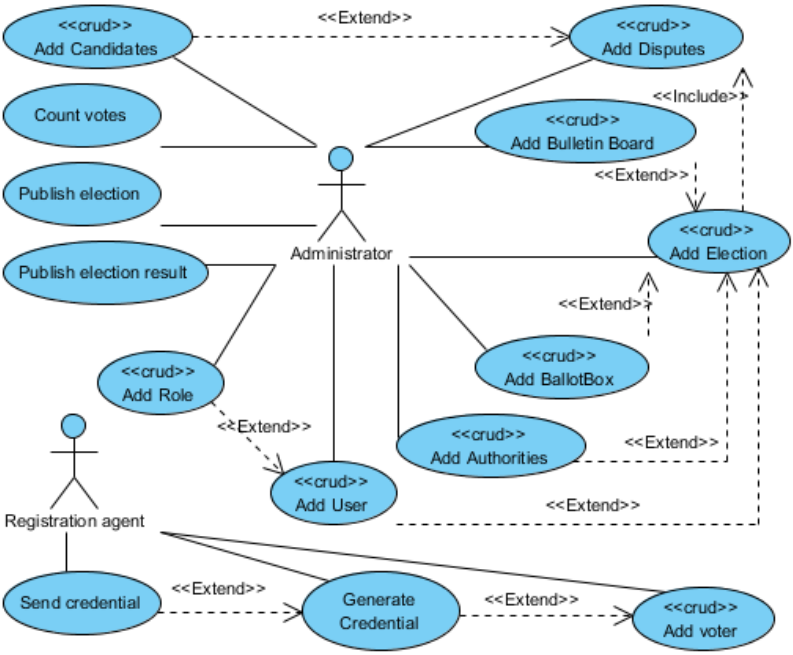

Figura 45: Diagrama de casos de uso mostrando os papéis do administrador e do escrivão

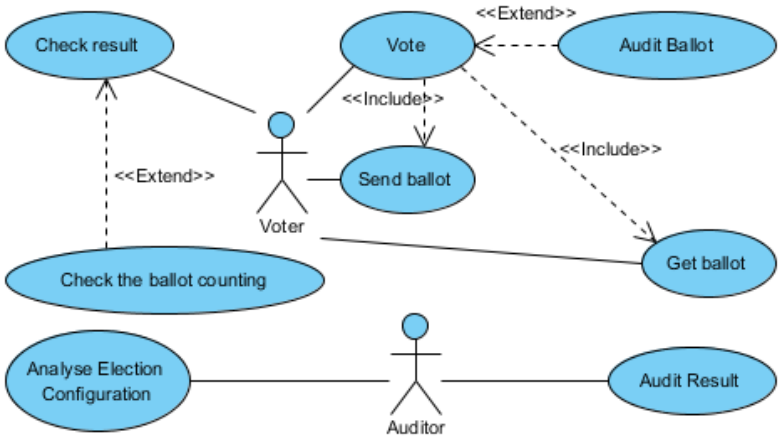

Figura 46: Diagrama de casos de uso mostrando os papéis do votante e do auditor

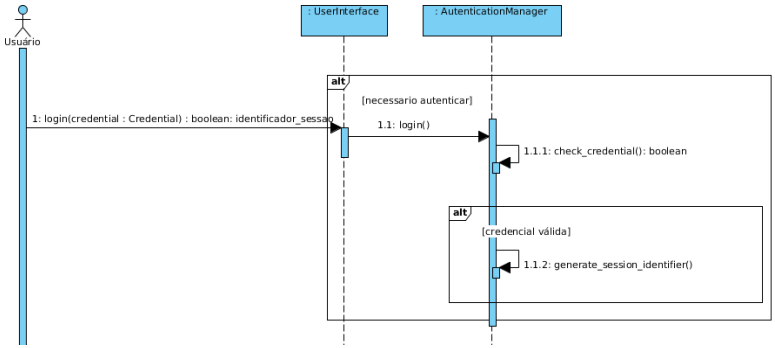

Figura 47: Diagrama de sequência que mostra a autenticação dos usuários no sistema

