UNIVERSIDAD DE LA REPÚBLICA URUGUAY

FACULTAD DE INGENIERÍA

Voto electrónico en Uruguay utilizando blockchain

Informe de Proyecto de Grado presentado por

Juan Ignacio Medina Cruz, Bruno Ferrando Bergeret

en cumplimiento parcial de los requerimientos para la graduación de la carrera de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de la República

Supervisores

Carlos Luna
Pablo Blanco

Montevideo, 16 de abril de 2026

Agradecimientos

A nuestras familias y amigos, por estar siempre, por el apoyo y la paciencia durante todo este proceso. El camino de la carrera fue largo; aprendimos mucho y supimos perseverar hasta el final. A nuestros tutores, por tomar una idea que surgió de nosotros para el proyecto de grado, por la guía brindada, el tiempo dedicado y las sugerencias que nos permitieron llevar adelante y mejorar este trabajo.

Resumen

El presente proyecto aborda la problemática del voto electrónico en Uruguay, con especial énfasis en la aplicación de tecnologías basadas en blockchain para la implementación de urnas electrónicas. El objetivo principal es analizar la viabilidad de utilizar una red blockchain privada como soporte tecnológico para un sistema de votación que combine transparencia y privacidad de los votos.

Para ello, se desarrolló un prototipo funcional empleando *Hyperledger Fabric*, una plataforma de blockchain de tipo permisionado orientada al ámbito empresarial. En particular, se utilizó su mecanismo de colecciones de datos privados (Private Data Collections), que permite restringir el acceso a información sensible únicamente a los nodos autorizados dentro de la red. Este enfoque resulta especialmente relevante en el contexto electoral, ya que garantiza la confidencialidad del voto al tiempo que preserva la auditabilidad de las transacciones.

Durante la etapa de desarrollo se implementaron distintos chaincodes (contratos inteligentes) que representan las funciones esenciales del proceso electoral: la emisión de tokens de votación por parte de la autoridad electoral, el registro del voto por el ciudadano y el cierre de la urna con la publicación de los resultados. La red se diseñó considerando una topología distribuida compuesta por tres organizaciones: la Autoridad Electoral, las Terminales de Voto y la Autoridad de Escrutinio, cada una con roles definidos en la validación y el registro de las transacciones.

Los resultados obtenidos muestran que la tecnología blockchain, en particular *Hyperledger Fabric*, ofrece un marco adecuado para garantizar la integridad y trazabilidad de los votos, reduciendo los riesgos de manipulación.

Los principales desafíos se identificaron durante el desarrollo del prototipo y están asociados, una vez superada la curva inicial de aprendizaje de tecnologías blockchain e *Hyperledger Fabric*, a la definición de una red personalizada, a las decisiones de arquitectura adoptadas como el uso de private data collections y a la incorporación dinámica de organizaciones al sistema.

Se analiza la factibilidad de adopción del sistema propuesto, desde el punto de vista esencialmente tecnológico, mediante un esquema de implementación gradual, basado en un plan piloto que permita la coexistencia temporal del voto electrónico con el voto en papel. Este enfoque busca reducir riesgos técnicos y sociales, validar el funcionamiento del sistema en contextos controlados y fortalecer la confianza pública antes de una adopción a mayor escala, si se dieran condiciones y decisiones que exceden lo tecnológico.

Palabras clave: Template, Proyectos de Grado, Computación, Elecciones, Voto Electrónico, Blockchain, Hyperledger, Fabric

Índice general

Capítulo 1

Introducción

El voto electrónico constituye uno de los temas centrales en el debate sobre la modernización de los procesos electorales a nivel mundial. En términos generales, se entiende por voto electrónico a todo sistema en el que el sufragio se emite total o parcialmente por medios digitales, abarcando tanto modelos completamente digitales como híbridos que combinan registro electrónico con respaldo físico en papel [MF25].

1.1. Contexto y Antecedentes

Uruguay cuenta con un sistema electoral consolidado, regulado por la Ley N.º 7.812 [IMP25], que estructura el proceso en torno a papeletas impresas, sobres y urnas físicas. Este marco ha sostenido décadas de elecciones reconocidas por su solidez, pero no contempla ninguna forma de votación electrónica. A nivel operativo, cada elección nacional implica la impresión de cientos de millones de hojas: solo en octubre de 2009 se pusieron a disposición más de 100 millones de papeletas para los 6.868 circuitos del país [Tap09].

A nivel internacional, distintos países han explorado la incorporación de tecnologías digitales en sus procesos electorales, con enfoques que van desde urnas electrónicas dedicadas hasta sistemas de voto por Internet. Algunas experiencias, como la de Estonia o Brasil, han logrado implementaciones a escala nacional, mientras que otras, como Helios Voting o ElectionGuard, se han orientado a contextos institucionales con foco en la auditabilidad y verificabilidad del proceso [MF25]. Estas experiencias son analizadas en detalle en la Subsección 2.1.1. En paralelo, la tecnología blockchain [SAP24] ha emergido como una alternativa de interés para el dominio electoral, gracias a sus propiedades de inmutabilidad, trazabilidad y validación distribuida, abriendo nuevas posibilidades para el diseño de sistemas de votación transparentes y resistentes a la manipulación.

Para que un sistema de votación electrónica sea válido en el contexto uruguayo, debe satisfacer un conjunto de requisitos fundamentales que derivan tanto de consideraciones técnicas como del marco legal electoral vigente. En particular,

el sistema debe garantizar que cada persona habilitada emita exactamente un voto, que no sea posible vincular la identidad del votante con el contenido de su sufragio, que los registros sean inmutables una vez emitidos, y que el proceso sea auditable de forma independiente. Estos requisitos, que se detallan en profundidad en este trabajo, constituyen el marco de referencia sobre el cual se construye y evalúa la propuesta desarrollada en el presente proyecto de grado.

1.2. Motivación y Objetivos

Este trabajo surge de la pregunta: ¿la tecnología blockchain puede constituir una base viable para implementar un sistema de votación electrónica, aplicado al contexto particular de Uruguay? Esta interrogante es relevante tanto desde el punto de vista técnico, dado que blockchain ofrece propiedades atractivas para el dominio electoral, como la inmutabilidad de los registros y la validación distribuida, como desde el punto de vista social e institucional, considerando que Uruguay cuenta con un sistema electoral consolidado, que cualquier alternativa debería poder igualar en garantías.

A partir de esta motivación, el objetivo general del proyecto es evaluar la viabilidad de utilizar una red blockchain como soporte tecnológico para un sistema de urna electrónica en el contexto uruguayo.

Los objetivos específicos de este proyecto son:

1. Realizar un relevamiento del estado del arte en sistemas de votación electrónica, identificando enfoques tecnológicos, niveles de adopción y mecanismos de seguridad utilizados a nivel internacional.
2. Especificar y diseñar un sistema de urna electrónica basado en blockchain que satisfaga los requisitos fundamentales del proceso electoral uruguayo: unicidad del voto, anonimato del votante, inmutabilidad de los registros y auditabilidad.
3. Implementar un prototipo funcional que permita validar en la práctica las decisiones de diseño adoptadas.
4. Discutir la viabilidad de adopción del sistema propuesto en Uruguay, considerando aspectos técnicos, organizacionales y legales.

1.3. Estructura del Informe

El resto del documento se organiza de la siguiente manera:

- El **Capítulo 2** analiza algunos trabajos relacionados y realiza un estado del arte en materia de voto electrónico. Se consideran experiencias internacionales representativas, incluyendo Helios Voting [Har10], ElectionGuard [Mic24b], la Boleta Única Electrónica argentina [Per25a], el sistema brasileño [Tri24] y el i-Voting estonio [Sta24], identificando sus enfoques tecnológicos, mecanismos de seguridad y principales limitaciones.

- El **Capítulo 3** introduce el marco teórico del proyecto. Se definen los conceptos fundamentales de urna electrónica y blockchain, y se describe en detalle la plataforma seleccionada para este proyecto: su arquitectura, componentes principales y los mecanismos de privacidad utilizados en el diseño.
- El **Capítulo 4** presenta la especificación y diseño del sistema propuesto. Se detallan los requisitos funcionales y legales del sistema de votación, el flujo de votación propuesto, el modelo de red y organizaciones, el diseño de los smart contracts [Sza94] [Sza96], y los mecanismos de privacidad y auditoría contemplados.
- El **Capítulo 5** describe la implementación del prototipo funcional. Se explica la arquitectura de software, la implementación de cada funcionalidad (registro de voto, emisión y uso de tokens, cierre de urna), el mecanismo de cifrado asimétrico empleado, y se incluye un análisis de factibilidad que discute limitaciones, comparación con otras soluciones y una propuesta de implementación gradual para Uruguay.
- El **Capítulo 6** expone la experimentación realizada para validar el prototipo. Se describen los experimentos orientados a verificar el control de elegibilidad mediante credenciales de votación y la preservación del anonimato del votante, junto con los resultados obtenidos.
- El **Capítulo 7** expone las conclusiones del trabajo y las líneas de trabajo futuro identificadas, incluyendo la configuración de una red personalizada, la gestión de claves criptográficas, la integración con sistemas institucionales y la incorporación dinámica de organizaciones a la red.

Capítulo 2

Trabajos relacionados

En este capítulo se presentan los trabajos relacionados que sirvieron de base y referencia para el desarrollo del proyecto. El análisis se apoya principalmente en el relevamiento previo realizado sobre el estado del arte del voto electrónico, en el cual se estudiaron diversas experiencias internacionales y enfoques tecnológicos [MF25]. A partir de ese trabajo, se identifican los aportes más relevantes y las líneas de investigación existentes que guardan relación directa con los objetivos del presente proyecto, permitiendo enmarcarlo dentro del panorama actual de soluciones basadas en tecnología blockchain y sistemas de votación electrónica. Información adicional está disponible en el reporte [MF25].

A diferencia de los sistemas relevados, el enfoque propuesto en este proyecto se centra en un despliegue distribuido por circuito de votación, en el cual los votos se almacenan directamente en una red blockchain privada. Este diseño busca aprovechar las propiedades de inmutabilidad y trazabilidad de la tecnología para garantizar la integridad de los resultados, manteniendo al mismo tiempo la autonomía operativa de cada circuito dentro del proceso electoral.

2.1. Estado del arte

2.1.1. Introducción

El voto electrónico constituye uno de los temas debatidos en la modernización de los procesos electorales. En los últimos años, distintos países y organizaciones han explorado su implementación con distintos grados de alcance, desde experiencias locales hasta elecciones nacionales. El objetivo de este capítulo es presentar una síntesis del estado del arte sobre tecnologías aplicadas a la votación electrónica, identificando los enfoques predominantes, sus fundamentos técnicos y los principales desafíos observados a nivel mundial.

A partir del relevamiento, se identificaron distintos enfoques que ilustran el estado actual del voto electrónico a nivel mundial. Entre ellos destacan los siguientes casos.

Helios Voting y Participa UChile

Helios Voting es un sistema de votación en línea de código abierto desarrollado en el MIT, utilizado por instituciones académicas y organizaciones que buscan elecciones seguras y auditables. Basado en el cifrado ElGamal, permite contabilizar votos sin descifrarlos individualmente, preservando así la privacidad del votante.