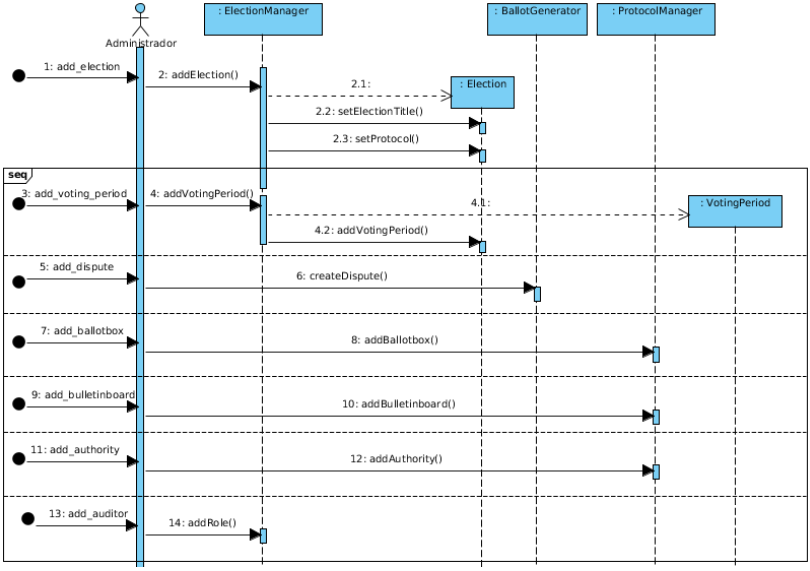

Figura 48: Diagrama de sequência mostrando como cadastrar uma nova eleição

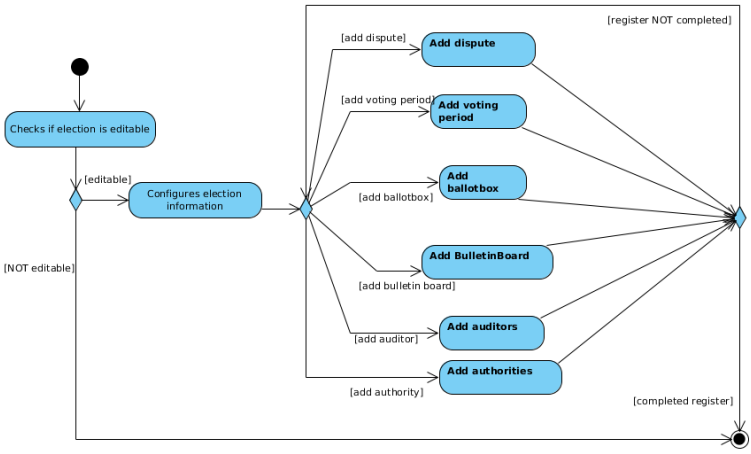

Figura 49: Diagrama de atividade mostrando como cadastrar uma eleição

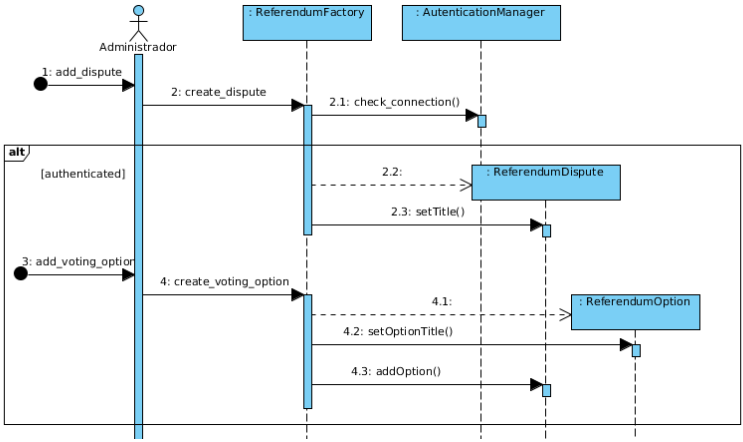

Figura 50: Diagrama de sequência mostrando como adicionar uma nova disputa do tipo referendo

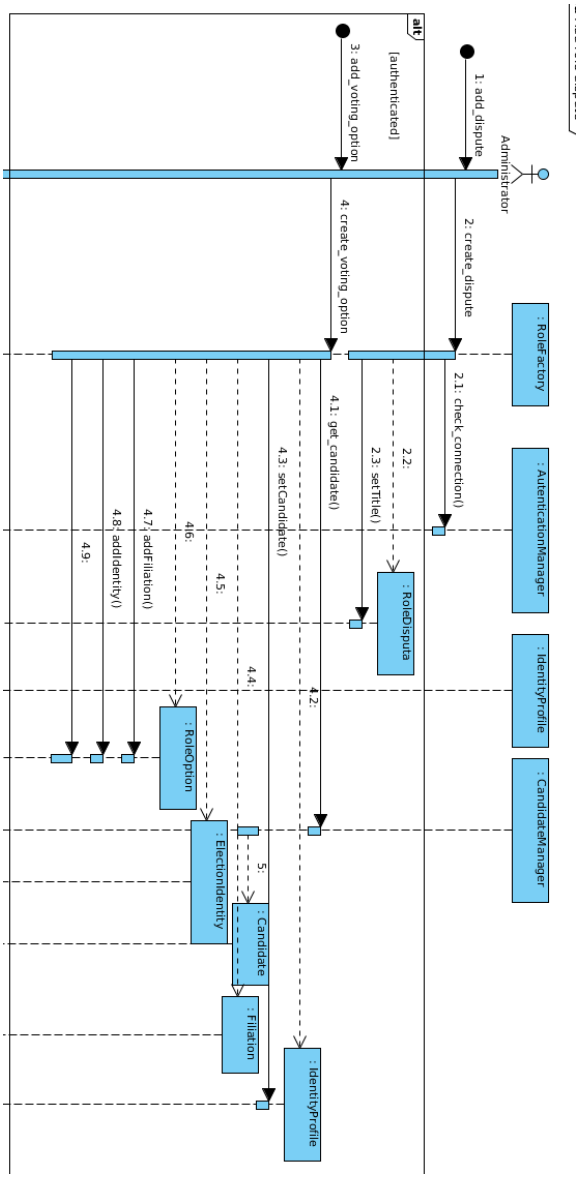

Figura 51: Diagrama de sequência mostrando como adicionar uma nova disputa do tipo cargo