Su principal aporte radica en la transparencia y verificabilidad del proceso: cada votante obtiene una huella digital única de su voto y puede comprobar posteriormente su registro. No obstante, el propio autor aclara que Helios no está preparado para elecciones gubernamentales, debido a vulnerabilidades relacionadas con la duplicación de votos, la resistencia a la coerción y la complejidad técnica de las auditorías. [Har10]

Una adaptación destacada es *Participa UChile*, utilizada en la Universidad de Chile sobre la base de Helios. [Uni22b] Este sistema refuerza la confidencialidad al dividir la clave de desencriptado entre varios custodios, de manera que ningún voto individual pueda ser revelado. Además, permite auditorías matemáticas abiertas, lo que representa un ejemplo de aplicación universitaria exitosa y verificable, aunque limitada a entornos controlados. [Uni22a]

ElectionGuard

Desarrollado por Microsoft, *ElectionGuard* es un conjunto de herramientas de código abierto orientadas a fortalecer la seguridad y la transparencia electoral. Utiliza cifrado homomórfico, lo que posibilita sumar votos cifrados sin descifrarlos, preservando la privacidad del sufragio. [Mic24b]

A diferencia de otros sistemas, no busca reemplazar las boletas de papel ni habilitar el voto remoto, sino integrarse con los sistemas presenciales existentes para mejorar su auditabilidad. El votante recibe un código de seguimiento que permite verificar que su voto fue incluido correctamente, y los resultados pueden ser auditados por terceros mediante artefactos públicos y especificaciones abiertas [Mic24c]. El sistema ha sido probado en elecciones de varios estados de Estados Unidos y en procesos locales en Europa [Mic24a], siempre como complemento y no como sustituto del voto tradicional.

Boleta Única Electrónica en Argentina

El sistema de Boleta Única Electrónica (BUE) combina el registro digital con un respaldo físico en papel. Se utiliza en la provincia de Salta desde 2009 [Cla25] y en la Ciudad Autónoma de Buenos Aires en varias elecciones [Per25b]. El votante selecciona sus opciones en una máquina, que imprime la boleta con el detalle en texto claro y graba la información en un chip o banda magnética.

El escrutinio se realiza en dos etapas: un conteo electrónico preliminar y un conteo físico definitivo, que prevalece en caso de discrepancias. A pesar de su adopción sostenida, el sistema ha recibido críticas por vulnerabilidades detectadas en auditorías independientes, como la posibilidad de reescritura de chips

RFID y la falta de cifrado robusto. También se ha cuestionado la accesibilidad del sistema para ciertos grupos de votantes. [Per25a]

Sistema de votación en Brasil

Brasil constituye el caso más consolidado de voto electrónico a nivel nacional. Desde 1996 utiliza urnas electrónicas dedicadas, sin conexión a Internet durante la votación, lo que reduce los riesgos de ciberataques. El sistema incluye identificación biométrica, cifrado y hashing de los votos, y mecanismos de redundancia y recuperación ante fallos. [Tri24]

El conteo es descentralizado y los resultados se transmiten por redes seguras a los centros de tabulación. Aunque su código no es abierto, el sistema ha mostrado eficacia y rapidez, consolidándose como una experiencia única de escala nacional.

Voto electrónico en Estados Unidos

En Estados Unidos coexisten múltiples modalidades de votación según cada estado, lo que genera una diversidad de enfoques y niveles de seguridad. El informe de la *National Academies of Sciences* (2018) destaca el concepto de *verificabilidad extremo a extremo* (E2E-V), que busca garantizar que cada votante pueda confirmar el registro y conteo correcto de su voto. [Nat18]

El mismo informe advierte sobre los riesgos de aplicar blockchain y de permitir el voto remoto, señalando que las limitaciones actuales en seguridad informática impiden asegurar un entorno totalmente confiable para este tipo de mecanismos.

Sistema I-Voting en Estonia

Desde 2005, Estonia implementa el *i-Voting*, un sistema de voto por Internet integrado con su infraestructura nacional de identidad digital. Los ciudadanos pueden votar mediante tarjeta con chip, teléfono móvil o aplicación de autenticación, firmando digitalmente su voto. [Sta24]

El sistema permite emitir múltiples votos durante el período habilitado, considerando válido solo el último, e invalida el voto electrónico si luego se realiza uno presencial. Además, los votantes pueden verificar individualmente que su voto fue recibido sin alteraciones. Su adopción ha crecido sostenidamente, superando el 50 % del electorado en 2023. [e-E23]

2.1.2. Síntesis comparativa

Del relevamiento se desprende que la mayoría de las propuestas corresponden a proyectos experimentales o implementaciones acotadas. Se observa una amplia diversidad en cuanto a los mecanismos de identificación, el grado de digitalización y la posibilidad de auditoría. Los sistemas basados en código abierto como Helios y ElectionGuard priorizan la verificabilidad y la transparencia, mientras que otros, como la BUE argentina o el modelo brasileño, enfatizan la eficiencia y la logística operativa.

Aunque la criptografía y la seguridad informática han avanzado considerablemente, existe consenso en que el voto completamente remoto o basado en blockchain aún no es recomendable para elecciones nacionales. Las principales razones son las dificultades para garantizar el anonimato del votante, la resistencia a la coerción y la posibilidad de realizar auditorías independientes. [Nat18] [AM16]

En conjunto, los casos analizados muestran que la evolución del voto electrónico avanza en etapas graduales, combinando progresivamente soluciones digitales con mecanismos físicos de respaldo. Esta tendencia refuerza la importancia de desarrollar tecnologías auditables, adaptadas al contexto social e institucional de cada país, más que replicar modelos tecnológicos universales. [MF25].

Capítulo 3

Marco teórico

Entendemos por voto electrónico a aquellos sistemas de votación en los que el acto de sufragar se realiza mediante un sistema electrónico y el voto es registrado y persistido a través de dicho sistema. En esta definición, el medio electrónico no cumple únicamente una función auxiliar o de apoyo administrativo, sino que interviene directamente en la emisión y almacenamiento del sufragio. Se incluyen tanto los modelos en los que el registro es exclusivamente digital como aquellos que generan una constancia física complementaria, siempre que la expresión inicial del voto y su registro primario se realicen por medios electrónicos.

En el análisis se adopta una definición amplia de voto electrónico que incluye tanto los sistemas completamente digitales como los modelos híbridos que combinan el registro electrónico y el respaldo físico. Bajo esta perspectiva, el voto puede emitirse en una urna electrónica o de forma remota, siempre que se cumplan los principios fundamentales de verificabilidad, anonimato y control del proceso.

Típicamente, los procesos electorales siguen un flujo convencional, el cual se describe en el diagrama de secuencia de la Figura 3.1.

En este trabajo, el alcance se focaliza en las siguientes etapas del proceso:

- Habilitar un votante: Se asume que la verificación de identidad es un problema resuelto y se lo delega a la corte electoral (mesa).
- Entregar el material de votación.
- Depositar el voto.
- Cerrar la urna.
- Realizar el escrutinio.

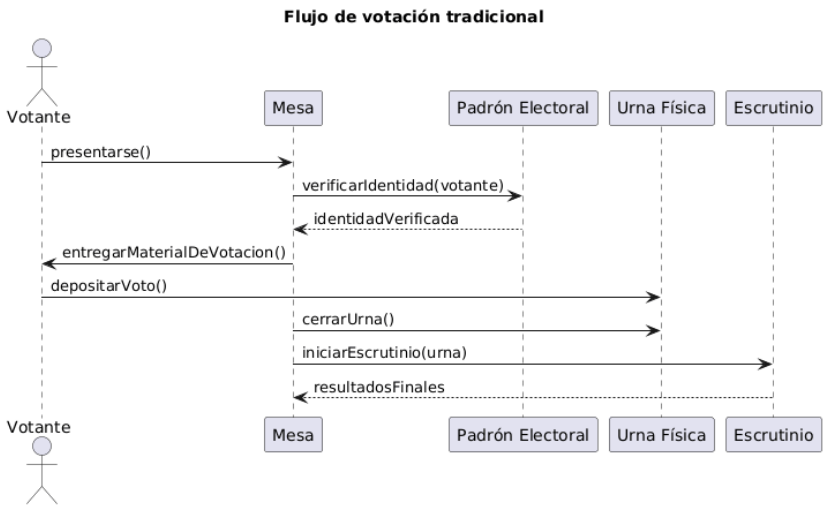

Figura 3.1: Flujo de votación tradicional

3.1. Definiciones

3.1.1. Urna electrónica

Definición

Se considera una **urna electrónica** como un sistema utilizado en procesos electorales que permite a los votantes ingresar su sufragio, cumpliendo las siguientes restricciones:

- Permite **exactamente un voto por persona**.
- Los votos ingresados son **inmutables** y no pueden ser modificados bajo ninguna circunstancia.
- El sistema **preserva el anonimato** del votante, sin posibilidad de vincular su identidad con el contenido del voto.
- Los votos son **almacenados de forma segura** para proteger su integridad. Solo se debe tener acceso al voto en el momento del conteo y por la autoridad encargada del escrutinio.
- Permite la realización de **auditorías independientes** que garanticen la exactitud y confiabilidad del escrutinio.

3.1.2. Blockchain

En términos generales, un blockchain es un ledger de transacciones inmutable, mantenido dentro de una red distribuida de nodos pares. Cada uno de estos

nodos mantiene una copia del ledger aplicando transacciones que han sido validadas por un protocolo de consenso, agrupadas en bloques que incluyen un hash que vincula cada bloque con el bloque anterior. La primera y más reconocida aplicación del blockchain es la criptomoneda Bitcoin [Nak08], aunque otras han seguido sus pasos. Ethereum [Eth26a], una criptomoneda alternativa, adoptó un enfoque diferente, integrando muchas de las mismas características que Bitcoin pero añadiendo *smart contracts* para crear una plataforma de aplicaciones distribuidas.

Conceptos importantes

Smart contracts: Son programas almacenados en la blockchain que operan bajo una lógica condicional: ante un determinado evento o condición, ejecutan automáticamente una acción predefinida. Su comportamiento está enteramente gobernado por el código, que no puede modificarse una vez creado.

Nick Szabo acuñó el término *contrato inteligente*. En 1994 [Sza94], escribió una introducción al concepto, y en 1996 [Sza96] escribió una exploración sobre lo que los contratos inteligentes podrían hacer.

Szabo imaginó un mercado digital donde procesos automáticos y criptográficamente seguros permiten la realización de transacciones y funciones comerciales sin intermediarios de confianza. [Eth26b]

Protocolo de consenso: Un mecanismo de consenso blockchain es un sistema automatizado que persigue dos objetivos principales: garantizar que una comunidad distribuida de validadores pueda acordar de forma eficiente y unánime los datos del ledger, y asegurar que todos los validadores sigan las reglas del protocolo de manera honesta.

Esto es especialmente crítico en sistemas descentralizados, donde no existe una autoridad central que certifique la validez de las transacciones. Sin un mecanismo de este tipo, información incorrecta, como un saldo falso o una transacción duplicada, podría comprometer la integridad de toda la base de datos, erosionando la confianza en el sistema.

Además de la validación, los mecanismos de consenso cumplen un rol clave en la seguridad de la red, actuando como barrera contra ataques de mayoría, es decir, intentos de tomar control de más del 50 % de la red.

Los principales mecanismos de consenso son:

- **Proof-of-work (PoW)** es un mecanismo de consenso que requiere que los validadores, destinen poder computacional para resolver competencias criptográficas para proponer nuevos bloques. La inversión en equipos y energía actúa como barrera de entrada para potenciales atacantes, ya que controlar la red implicaría acumular una cantidad desproporcionada de poder de cómputo.

- **Proof-of-stake (PoS)** es un mecanismo de consenso donde los validadores deben adquirir y conservar una cantidad de assets de la red (proceso

conocido como staking) para poder participar en la validación de bloques. Los validadores son seleccionados aleatoriamente, con mayores probabilidades para quienes más assests tengan. El comportamiento deshonesto puede ser penalizado mediante la confiscación parcial o total de los assests conservados, mecanismo conocido como "slashing". [Kra25]

Tipos de redes de blockchain

Red pública: Las redes blockchain públicas, como Bitcoin y Ethereum, son abiertas a cualquier persona: cualquiera puede leer el ledger, enviar transacciones y participar en el proceso de consenso sin necesidad de permisos, manteniendo el anonimato.

Red híbrida o semiprivada: Son operadas por una única organización que otorga acceso a los usuarios que satisfacen criterios preestablecidos. Aunque no están verdaderamente descentralizadas, este tipo de redes blockchain permisionadas restringen la participación a aquellos que cumplen con las condiciones definidas por el operador.

Red privada: También están controlados por una sola organización. Ella determina quién puede leerlo, presentar transacciones en él, y participar en el proceso de consenso.

Red de consorcio: En una red blockchain de consorcio, el proceso de consenso es controlado por un grupo preseleccionado de participantes en lugar de una única organización. El acceso para leer el ledger y enviar transacciones puede ser público o estar restringido según las reglas definidas por el grupo. Al distribuir el control entre múltiples actores, este modelo combina elementos de descentralización con un entorno de permisos controlado. Por estas características, el blockchain de consorcio es el modelo más adoptado en entornos empresariales e institucionales, donde múltiples organizaciones necesitan colaborar y compartir información sin ceder el control a una única entidad. [SAP24]

3.1.3. Hyperledger Fabric

El proyecto Hyperledger [Hyp25h] fue creado por The Linux Foundation en 2015 con el objetivo de promover tecnologías blockchain para distintos sectores de la industria. En lugar de establecer un único estándar, Hyperledger propone un enfoque colaborativo y abierto al desarrollo de tecnologías distribuidas.

Hyperledger Fabric es una de las plataformas blockchain desarrolladas bajo este proyecto. Como otros sistemas blockchain, incluye un libro mayor (ledger), contratos inteligentes (smart contracts o chaincode) y permite la gestión de transacciones entre participantes. Sin embargo, se diferencia de blockchains públicas en que es una red privada y con permisos, lo que significa que solo actores autorizados pueden participar.

Privacidad

A diferencia de redes públicas, Fabric utiliza un sistema de identidades verificadas mediante un Proveedor de Servicios de Membresía (MSP) [Hyp25f]. Esto elimina la necesidad de algoritmos como proof of work para validar transacciones [Hyp25b]. Además, su diseño modular permite configurar distintos componentes: Mecanismos de consenso intercambiables. Distintos formatos para almacenar datos del libro mayor. Soporte para múltiples MSPs según las necesidades del sistema. Una característica clave es la posibilidad de crear canales privados [Hyp25a], donde solo ciertos participantes tienen acceso a un ledger (libro mayor) particular [Hyp25d]. Esto resulta útil en entornos donde existen relaciones confidenciales. *Hyperledger Fabric* ofrece niveles flexibles de privacidad. Gracias al uso de canales y control de acceso, permite adaptarse tanto a redes abiertas como a entornos B2B en los que los participantes desean compartir solo cierta información con determinados actores. Esta capacidad lo hace ideal para casos donde la confidencialidad es esencial.

Ledger Compartido

El ledger de *Hyperledger Fabric* se divide en dos partes:

- **World state:** representa el estado actual de los activos, almacenado en una base de datos como LevelDB o CouchDB. [Hyp25e]
- **Blockchain:** registra todas las transacciones que llevaron al estado actual. Cada participante tiene una copia sincronizada del libro mayor para cada red de la que forma parte. Esto garantiza transparencia, trazabilidad y coherencia del sistema.

Smart Contracts

Los contratos inteligentes en Fabric se denominan chaincode. Son programas que gestionan la lógica de negocio y permiten interactuar con el ledger, principalmente con el world state (por ejemplo, para consultar o modificar datos). Los chaincode pueden escribirse en lenguajes como Java, Go o Node.js, y son invocados por aplicaciones externas.

Consenso

Para mantener la coherencia del libro mayor, las transacciones deben ordenarse correctamente y validarse antes de ser registradas. *Hyperledger Fabric* permite elegir entre distintos mecanismos de consenso según las necesidades de la red. A diferencia de blockchains públicas como Bitcoin, que utilizan Proof-of-Work y requieren una alta capacidad de cómputo, Fabric evita la minería y adopta algoritmos más eficientes y configurables, lo que se traduce en un menor consumo de recursos y una validación más rápida de las transacciones.

Por ejemplo, el mecanismo de consenso **Raft** [OO14] (mecanismo utilizado para implementar el prototipo) es un algoritmo de tolerancia a fallas por caída

que garantiza que todos los nodos de las organizaciones Mesa, Urna y Escrutinio mantengan un estado idéntico del libro mayor. Este esquema asegura propiedades fundamentales como la integridad del voto, la prevención del doble voto y la consistencia en el recuento. Raft ofrece buen rendimiento y simplicidad de despliegue, por lo que resulta adecuado para entornos de prueba y validación del prototipo.

Sin embargo, para un escenario real de elecciones, sería recomendable utilizar un mecanismo más robusto, como SmartBFT, que introduce tolerancia a fallas bizantinas y protege al sistema frente a comportamientos maliciosos o intentos de manipulación del orden de las transacciones. En el contexto de Hyperledger Fabric, una falla bizantina se refiere a situaciones en las que uno o más nodos pueden comportarse de manera arbitraria o maliciosa por ejemplo, enviando mensajes contradictorios, alterando el orden de las transacciones o intentando introducir información inválida sin que exista una falla evidente del sistema. Los algoritmos con tolerancia a fallas bizantinas permiten que la red continúe operando correctamente siempre que la cantidad de nodos deshonestos no supere un umbral determinado, garantizando así la consistencia y el acuerdo sobre el estado del ledger. Aunque esta alternativa implica una mayor complejidad y un leve aumento en la latencia, ofrece mayores garantías de seguridad e integridad del proceso electoral, reforzando la confiabilidad del sistema en contextos institucionales.

3.2. Arquitectura típica de una aplicación desarrollada en fabric

La arquitectura de una aplicación desarrollada en *Hyperledger Fabric* se basa en una red distribuida y permisionada que permite la colaboración entre múltiples organizaciones, manteniendo la privacidad, integridad y trazabilidad de las transacciones.

Fabric distingue entre los roles de los participantes, los canales de comunicación y los mecanismos de control de acceso, ofreciendo una estructura flexible y modular adaptable a distintos contextos de uso.

- **Clients:** Son las aplicaciones externas que inician transacciones y consultas. Interactúan con la red mediante un SDK (por ejemplo, en Node.js o Java) y utilizan identidades digitales emitidas por una autoridad de certificación (CA).
- **Peers:** Son nodos que mantienen una copia del libro mayor (ledger) y ejecutan contratos inteligentes (chaincode). Un peer puede asumir el rol de *endorser*, al simular y firmar transacciones, y/o de *committer*, al validar y aplicar bloques al ledger.
- **Orderers:** Componen el *Ordering Service*, encargado de recibir las transacciones endosadas, ordenarlas cronológicamente y agruparlas en bloques que luego se distribuyen a los peers del canal correspondiente.

- **Chaincode:** Representa los contratos inteligentes que definen la lógica de negocio de la aplicación. Cada chaincode se instala y aprueba por cada organización en un canal, y se ejecuta dentro de contenedores aislados.
- **Ledger:** Cada peer mantiene un ledger compuesto por dos partes: (1) la *blockchain*, un registro inmutable de transacciones, y (2) la *base de datos de estado* (como LevelDB o CouchDB), que refleja el estado actual del sistema.
- **Channels:** Son redes lógicas privadas dentro de la red Fabric, que agrupan un subconjunto de organizaciones. Cada canal mantiene su propio ledger y puede tener contratos inteligentes particulares, lo que permite el aislamiento de datos y transacciones.
- **Organizations:** Cada entidad participante posee su propio *Membership Service Provider* (MSP), que gestiona las identidades válidas dentro de la red. Los miembros pueden poseer peers, participar en canales, aprobar chaincodes y definir políticas de acceso.
- **Certificate Authorities (CAs):** Son los componentes responsables de emitir certificados digitales (X.509) a usuarios y nodos. Estos certificados autentican a los participantes y son fundamentales para la firma y validación de transacciones.
- **Private Data Collection (PDC):** Son un mecanismo que permite a un subconjunto de organizaciones en un canal compartir y mantener información confidencial de forma privada [Hyp25c] . En lugar de almacenar los datos directamente en el *ledger* compartido, la colección guarda únicamente un *hash* del contenido en el libro mayor, garantizando la inmutabilidad y la integridad de la información. Los datos reales se distribuyen de manera punto a punto (*peer-to-peer*) únicamente entre los nodos autorizados a la colección. De esta forma, las PDCs permiten cumplir con requisitos de privacidad y confidencialidad sin sacrificar las propiedades de consenso y trazabilidad que ofrece la red blockchain.

El flujo de una transacción en *Hyperledger Fabric* sigue las siguientes etapas principales:

1. La aplicación cliente construye y firma una propuesta de transacción, que es enviada a un conjunto de *endorsing peers*.
2. Cada peer simula la ejecución del chaincode sin modificar el ledger, y responde con una propuesta firmada (endorsement).
3. El cliente reúne las respuestas válidas y envía la transacción al *ordering service*.
4. El ordenamiento agrupa las transacciones en bloques y los distribuye a los peers del canal correspondiente.

5. Los peers validan las transacciones de acuerdo con las políticas definidas y actualizan su ledger.

La figura 3.2 ilustra el flujo típico de ejecución de una transacción en una red Hyperledger Fabric. En primer lugar, la aplicación cliente (A) se conecta a un peer (P1) y envía una propuesta de invocación de chaincode. El peer ejecuta el contrato inteligente (S1) en modo simulación, generando una propuesta de respuesta que incluye el conjunto de lecturas y escrituras, sin modificar aún el ledger (L1). Una vez obtenidas las respuestas requeridas conforme a la política de endoso definida, la aplicación solicita al servicio de ordenamiento (O1) que procese la transacción. El ordering service agrupa las transacciones en bloques y los distribuye a los peers del canal, quienes validan las firmas y políticas correspondientes antes de actualizar finalmente el ledger. Como resultado, se emite un evento de actualización que puede ser recibido por la aplicación cliente, completando así el ciclo de la transacción.