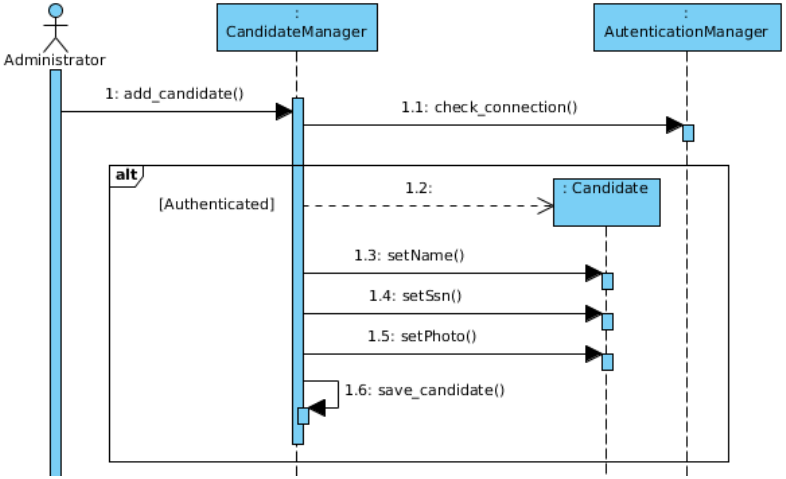

Figura 52: Diagrama de sequência mostrando o processo de adicionar candidatos

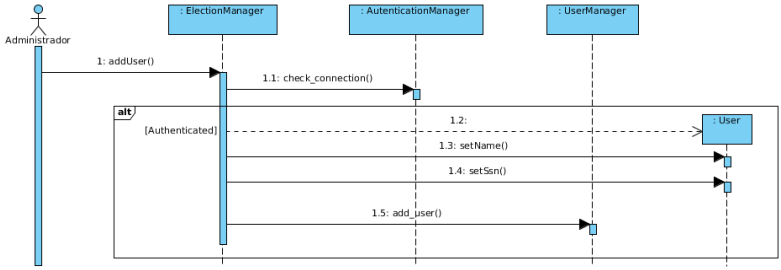

Figura 53: Diagrama de sequência mostrando como adicionar usuários ao sistema

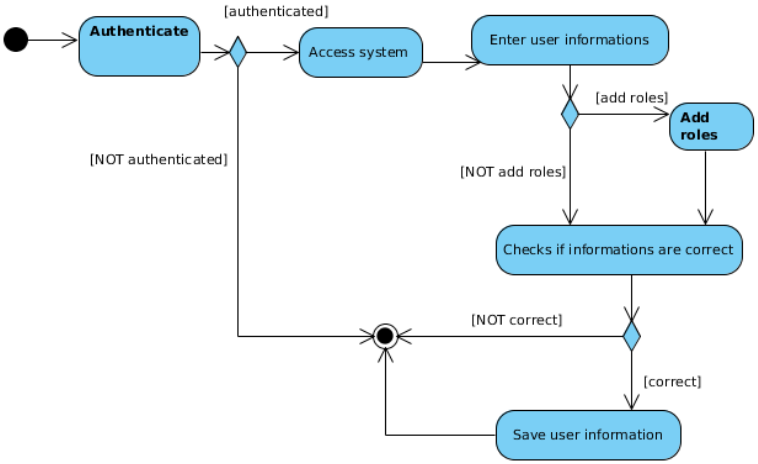

Figura 54: Diagrama de atividade mostrando como adicionar usuários ao sistema

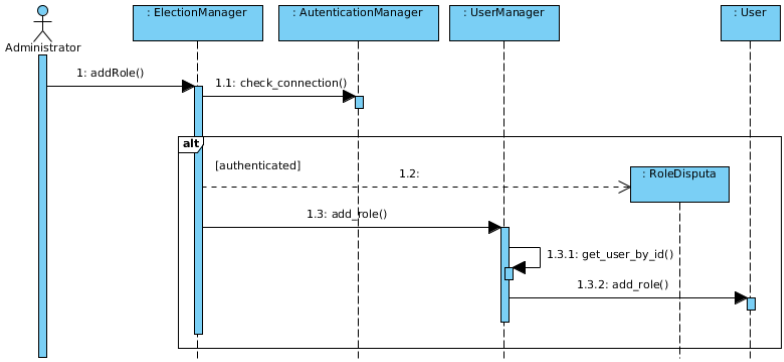

Figura 55: Diagrama de sequência mostrando como atribuir papéis a determinado usuário

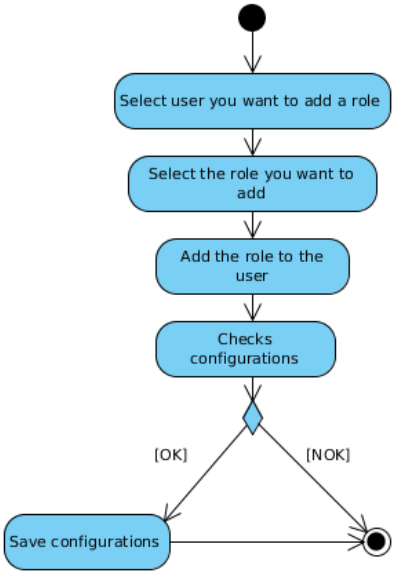

Figura 56: Diagrama de atividade mostrando como atribuir papéis a determinado usuário