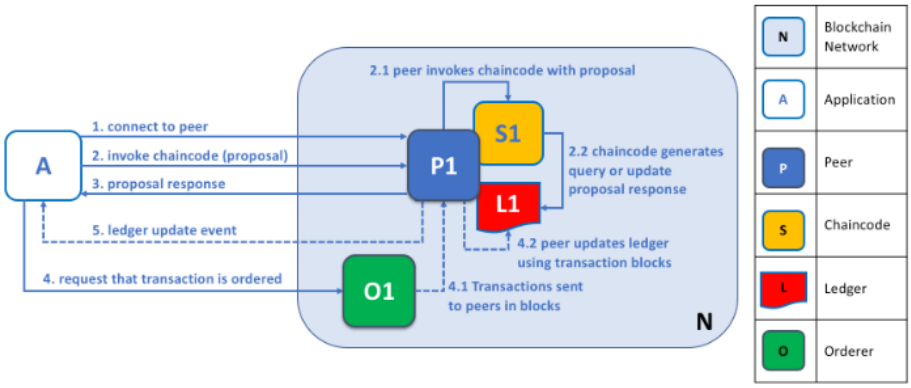

Figura 3.2: Componentes dentro del sistema

Este diseño modular y distribuido de Fabric permite construir aplicaciones seguras, escalables y auditables, adecuadas para entornos donde participan múltiples organizaciones con intereses diversos [Hyp17a], [Hyp17b].

Capítulo 4

Especificación y diseño de un sistema de urna electrónica

4.1. Introducción

En este capítulo se presenta el diseño de un sistema de urna electrónica basado en tecnología blockchain. Durante la etapa de investigación del proyecto, nuestro tutor, Carlos Luna, nos puso en contacto con un estudiante de maestría que también estaba siendo tutorado por él, quien desarrollaba su tesis de posgrado sobre un tema relacionado con blockchain. Este estudiante nos recomendó Hyperledger Fabric, plataforma que él mismo utilizaba en su proyecto. Luego de revisar la documentación de la plataforma y realizar algunas pruebas de concepto, concluimos que se trataba de una tecnología que se adecuaba bien a los objetivos del proyecto por varias razones: permite definir organizaciones con roles y permisos diferenciados, ofrece mecanismos de privacidad que posibilitan restringir el acceso a información sensible únicamente a los participantes autorizados, y evita la necesidad de algoritmos de consenso costosos computacionalmente como los empleados en redes públicas. Estas características la hacen apropiada para modelar las distintas entidades que intervienen en un proceso electoral y las relaciones de confianza entre ellas.

A lo largo del capítulo se definen los conceptos fundamentales del sistema, tales como qué entendemos por voto y por urna, los distintos componentes, los mecanismos de acceso a los datos, los actores involucrados y los roles que desempeña cada uno. Es importante destacar que el término *token* se utiliza de forma genérica, ya que puede representar tanto un dispositivo físico como una credencial lógica, quedando su implementación final abierta a interpretación. Como posible mecanismo concreto, *PKCS#11* [AGE25] podría ser una opción adecuada, dado que es el estándar utilizado por la cédula de identidad electrónica

uruguaya.

4.2. Requisitos del Sistema de Votación

El sistema de urna electrónica propuesto debe cumplir con una serie de requisitos fundamentales para garantizar su validez, confiabilidad y adaptabilidad al proceso electoral. Estos requisitos son:

- **Recepción de votos válidos:** El sistema debe permitir registrar únicamente votos provenientes de personas habilitadas para sufragar, asegurando que cada votante pueda emitir un único voto. Este requisito se corresponde con lo dispuesto por la Ley N.º 7.812 [IMP25], que establece en su artículo 1º que son electores las personas inscritas en el Registro Cívico Nacional habilitadas para votar, y en el artículo 82 que la Comisión Receptora debe verificar mediante la lista ordinal, que el elector no haya sufragado previamente. En este contexto, el sistema debe incorporar mecanismos que reflejen este control, garantizando la unicidad y validez de cada voto emitido.

- **Anonimato del votante:** No debe ser posible vincular un voto con la identidad del votante. El sistema debe preservar la privacidad del elector sin comprometer la validez del proceso. Este principio se alinea con el carácter secreto del sufragio establecido por la normativa electoral. Los artículos 47 y 84 de la Ley N.º 7.812 [IMP25] establecen el cuarto secreto como mecanismo para garantizar que el elector pueda çolocar en el sobre la hoja de votación"sin ser visto, separando explícitamente la identificación del elector (artículo 82) del acto de emisión del voto. Cualquier sistema electrónico que se proponga como alternativa debe respetar esta separación funcional, asegurando que la identidad del votante no pueda ser inferida a partir del voto registrado.

- **Auditoría del proceso:** Debe ser posible verificar, de forma transparente, la cantidad de votos emitidos durante el proceso electoral, sin acceder al contenido de los votos ni a información sensible. La legislación electoral prevé múltiples mecanismos de fiscalización y control del proceso. El artículo 42 [IMP25], literal C, establece que las Comisiones Receptoras deben efectuar escrutinios primarios, mientras que el artículo 166 reconoce el derecho de las agrupaciones políticas a designar delegados para "presenciar y fiscalizar todos los actos referentes a la votación y los escrutinios". El artículo 97 establece la obligación de extender actas de clausura con el número de electores que han sufragado. En este sentido, el sistema debe permitir auditorías que reflejen estas prácticas, habilitando el control del proceso sin comprometer el secreto del sufragio.

- **Inmutabilidad de los registros:** Una vez emitido, cada voto debe almacenarse de forma inalterable. El sistema debe garantizar que no existan modificaciones ni eliminaciones posteriores. Este requisito está alineado

con la normativa vigente, que establece rigurosos controles para prevenir la adulteración. El artículo 191 [IMP25], numeral 9º, tipifica como delito electoral "la adulteración, modificación o sustracción, falsificación de las actas y documentos electorales", con penas establecidas en el artículo 192. Adicionalmente, el artículo 115 dispone que las urnas deben ser çerradas, lacradas y selladas"para garantizar la integridad de su contenido. La inmutabilidad técnica de los registros constituye, por lo tanto, un requisito esencial para preservar la integridad del proceso electoral.

- **Integridad y trazabilidad:** Toda transacción registrada en el sistema debe ser verificable y estar respaldada por mecanismos que aseguren su autenticidad. La legislación electoral contempla distintos instrumentos de control y verificación. El artículo 83 [IMP25] establece la obligación de llevar una "lista ordinal"que registre el número de orden del votante y los datos de su inscripción. El artículo 111 requiere que las Comisiones Receptoras levanten actas detalladas del escrutinio, consignando el número total de sobres y votos de cada clasificación. El artículo 143 dispone que todos los documentos relacionados con la elección queden bajo custodia de la Junta Electoral hasta la resolución sobre la validez del proceso. El sistema propuesto debe reflejar estos principios de control documental, proporcionando trazabilidad de las operaciones realizadas sin exponer información sensible ni vulnerar el anonimato del votante.

4.3. Flujo de Votación Propuesto

El votante llega al circuito y realiza el control de identidad en la mesa de votación, llevado a cabo por los integrantes de la mesa de la misma forma en que se realiza actualmente. Una vez validada la identidad y habilitado para votar, se le entrega un token mediante algún formato físico. Este token es un código que habilita a la persona a votar frente a la urna, con un funcionamiento muy similar al de los sobres físicos, y sus identificadores, que se usan actualmente. Con dicho token, el votante se dirige a la urna electrónica. Allí, verá desplegadas las opciones disponibles y seleccionará la que prefiera. Una vez finalizada la selección, se le presentará un resumen de los votos (puede que se vote más de una cosa) y se le pedirá que confirme la elección. Al confirmar, el voto será encriptado e ingresado al sistema. Finalmente, el votante vuelve a la mesa de votación, devuelve el token, la mesa comprueba que haya sido utilizado, le entrega el comprobante de votación y el votante se retira.

4.4. Diseño Propuesto con Hyperledger Fabric

4.4.1. Diagrama de sistema y actores

El sistema propuesto se despliega en cada circuito de votación. En la actualidad, cada centro de votación contiene varios circuitos, y cada circuito está compuesto por una mesa y una urna. La mesa es el punto donde se autentica al votante y se emite el token que habilita el voto. La urna, por su parte, es el dispositivo donde se registra efectivamente el sufragio.

Al finalizar la votación, cuando las urnas son cerradas, los miembros de la mesa agregan al sistema a la autoridad de escrutinio. Esta figura es la única con capacidad de revelar los votos y realizar el conteo correspondiente en una urna determinada. Una vez concluido el conteo en cada mesa, los resultados son transmitidos a la Corte Electoral, manteniendo el mismo procedimiento ya existente hoy en día: los integrantes de la mesa utilizan un dispositivo electrónico para remitir el acta de escrutinio al sistema central de la Corte Electoral.

Actores del sistema

- **Votantes:** Se autentican en la mesa, reciben un token que habilita su voto, emiten el sufragio en la urna, obtienen un certificado de voto y finalmente se retiran del centro.
- **Miembros de la mesa:** Son responsables de validar la identidad de los votantes, emitir el token que habilita el voto y verificar posteriormente que dicho token haya sido utilizado, para luego entregar el certificado de voto. Además, al cierre de la votación, incorporan a la autoridad de escrutinio al sistema.
- **Autoridad de escrutinio:** Es incorporada al sistema por los miembros de la mesa al momento del cierre de la urna. Posee la facultad exclusiva de revelar los votos y realizar el conteo de una urna específica.

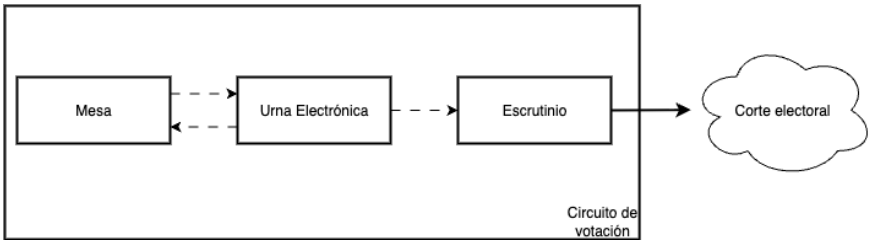

Figura 4.1: Componentes/Organizaciones dentro del sistema

En la figura 4.1 se puede ver como se despliegan los diferentes actores y como interactúan entre ellos.

- **Mesa → Urna electrónica:** La mesa emite un token de habilitación para cada votante, que es enviado a la urna electrónica.
- **Urna electrónica → Mesa:** Una vez utilizado el token en la urna, se genera la confirmación de que el votante ya ejerció su derecho, lo que habilita a la mesa a entregar el certificado de voto.
- **Urna electrónica → Autoridad de escrutinio:** Al cierre de la votación, la urna entrega los votos cifrados a la autoridad de escrutinio, quien tiene la capacidad de revelar y contabilizar los resultados.
- **Autoridad de escrutinio → Urna electrónica:** La autoridad accede a los datos de la urna para realizar el conteo, manteniendo el vínculo hasta finalizar el proceso de escrutinio.
- **Autoridad de escrutinio → Corte Electoral:** Una vez concluido el conteo en cada mesa, la autoridad de escrutinio transmite los resultados a la Corte Electoral.
- **Corte Electoral (Fuera del sistema):** Recibe los conteos parciales provenientes de las diferentes mesas y los integra al sistema central de resultados, igual que ocurre en el procedimiento actual.

4.4.2. Modelo de red y organizaciones

La red contendría las siguientes organizaciones:

- **Mesa de votación:** Valida la identidad del votante previo a emitir un token. Una vez validado que la persona esta habilitada a votar en ese circuito emite un token firmado por ella.
- **Urna electrónica:** Presenta las opciones al votante y luego de la selección, se usa el token firmado por la mesa para emitir un voto valido. Llegada la hora, se cierra con una firma particular para no permitir el ingreso de mas votos.
- **Autoridad de escrutinio:** Con la urna cerrada, puede consultar el conteo del total de votos por opción de voto.
- **Auditoria de corte electoral:** Esta organización debe ser capaz de auditar el registro de tokens usados contra los votos registrados. Es decir, debe chequear la relación entre tokens utilizados y votos emitidos.

4.4.3. Diseño de los chaincodes

A continuación, se detallan los *chaincodes* mínimos que consideramos necesarios para garantizar el correcto funcionamiento de la urna electrónica. Cada uno de ellos implementa una parte de la lógica del sistema, asegurando la emisión, validación, registro y conteo de los votos de forma segura.

1. **Registrar voto** - Urna electrónica
   Permite que un votante emita su voto. Valida que tenga un token habilitado y que no haya votado previamente. Registra el voto de forma inmutable en el *ledger* y es invocado por la urna electrónica. El token utilizado debe haber sido emitido por la mesa y no debe haber sido previamente utilizado.

2. **Emitir token para votante** - Mesa de votación
   Emite un token de un solo uso para un votante en particular, el cual debe ser utilizado para emitir el voto. El token se guarda en una colección privada pero visible para la urna y, una vez utilizado, se marca como revocado.

3. **Chequeo estado token votante** - Mesa votación
   Chequea si el token emitido fue efectivamente utilizado o esta aun disponible para su uso. Cuando el token fue utilizado, la mesa puede afirmar con certeza que un votante emitió su voto.

4. **Cierre de urna** - Mesa de votación
   Una vez llegada la hora, la mesa de votación puede invocar el cierre de la urna, que consiste en bloquear el ingreso de nuevos votos e incorporar a la organización de escrutinio al canal.

5. **Conteo de votos** - Autoridad de escrutinio
   La autoridad de escrutinio, una vez incorporada al canal, puede invocar el chaincode que devuelve, para cada candidato, alternativa u opción de la elección, cuántos votos obtuvo cada uno.

6. **Obtener opciones de voto** - Urna electrónica
   Carga el conjunto de opciones disponibles en la urna correspondiente. Debe obtener y cargar las opciones posibles de la elección correspondiente para que el votante realice su selección.

7. **Publicar resultados** - Autoridad de escrutinio
   Extrae los resultados del *ledger* y los entrega a una aplicación externa. Garantiza que los resultados sean verificables e idénticos para todos los observadores.

4.4.4. Privacidad y anonimato del votante

El sistema se puede diseñar para preservar la privacidad del votante en todo momento. Para ello, no se almacena ninguna información identificatoria junto con el voto emitido. En su lugar, se registra únicamente un *hash* del voto en el *ledger*, el cual garantiza la integridad de la información sin revelar su contenido exacto ni su origen.

4.4.5. Mecanismos de seguridad y auditoría

Uno de los aspectos a considerar del sistema es la posibilidad de auditar la cantidad de votos registrados de forma confiable y transparente. Para ello, se

disponibilizará un *chain code* que permite consultar en tiempo real la cantidad de votos ingresados en la urna

Este mecanismo puede ser utilizado por las autoridades de mesa o auditores, quienes cuentan con información independiente sobre los tokens que emitieron previamente para habilitar el acceso de los votantes a la urna. De este modo, es posible verificar que la cantidad de votos registrados coincida con la cantidad de accesos autorizados, sin comprometer la identidad ni el contenido del voto.

Esta auditoría cruzada entre los tokens emitidos y los votos registrados refuerza la integridad del sistema y permite detectar posibles anomalías de manera temprana.

Capítulo 5

Implementación de un prototipo

5.1. Prototipo

Con el objetivo de validar la viabilidad técnica del enfoque planteado en los capítulos anteriores, se desarrolló un prototipo funcional del sistema de urna electrónica basado en *Hyperledger Fabric*. Este prototipo permite comprobar de forma práctica el comportamiento de los principales componentes del sistema: la Mesa de Votación, la Urna Electrónica y la Autoridad de Escrutinio, así como evaluar el cumplimiento de los requisitos de integridad y anonimato definidos en la etapa de diseño. Además, se incorpora la posibilidad de que cada votante pueda verificar posteriormente que su voto fue registrado correctamente a través de una plataforma de consulta, aspecto que se busca evaluar dentro del alcance del prototipo.

El desarrollo se orientó a reproducir, en un entorno controlado, las interacciones fundamentales del proceso electoral: la emisión de tokens de habilitación, el registro cifrado de los votos y el cierre de la urna para habilitar el conteo. De esta forma, se buscó verificar que la tecnología blockchain pueda garantizar la inmutabilidad y coherencia de los registros sin comprometer la confidencialidad de la información sensible.

Si bien el prototipo no persigue fines productivos ni busca cubrir la totalidad del proceso electoral, constituye una prueba de concepto que permite identificar los desafíos técnicos y organizacionales asociados a la adopción de un sistema de voto electrónico distribuido. El código completo del prototipo, documentado, está disponible en el Apéndice A

5.1.1. Descripción general

El prototipo será una red que cuente con tres organizaciones: Mesa de votación, Urna electrónica y Autoridad de escrutinio. Se creará un único canal que

unirá a la Mesa con la Urna, y una vez cerrada esta última, también se unirá la Autoridad de escrutinio al mismo.

Para poder implementar la emisión de tokens, la validación correspondiente a la hora de ingresar un voto, y el manejo de los votos mismos, decidimos utilizar PDCs. Se tendrá una colección de tokens emitidos por la mesa, a la cual pueden acceder tanto la mesa como la urna, consultando datos distintos (la mesa para poder darlos de alta, la urna para validar que el token sea uno válido). Por otro lado, estará la colección de votos, accesible por la urna. En ella se guardarán los datos de los votos cifrados.

5.1.2. Esquema de componentes

Las Figuras 5.1 y 5.2 presentan la arquitectura de software del sistema propuesto. En primer lugar, se muestra una vista a nivel de contenedores, que permite identificar los principales componentes del sistema y sus responsabilidades, así como las interacciones entre el application gateway y la red *Hyperledger Fabric*. A continuación, se detalla una vista a nivel de componentes del application gateway, donde se explicitan las organizaciones participantes y las operaciones que cada una ejecuta sobre los chaincodes.

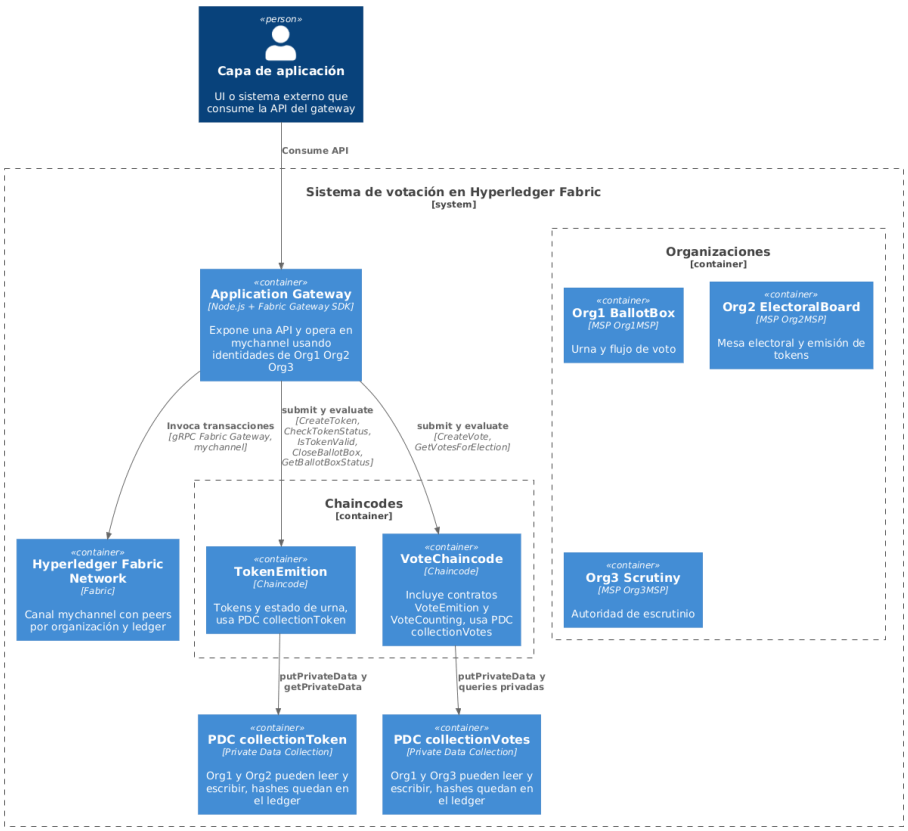

Figura 5.1: Arquitectura de software del sistema a nivel de contenedores.

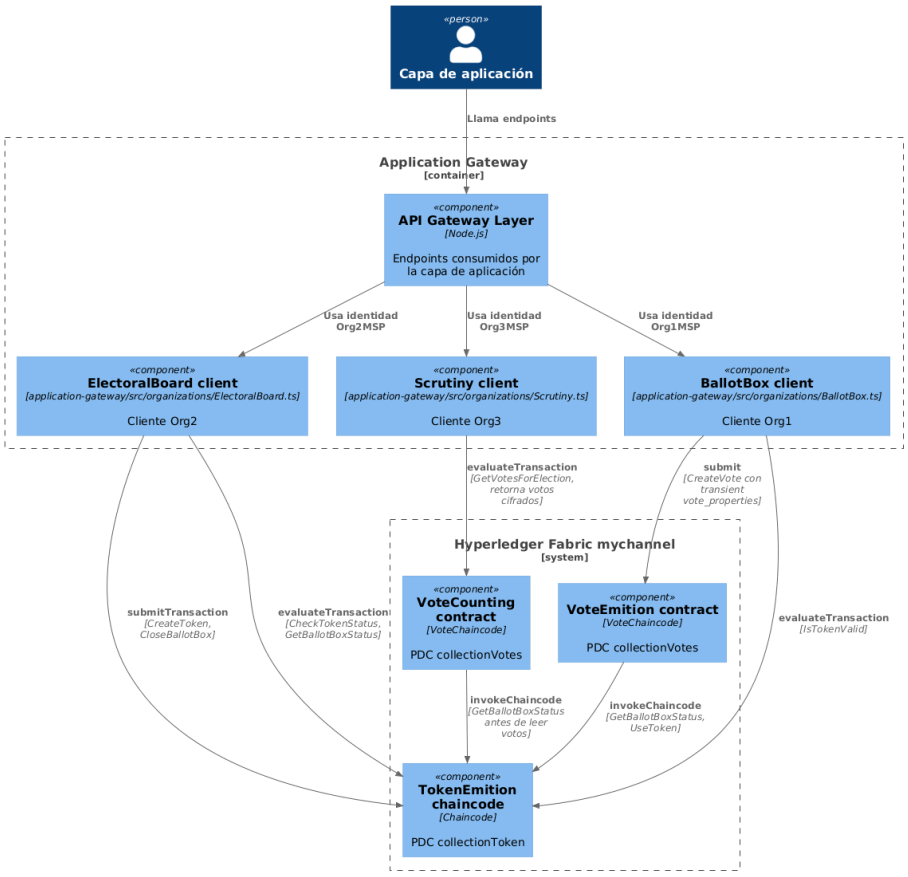

Figura 5.2: Vista a nivel de componentes del application gateway (modelo C4).