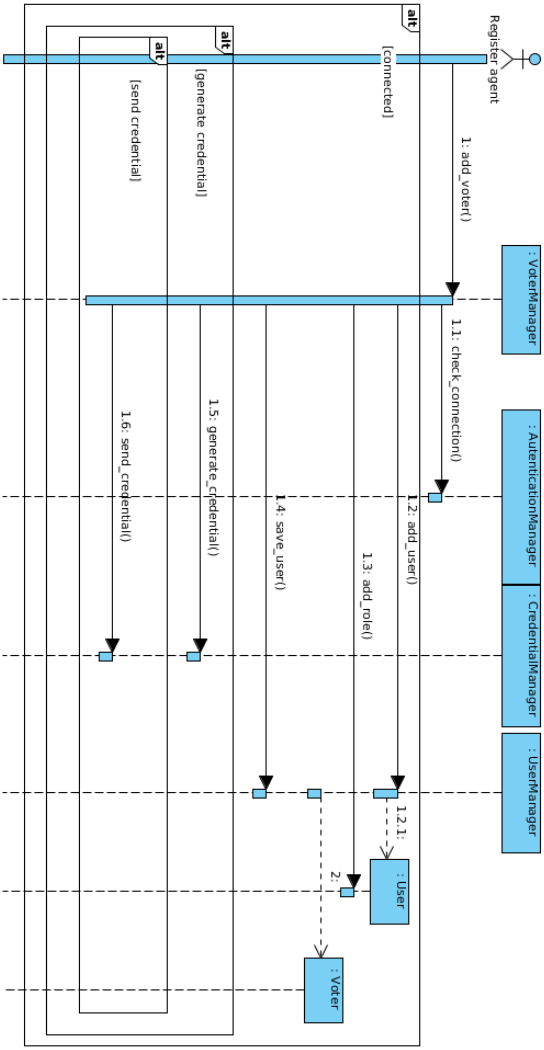

Figura 57: Diagrama de sequência mostrando como adicionar um votante a determinada eleição

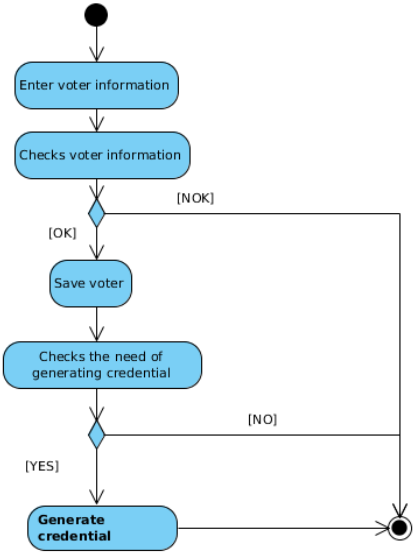

Figura 58: Diagrama de atividade mostrando como adicionar um votante a determinada eleição