5.1.3. Implementación de chaincodes

De los chaincodes mencionados previamente en la sección 4.4.3, se implementaron los siguientes:

- **VoteCounting**
  - GetVotesForElection
- **VoteEmition**
  - CreateVote
- **TokenEmition**
  - CreateToken
  - CheckTokenStatus
  - IsTokenValid
  - UseToken
  - CloseBallotBox
  - GetBallotBoxStatus

Las operaciones CloseBallotBox y GetBallotBoxStatus fueron incorporadas dentro del chaincode TokenEmition, aunque conceptualmente podrían desacoplarse en un chaincode independiente orientado a la gestión del estado de la urna.

5.1.4. Registro de Voto

Se recibe la elección del votante y se registra el voto en la PDC correspondiente, encriptando la elección del votante.

Tanto la creación del voto como la actualización del estado del token utilizado se ejecutan dentro de una **misma transacción blockchain**, garantizando así la **atomicidad del proceso**. Esto implica que ambas operaciones, el registro del voto y el marcado del token como usado, se completan de forma indivisible: o bien ambas se aplican correctamente al *ledger*, o bien ninguna de ellas se registra en caso de error.

Este comportamiento atómico, como se ve en el fragmento de código 5.1, evita inconsistencias en el sistema, tales como votos almacenados sin un token asociado o tokens marcados como utilizados sin que se haya registrado el voto correspondiente. De esta forma se asegura la coherencia entre los estados de las colecciones privadas de **Votos** y **Tokens**, preservando la integridad del flujo de votación.

Fragmento de código 5.1: Creación del voto y uso del token - VoteEmition.ts

```
@Transaction()
public async CreateVote(ctx: Context): Promise<void> {
  //Se valida la organizacion que invoca el chaincode, y
      que el estado de la urna sea OPEN
  //...

  //Se crea la entidad del voto, con la elección del
      votante
  const vote: Vote = {
    election: voteAsset.election,
    candidate: voteAsset.candidate,
    createdAt: new Date()
  }

  // Guarda el voto en la PDC de votos.
  await ctx.stub.putPrivateData(assetCollection, voteAsset
      .sign, this.voteToUint8Array(vote));

  // Se invoca el uso del token, dentro de la transacción,
       de forma que el ingreso del voto y el uso del token
      se de de manera atómica
  await this.useToken(ctx, voteAsset.tokenId);
}
```

Encriptado del voto

Se protege un campo específico del objeto de voto de modo que sólo el destinatario autorizado pueda leerlo. El resto del objeto permanece legible para facilitar el procesamiento en la aplicación y en el chaincode (el resto de los atributos son campos genéricos como por ejemplo la elección, bien podría guardarse solo el candidato elegido). El mecanismo de cifrado utilizado es **asimétrico**, usa un par de claves pública y privada, y **no determinista**, ya que incorpora elementos aleatorios que hacen que el mismo contenido genere resultados distintos cada vez que se cifra.

Idea general

El sistema combina dos etapas

- Cifrado del contenido con una clave efímera de uso único
- Protección de esa clave efímera con la clave pública del destinatario

De esta manera el dato queda oculto y sólo el destinatario que posee la clave privada correspondiente puede recuperarlo.

Por qué el mismo dato produce resultados distintos

Cada vez que se cifra el mismo contenido se generan valores aleatorios nuevos. Esto asegura que el resultado cifrado cambie en cada ocasión y evita que un observador pueda comparar mensajes para inferir información.

Mecanismo de cifrado

1. La aplicación genera al instante una clave efímera
2. Con ellos cifra únicamente el campo sensible del objeto
3. Luego protege esa clave efímera con la clave pública del destinatario
4. El resultado final se guarda como un pequeño paquete con la información necesaria para que el destinatario pueda restaurar el valor original y verificar que no se alteró

En el fragmento de código 5.2 se puede ver la configuración que se tiene para la encriptación de los votos.

Mecanismo de descifrado

1. El destinatario usa su clave privada para recuperar la clave efímera protegida
2. Con esa clave efímera restablece el valor original del campo
3. Durante el proceso se verifica la integridad del paquete. Si hubo cambios o errores el descifrado falla

Fragmento de código 5.2: Encriptación del voto - cryptoFields.ts

```
// 1) AES-256-GCM (no determinista por IV)
const aesKey = randomBytes(32);
const iv = randomBytes(12);
const cipher = createCipheriv("aes-256-gcm", aesKey, iv);
if (aad) cipher.setAAD(Buffer.from(aad));
const ct = Buffer.concat([cipher.update(plain), cipher.
    final()]);
const tag = cipher.getAuthTag();

// 2) Envolver clave AES con publica (RSA-OAEP-SHA256)
const ek = publicEncrypt({ key: publicKeyPEM, oaepHash: "
    sha256" }, aesKey);

const wrapped: WrappedField = {
  v: 1,
  scheme: "RSA-OAEP-256+AES-256-GCM",
  kid,
  iv: b64u.enc(iv),
  tag: b64u.enc(tag),
  ct: b64u.enc(ct),
  ek: b64u.enc(ek),
  aad,
};
```

Comportamiento de la clave pública y la clave privada

- La **clave pública** se comparte y sirve para proteger información destinada al titular de la clave privada
- La **clave privada** se mantiene en secreto y permite recuperar la información. Sin la clave privada el contenido no puede leerse

Propiedades del diseño

- **Confidencialidad** Sólo quien tiene la clave privada correspondiente puede leer el campo protegido
- **Integridad** Si el paquete es modificado el descifrado lo detecta y falla
- **No determinismo** El mismo dato cifrado en momentos distintos produce resultados distintos

Protección y generación de claves

En cuanto a la generación y protección de las claves criptográficas utilizadas para el cifrado y descifrado de los votos, en el prototipo se optó por una implementación simplificada. Para cada ejecución del flujo de votación, es decir, durante el ciclo de vida de la aplicación se genera un único par de claves pública y privada. Este enfoque resulta suficiente para validar el funcionamiento general del sistema, pero no aborda aspectos más avanzados como la persistencia segura, rotación o distribución de claves entre entidades participantes. La gestión adecuada de estas claves constituye una línea de trabajo futuro.

5.1.5. Emisión y Uso de Token

En la figura 5.3, se puede ver el ciclo de vida de un Token en nuestra implementación del prototipo.

En el fragmento de código 5.3 se muestra el método `UseToken`, responsable de validar y registrar el uso del token dentro de la colección privada correspondiente. Esta función verifica la identidad de la organización que intenta utilizar el token, confirma que haya sido emitido por la entidad autorizada y que se encuentre en estado `ISSUED`. Una vez validado, el token se marca como `SPENT`, impidiendo su reutilización y garantizando que cada habilitación se asocie exactamente a un único voto.

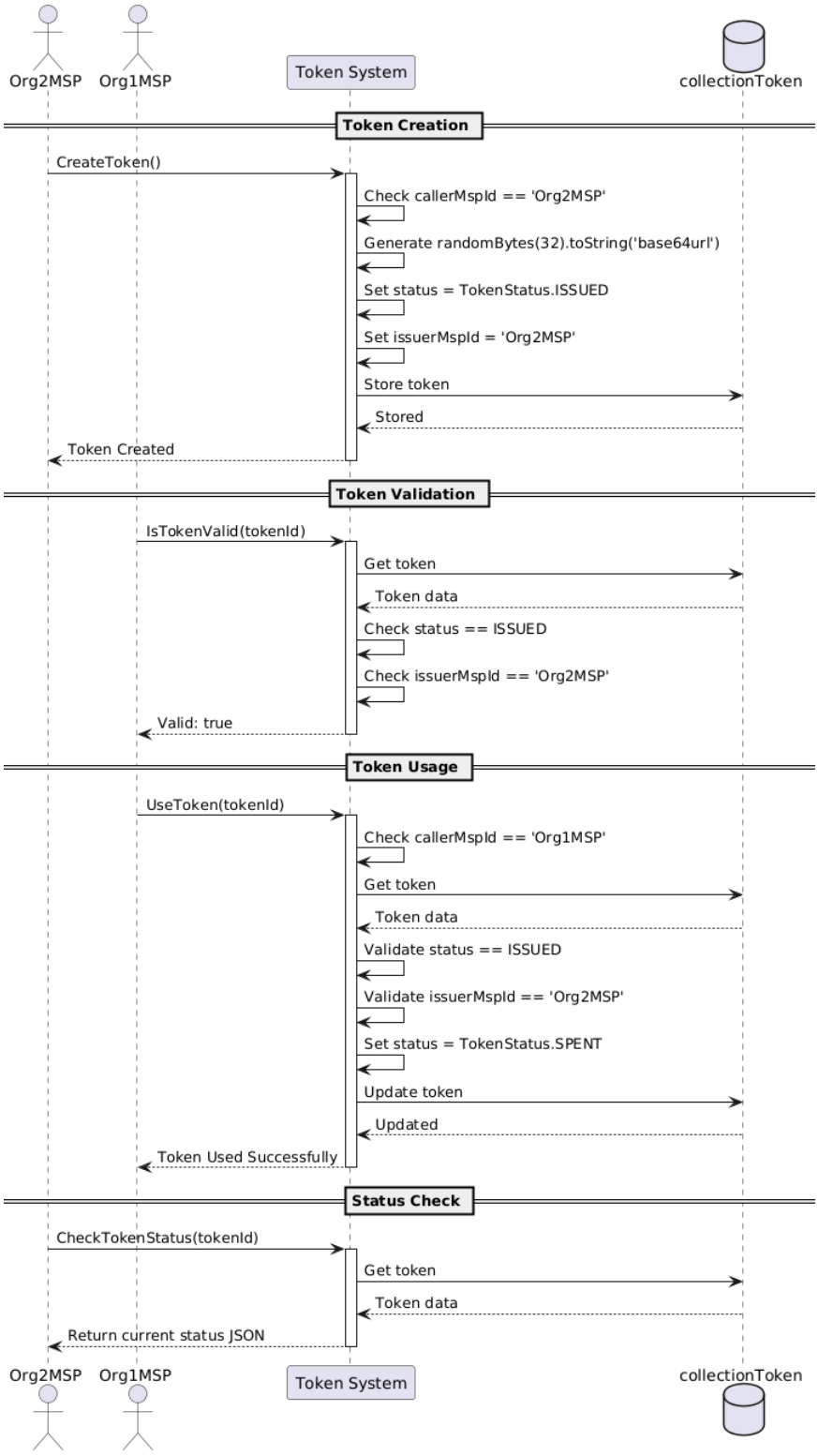

Figura 5.3: Ciclo de vida del token de voto

Fragmento de código 5.3: Uso del Token - TokenEmition.ts

```
@Transaction()
public async UseToken(ctx: Context, tokenId: string):
    Promise<void> {
  const callerMspId = ctx.clientIdentity.getMSPID();

  // Solo Org1MSP (Urna) puede usar tokens
  if (callerMspId !== 'Org1MSP') {
    throw new Error(`Only Org1MSP can use tokens. Caller:
        ${callerMspId}`);
  }

  const tokenBytes = await ctx.stub.getPrivateData(
      assetCollection, tokenId);

  if (!tokenBytes || tokenBytes.length === 0) {
    throw new Error(`Token with ID ${tokenId} not found`);
  }

  const token = this.bytesToToken(tokenBytes);

  // Chequeo que el token este emitido pero no utilizado o
       revocado.
  if (token.status !== TokenStatus.ISSUED) {
    throw new Error(`Token ${tokenId} is not valid.
        Current status: ${token.status}`);
  }

  // Chequeo que el token haya sido emitido por la mesa de
       votacion.
  if (token.issuerMspId !== 'Org2MSP') {
    throw new Error(`Token ${tokenId} was not issued by
        authorized organization. Issuer: ${token.
        issuerMspId}`);
  }

  // Se marca el token como usado.
  token.status = TokenStatus.SPENT;
  await ctx.stub.putPrivateData(assetCollection, token.
      tokenId, this.tokenToUint8Array(token));
}
```

5.1.6. Cierre de Urna

El sistema incorpora un mecanismo de cierre de urnas para consolidar los votos emitidos y proceder al recuento. El proceso de cierre lo inicia exclusivamente Org2MSP (Mesa de votación) mediante el método CloseBallotBox del chaincode TokenEmition, que actualiza el estado global de la urna a `CLOSED`

y registra la fecha y hora de cierre. Una vez cerrada, el chaincode VoteEmition no podrá ingresar más votos, validando el estado de la urna y rechazando automáticamente los intentos de voto. El chaincode VoteCounting exige que el recuento de votos solo se realice cuando la urna esté cerrada, lo que requiere que Org3MSP verifique el estado de cierre antes de acceder a los datos de los votos. Esto se puede ver en el fragmento de código 5.4

Esta implementación busca que los períodos de votación tengan puntos finales definitivos, evita la manipulación de votos posterior al cierre y mantiene la separación de preocupaciones entre las organizaciones de votación (Org1MSP), gestión de elecciones (Org2MSP) y escrutinio (Org3MSP).

Fragmento de código 5.4: Cierre de Urna - TokenEmition.ts

```
@Transaction()
public async CloseBallotBox(ctx: Context): Promise<void> {
  const callerMspId = ctx.clientIdentity.getMSPID();

  // Solo Org2MSP (Mesa de votacion) puede cerrar la urna.
  if (callerMspId !== 'Org2MSP') {
    throw new Error(`Only Org2MSP (Election Board) can
        close ballot box. Caller: ${callerMspId}`);
  }

  await ctx.stub.putState('ballotBoxStatus', Buffer.from(
      BallotBoxStatus.CLOSED));
  await ctx.stub.putState('ballotBoxClosedAt', Buffer.from
      (new Date().toISOString()));
}
```

5.1.7. Organización de los Chaincodes

El prototipo utiliza un modelo de despliegue basado en **dos paquetes**, con una clara **separación de responsabilidades**:

- **Paquete 1: Chaincode de Voto (`vote-private-data`)**
  Contiene dos smart contracts encargados de la gestión de los votos:
  - **Contrato `VoteEmition`** – Gestiona el registro de votos.
    - `CreateVote`: Registra un nuevo voto.
    - `GetAllVotes`: Recupera todos los votos (función administrativa).
    - `GetBallotBoxStatus`: Verifica si la urna se encuentra abierta o cerrada.
  - **Contrato `VoteCounting`** – Gestiona el conteo de votos.
    - `GetVotesForElection`: Recupera los votos correspondientes a una elección (acceso exclusivo de la Autoridad de Escrutinio).
- **Paquete 2: Chaincode de Token (`token-private-data`)**
  Contiene un smart contract que evita el doble voto:

- **Contrato** `TokenEmition` – Administra los tokens de votación y el ciclo de vida de la elección.
  - `CreateToken`: Emite un nuevo token de votación.
  - `UseToken`: Marca un token como utilizado (consumido al emitir el voto).
  - `IsTokenValid`: Verifica si un token aún puede ser utilizado.
  - `CheckTokenStatus`: Recupera información sobre el estado de un token.
  - `CloseBallotBox`: Finaliza el período de votación,, cierra la urna.
  - `GetBallotBoxStatus`: Verifica el estado de la urna.

Cada uno de estos *chaincodes* mantiene su propio *World State* con variables de estado independientes. Por lo tanto, por ejemplo, cuando el *chaincode* de tokens actualiza el estado de la urna a `CLOSED`, este cambio no es directamente visible para el *chaincode* de votos. Para resolver esto, se emplean invocaciones *Cross-Chaincode* (`invokeChaincode`), que permiten a un *chaincode* consultar o invocar transacciones en otro, manteniendo la separación de responsabilidades y la modularidad del sistema. Estas invocaciones preservan el contexto de identidad del invocador original, garantizando que los controles de acceso basados en MSP se mantengan consistentes y que las operaciones no puedan ser suplantadas por organizaciones no autorizadas.

En el fragmento de código 5.1, se puede observar la llamada a un método *UseToken*. A continuación se muestra su implementación:

Fragmento de código 5.5: Invocacion cross-chaincode - VoteEmition.ts

```
const tokenChaincodeName = 'TokenEmition';
//...
      private async useToken(ctx: Context, tokenId: string):
          Promise<void> {
    try {

      // Cross-chaincode invocation: Utilizada para marcar
          el token como utilizado en el chaincode
          TokenEmition al momento de crear un voto.
      await ctx.stub.invokeChaincode(tokenChaincodeName, ['
          UseToken', tokenId], 'mychannel');

    } catch (error) {
      // Manejo de error
    }
  }
```

5.1.8. Ejemplo de flujo de votación

En la figura B.1 del anexo se puede observar un diagrama del flujo de votación completo que abarca el prototipo.

5.1.9. Comportamiento del sistema en casos particulares

Token emitido sin uso

Se podría implementar en una futura iteración un mecanismo de tiempo de revocación de tokens. Los tokens incluirían un campo *revokeTime* que contiene una marca de tiempo específica en la que el token debería volverse inválido. El método *IsTokenValid* existente se optimizaría para comparar la hora actual con *revokeTime* y devolver automáticamente "false"para los tokens que hayan superado su tiempo de revocación, actualizando su estado a `REVOKED`.

Voto en blanco o anulado

Se propone que previo a habilitar la votación, la corte electoral, o la organización pertinente pueda cargar en el sistema las opciones de votación como una lista de elementos. En esta lista se puede habilitar el voto en blanco y el anulado como opciones, en caso de que así lo requiera la elección.

En el caso del voto observado se le podría agregar una flag al elemento del voto y al del token, y que esta sea levantada por parte de la mesa de votación en las situaciones que así lo considere, de forma que el sistema sepa que ese voto es observado.

5.1.10. Integridad, anonimato y validación a posteriori

Integridad: El principio de inmutabilidad propio de la tecnología blockchain garantiza que los votos registrados en el libro mayor sean efectivamente los computados durante el escrutinio. La replicación de los datos entre todos los *peers* de las organizaciones asegura que únicamente las transacciones validadas y consensuadas por los participantes de la red sean incorporadas, preservando la coherencia y la consistencia global del sistema.

Anonimato: El sistema mantiene el anonimato del votante al no establecer ningún vínculo entre la identidad de la persona y el token de votación. La Mesa autentica al ciudadano y le entrega un token válido, pero no registra a quién fue asignado. Posteriormente, la Urna solo verifica si el token fue utilizado o no, y al devolverlo físicamente, el sistema informa su estado sin revelar ninguna asociación con la identidad del votante.

Validación a posteriori: La validación posterior al acto de votación podría delegarse en la Corte Electoral, dado que el prototipo no contempla un mecanismo directo de autenticación dentro de la urna. Sin embargo, esta integración es técnicamente viable, ya que la Mesa —al emitir los certificados o tokens de habilitación— conserva la garantía de que la persona ejerció su voto. Asimismo, podría explorarse la posibilidad de exportar o disponibilizar el *ledger* para que los votantes interactúen con él y verifiquen que su participación fue registrada correctamente, fortaleciendo la transparencia y la confianza en el sistema. Esto es algo con lo que la mayoría de los sistemas de voto electrónico estudiados al momento de analizar el estado del arte contaban para darle tranquilidad a los votantes que su voto había sido ingresado correctamente.

5.2. Análisis de Factibilidad

En esta sección se analiza la factibilidad de la implementación del voto electrónico desde una perspectiva técnica y operativa, considerando tanto el contexto específico de Uruguay como la comparación con alternativas tecnológicas existentes. Se examinan las principales limitaciones y desafíos que podrían surgir en un escenario real de adopción, se contrasta el enfoque propuesto con otras soluciones disponibles, y se evalúa la viabilidad de una implementación gradual que permita mitigar riesgos. Finalmente, se destacan las ventajas del modelo planteado, integrando estos elementos en una valoración global sobre su aplicabilidad y sostenibilidad en el tiempo.

5.2.1. Limitaciones y desafíos para la implementación del voto electrónico en Uruguay

La implementación del voto electrónico en Uruguay enfrenta diversas limitaciones, tanto de tipo legal como social y cultural.

En primer lugar, la principal barrera es de carácter **normativo**. La Ley N.º 7.812, también conocida como Ley Orgánica de Elecciones, establece un procedimiento de votación basado en el uso de papeletas impresas, sobres y urnas físicas, lo cual estructura todo el sistema electoral uruguayo en torno al soporte papel [IMP25]. Esta ley no prevé ninguna forma de votación electrónica, y su eventual sustitución implicaría una reforma profunda del marco legal electoral. Por tanto, aunque la Constitución no prohíbe explícitamente el uso de tecnologías, en la práctica, la legislación vigente constituye una **limitación estructural** para cualquier avance en esta dirección.

A esta restricción legal se suman otras limitaciones importantes de carácter social y cultural. En particular, uno de los desafíos más relevantes es la **desconfianza que podría generar el uso del voto electrónico en una parte significativa de la población**. La confiabilidad del actual sistema de voto en papel que ha funcionado con estabilidad en las últimas décadas, puede provocar que un cambio hacia un sistema más abstracto y tecnológicamente complejo genere temores relacionados con el fraude, la manipulación informática o la pérdida del control ciudadano.

Por otro lado, no debe subestimarse la **brecha generacional y tecnológica** existente. Muchas personas mayores o con baja alfabetización digital podrían experimentar dificultades para adaptarse a un sistema de votación basado en dispositivos electrónicos. Esta situación podría afectar negativamente el principio de igualdad en el ejercicio del sufragio, introduciendo barreras involuntarias para el acceso de determinados sectores de la sociedad al acto electoral. Al mismo tiempo, las nuevas generaciones son cada vez más nativamente digitales, lo que podría llevar a que con el tiempo sean cada vez menos las personas con estas barreras tecnológicas.