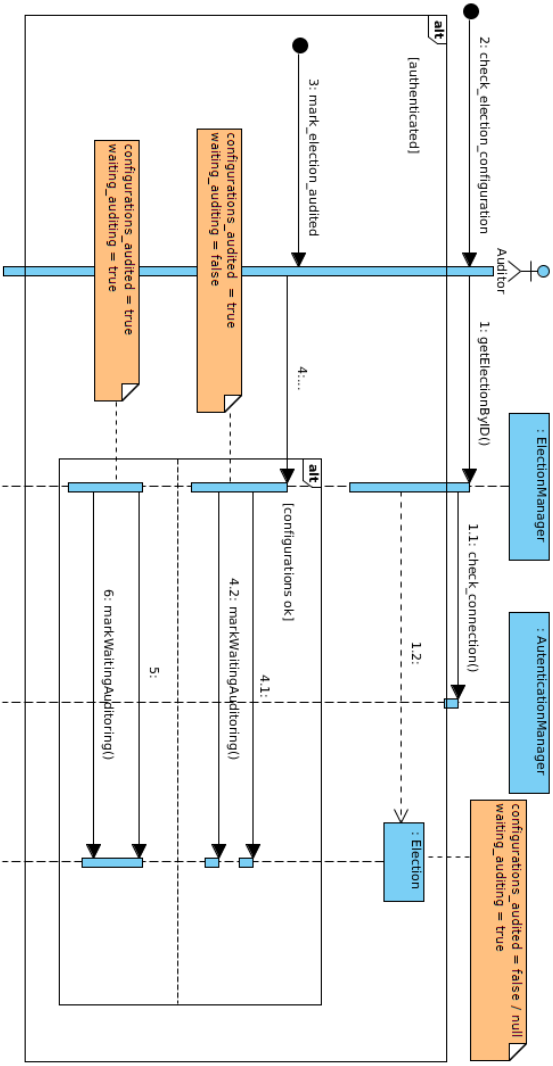

Figura 59: Diagrama de sequência mostrando o processo de auditoria

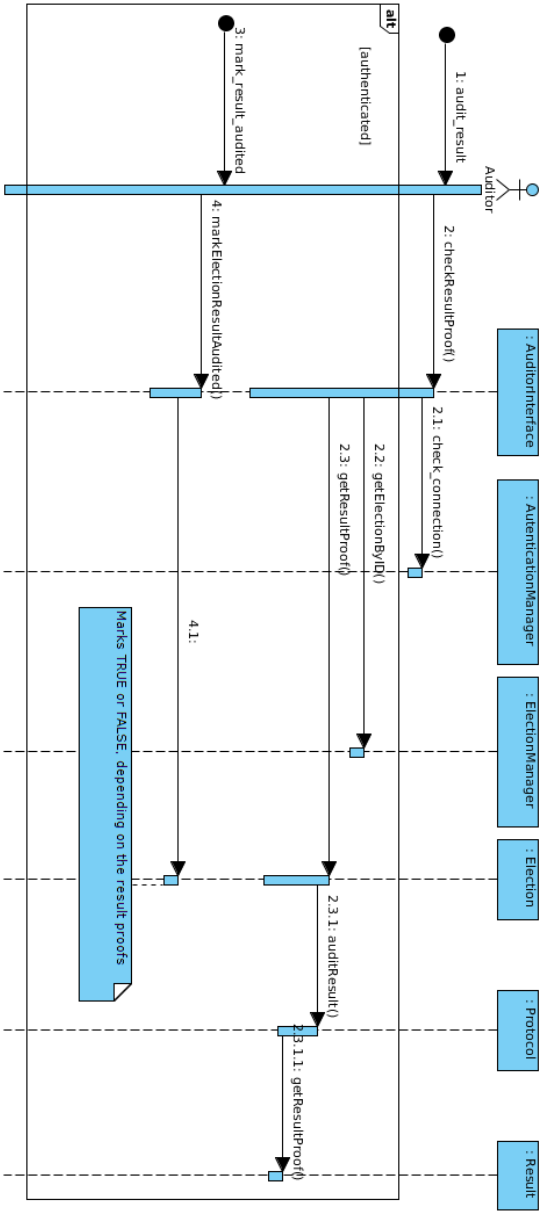

Figura 60: Diagrama de sequência mostrando o processo de auditoria do resultado da eleição

APÊNDICE C – Diagramas de visão geral de interação

Diagramas 61, 62, 63, 64, 65, 66 e 67 fazem parte do conjunto de diagramas de interação desenvolvidos durante a modelagem do framework.

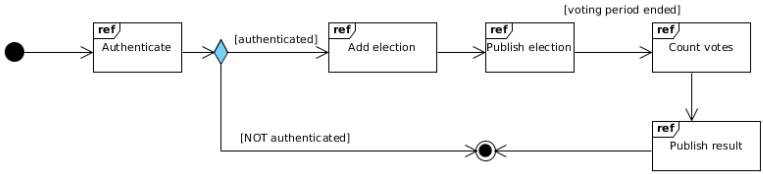

Figura 61: Diagrama de interação mostrando as etapas necessárias para o cadastro de um eleição

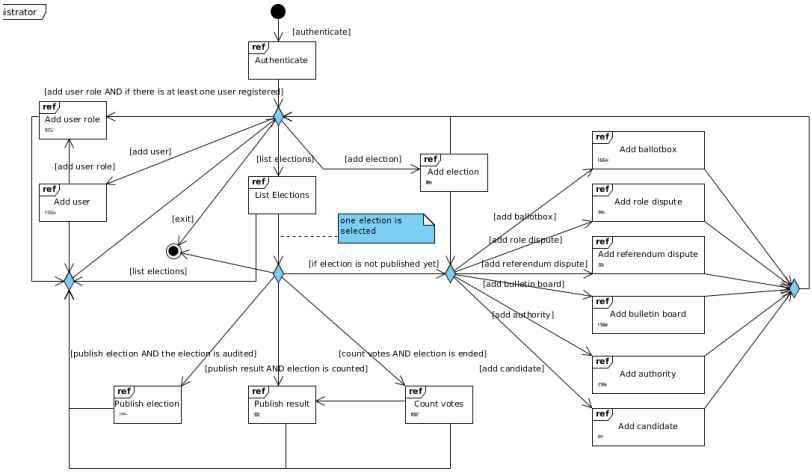

Figura 62: Diagrama de interação mostrando as atribuições do administrador e como elas se relacionam

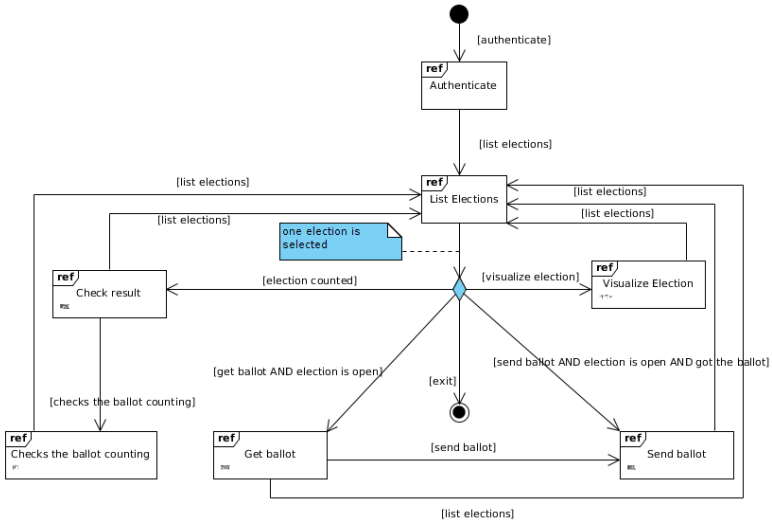

Figura 63: Diagrama de interação mostrando as atribuições do votante e como elas se relacionam

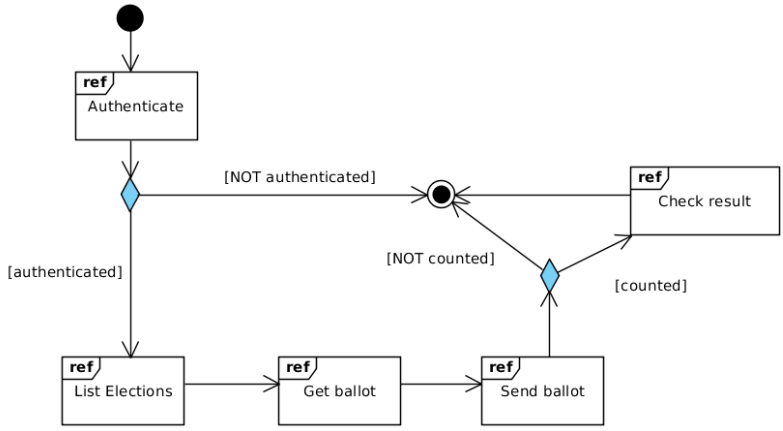

Figura 64: Diagrama de interação mostrando as etapas necessárias para a atividade de votar

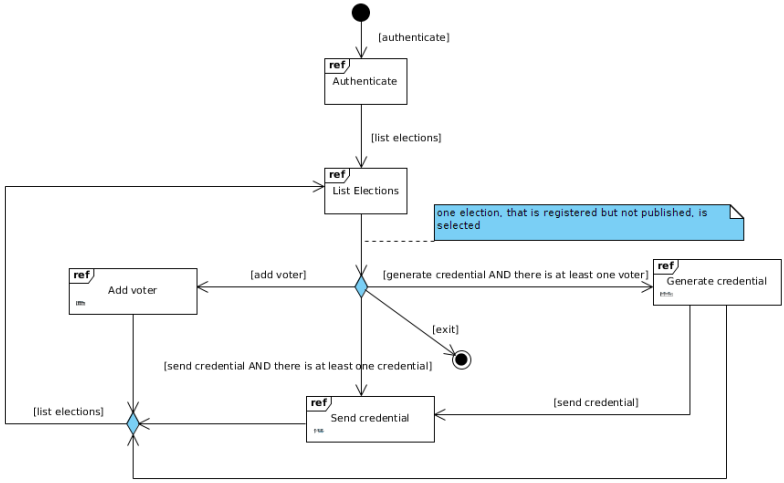

Figura 65: Diagrama de interação mostrando as atribuições do agente de registro e como elas se relacionam

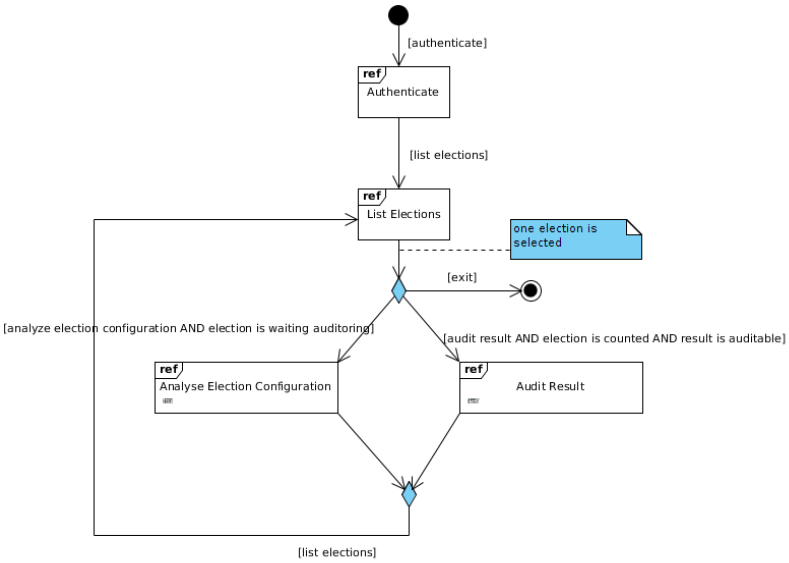

Figura 66: Diagrama de interação mostrando as atribuições do auditor e como elas se relacionam

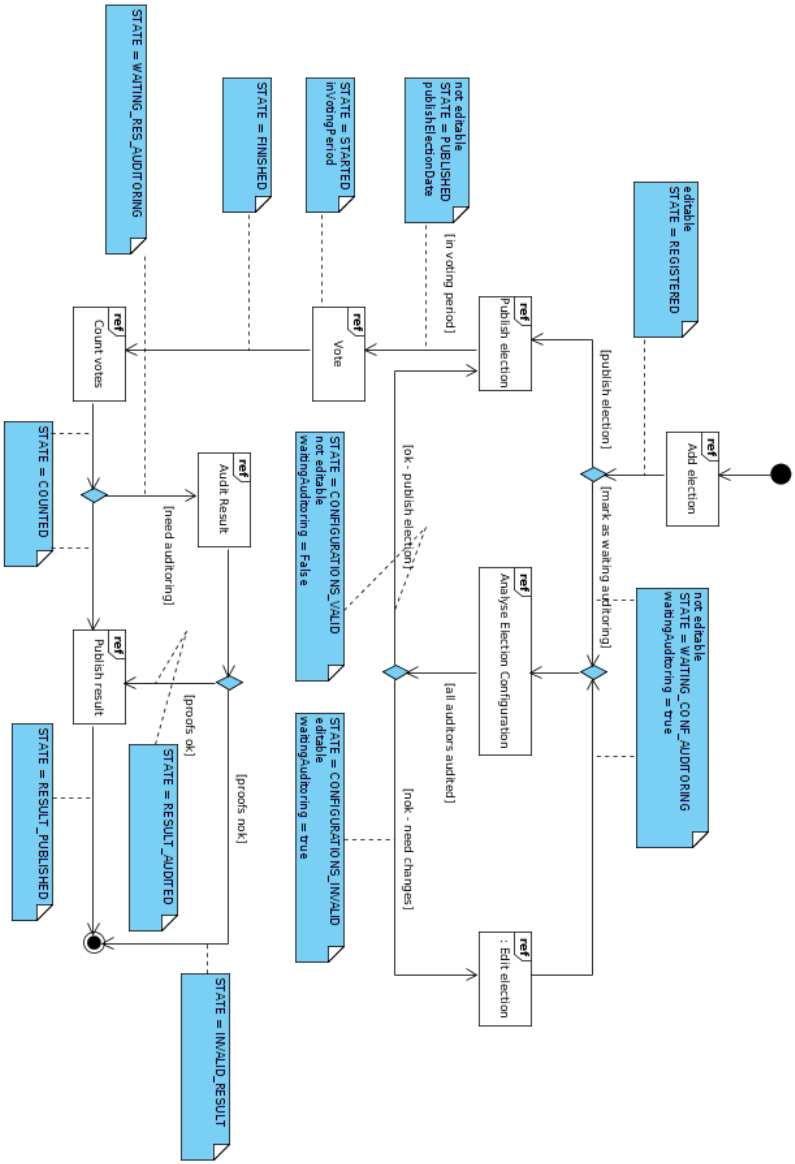

Figura 67: Diagrama de interação mostrando as etapas necessárias para a auditoria de um eleição