Otro obstáculo que puede aparecer son los propios participantes del sistema político. En 2009, el ministro de la Corte Electoral, Edgardo Martínez Zimarioff, lanzó una hipótesis de por qué no se implementa el sistema: "Los grupos

políticos más fuertes tienen la posibilidad de imprimir millones de hojas y repartirlas en todo el país, en cambio los pequeños solo pueden mandar a hacer poquitas. Frente a una urna electrónica, las chicas y las grandes agrupaciones son iguales. Entonces, a los principales jugadores de la política no les conviene hacer modificaciones"[Tap09].

5.2.2. Comparación con otras soluciones

Nuestra solución se acerca al sistema empleado en Argentina con la Boleta Única Electrónica (BUE). En dicho sistema, la urna electrónica imprime en una boleta con chip RFID el voto, y esta se coloca en una urna física [Cla25]. Nosotros apuntamos a reemplazar por completo la urna física, y que, luego de ser autorizado por la mesa de votación en el circuito, el votante se acerque a una urna electrónica y en esta quede almacenada la elección.

El sistema Argentino no es de código abierto, se sabe que las máquinas corren en Linux Ubuntu, y el software de la votación está implementado en Python y Javascript.

5.2.3. Implementación gradual

Como estrategia de adopción progresiva, se propone la implementación de un plan piloto que permita la coexistencia del voto electrónico con los mecanismos tradicionales en papel. Este enfoque busca reducir las barreras de desconfianza, usabilidad y aceptación social asociadas a la introducción de nuevas tecnologías en procesos electorales. En una primera etapa, el sistema electrónico no reemplaza al voto en papel, sino que actúa como un mecanismo complementario y verificable, permitiendo contrastar resultados y evaluar su desempeño en un entorno controlado.

El despliegue inicial del sistema debería focalizarse en elecciones de menor impacto institucional, como elecciones barriales, universitarias o, eventualmente, elecciones municipales o departamentales. Estos contextos parecen escenarios adecuados para validar aspectos técnicos, operativos y organizativos del sistema. En particular, las elecciones universitarias resultan un caso de estudio relevante, ya que involucran mayoritariamente a un electorado joven, que tiende a adaptarse con mayor rapidez a soluciones digitales y puede aportar retroalimentación valiosa sobre la experiencia de uso.

A partir de los resultados obtenidos en estas instancias piloto, y tras sucesivas auditorías técnicas y evaluaciones de confianza pública, el sistema podría ampliar progresivamente su alcance hacia elecciones de mayor escala. Este proceso permitiría una transición gradual, en la que el voto en papel se vaya sustituyendo de forma controlada, minimizando riesgos y fortaleciendo la legitimidad del sistema electrónico antes de su adopción plena en elecciones nacionales.

5.2.4. Ventajas del enfoque propuesto

Dentro de las primeras ventajas del sistema electrónico se podría conseguir un gran ahorro en el papel utilizado para papeletas. Para poner un ejemplo, en las elecciones de octubre de 2009, se presentaron 504 listas en todo el país. En cada uno de los 6.868 circuitos debió haber al menos 30 de cada partido. La cuenta da que 103.844.160 hojas de votación fueron impresas. Pero la cifra no es la real, ya que corresponde solo a las que se pusieron a disposición de la Corte Electoral [Tap09].

También se podría conseguir un conteo de votos casi instantáneo, lo que no solo daría un resultado más rápido, sino que además evitaría que los integrantes de las mesas se queden hasta altas horas de la noche contando las papeletas.

Uruguay cuenta con un nivel tecnológico suficiente para implementar un sistema de voto electrónico. La infraestructura digital del país, junto con una población de aproximadamente 3,5 millones de personas, ofrece una escala manejable que facilita una posible transición ordenada y controlada.

En comparación, Estonia, con 1,5 millones de habitantes, ha desarrollado un ecosistema digital avanzado y utiliza el voto electrónico desde 2005. Hoy, cerca del 40 % de su población vota por medios digitales. Aunque Estonia es un referente en gobierno digital, Uruguay presenta condiciones demográficas y tecnológicas favorables que permiten considerar seriamente este camino.

Capítulo 6

Experimentación

En este capítulo se presenta la etapa de experimentación del sistema propuesto, cuyo objetivo es validar empíricamente las propiedades funcionales y de seguridad definidas en el diseño. En particular, se describen los mecanismos implementados para verificar el control de elegibilidad de los votantes y la preservación de la privacidad durante el proceso de emisión del voto.

6.1. Validación del prototipo

Con el objetivo de validar que únicamente los electores habilitados puedan emitir un voto, se diseñó y ejecutó un conjunto de experimentos centrados en el mecanismo de autenticación basado en tokens. Dicho mecanismo asegura que cada voto aceptado por el sistema esté respaldado por un identificador previamente emitido por la autoridad electoral y validado por la urna electrónica.

El flujo experimental se apoya en un orquestador que inicializa fabric gateways [Hyp25g] independientes para las distintas organizaciones que participan del proceso electoral. Esta separación permite simular interacciones realistas entre organizaciones y evaluar el comportamiento del sistema bajo distintos escenarios de validez de credenciales.

6.1.1. Uso correcto de tokens

Durante la ejecución del experimento, el orquestador invoca repetidamente el proceso de emisión de votos, habilitando de forma controlada escenarios con tokens válidos e inválidos. Esta parametrización permite verificar explícitamente que el sistema rechaza intentos de votación no autorizados sin afectar la ejecución de los casos legítimos. Para los votos considerados legítimos, el flujo obtiene la credencial de votación mediante la invocación al contrato de emisión de tokens de la autoridad electoral. De esta forma, se garantiza que el identificador utilizado proviene exclusivamente de una entidad autorizada. En contraste, los escenarios maliciosos reemplazan este paso por el uso de un identificador

arbitrario, representativo de cualquier token que no haya sido emitido por la autoridad electoral.

Previo a la construcción y envío de la transacción de voto, el sistema realiza una verificación explícita del token contra el contrato de la urna. Esta validación consiste en contrastar el identificador recibido con el registro de tokens emitidos por la autoridad electoral. En caso de que la verificación resulte negativa, el flujo se interrumpe inmediatamente mediante una excepción, evitando que el voto sea persistido en el ledger. De esta manera, se garantiza que ningún voto respaldado por tokens no autorizados pueda ser almacenado.

Únicamente cuando la validación del token es satisfactoria, el sistema procede a sellar la selección del candidato y a publicar la transacción de voto. El token es incorporado dentro de los datos transitorios de la transacción, lo que asegura que toda boleta almacenada en el ledger haya sido previamente autenticada por la urna contra las emisiones oficiales de la autoridad electoral, sin exponer el identificador de forma permanente.

Finalmente, una vez registrado el voto, se consulta el estado del token para marcarlo como utilizado o expirado. Este paso impide la reutilización de la credencial y refuerza la propiedad de unicidad del voto, garantizando que cada token emitido por la autoridad electoral pueda respaldar, como máximo, un único voto aceptado por el sistema.

En conjunto, estos experimentos demuestran que el sistema implementa correctamente el control de elegibilidad de votantes, rechazando de forma determinística cualquier intento de votación que no esté respaldado por un token válido emitido por la autoridad electoral y validado por la urna.

Por ejemplo, en el bloque de código 6.1 se observa cómo se inicializan los contratos y se generan múltiples votos, para posteriormente proceder al cierre de la urna y habilitar así la etapa de escrutinio.

Por su parte, en el bloque 6.2 se muestra cómo se gestiona la validez de los tokens, permitiendo controlar explícitamente los distintos escenarios de votación según el estado y la autenticidad del token presentado.

6.1.2. Privacidad

El mecanismo de autenticación implementado preserva la privacidad del votante al desacoplar explícitamente la identidad del elector del contenido del voto. Durante la autenticación, el elector se valida frente a la autoridad electoral y recibe un token físico que habilita la emisión del sufragio. Dicho token no se encuentra asociado en el ledger a ningún identificador personal del votante, sino que actúa únicamente como una credencial anónima de elegibilidad. En consecuencia, una vez registrado el voto, no es posible trazar la relación entre una boleta almacenada y la identidad del elector que la emitió, garantizando el secreto del voto.

La autoridad electoral conserva la facultad de consultar el estado de los tokens emitidos y determinar si un identificador ya ha sido utilizado. Este control se realiza sin comprometer la privacidad del votante, ya que la validación se limita al estado del token y no requiere vincularlo con información personal.

Fragmento de código 6.1: Flujo de Votacion - application-gateway/index.ts

```
// Incializacion/Obtencion de chaincodes
const { publicKey, privateKey } = generateRsaKeyPairPEM
    ();
const contractBallotBox = getContractVoteBallotBox(
    gatewayBallotBox);
const contractTokenBallotBox = getContractTokenBallotBox
    (gatewayBallotBox);
const contractTokenElectoralBoard =
    getContractTokenElectoralBoard(gatewayElectoralBoard)
    ;
const contractVoteCounting = getContractVoteCounting(
    gatewayScrutiny);

// Creacion de votos: el ultimo parametro indica si el
    token que se usara para votar es valido o no, esto
    con el fin de probar ambos escenarios
await createVote(contractBallotBox,
    contractTokenElectoralBoard, contractTokenBallotBox,
    'Candidato1', crypto.randomUUID(), publicKey, true);
...
// insercion de mas votos
...
// Cerrar la urna para proceder al conteo
await closeBallotBox(contractTokenElectoralBoard);
```

Fragmento de código 6.2: Emision de voto, VoteFlow.ts

```
function createVote(voteContract: Contract,
    electoralBoardContract: Contract, ballotBoxContract:
    Contract, candidate: string, key: string, publicKey:
    string, tokenShouldBeValid: boolean): Promise<void> {
        // ...
        const tokenId = tokenShouldBeValid == true ? await
            createToken(electoralBoardContract) : "
            maliciousToken";
        const isValid = await isTokenValid(ballotBoxContract
            , tokenId);
        if (isValid) {
          console.log("Valid token");
        } else {
          console.log(tokenShouldBeValid ? "ERROR WITH
              VALIDATION => Vote was not created" : "Not
              valid token => Vote was not created");
          throw new Error('Token is not valid');
        }
        // Enviar el voto al chaincode, incluyendo el token
        try {
          await voteContract.submit('CreateVote', {
            transientData: { vote_properties: JSON.stringify
                (voteWithToken) }
          });
          console.log(`Vote created successfully for
              candidate: ${vote.candidate} `);
        } catch (e) {
          console.error('Error creating vote:', e);
        }
        // Verificar el estado del token despues de emitir
            el voto
        const isTokenIssued = await checkTokenStatus(
            electoralBoardContract, tokenId);
        console.log(`Token with id: ${tokenId} now is ${
            isTokenIssued ? 'issued' : 'expired'}`);
```

6.1.3. Control de permisos y estados

Adicionalmente, se realizaron pruebas orientadas a verificar el correcto funcionamiento de los mecanismos de control de permisos organizacionales y de transición de estados del proceso electoral. En particular, se evaluó que únicamente las organizaciones autorizadas puedan ejecutar operaciones sensibles, tales como la emisión de tokens, el cierre de la urna y la invocación del contrato de escrutinio.

El estado de la urna es modelado mediante una variable almacenada en el *world state* del chaincode `TokenEmition`, bajo la clave `'ballotBoxStatus'`, y gestionada exclusivamente por dicho chaincode. Esta variable puede tomar los valores `OPEN` o `CLOSED`, siendo `OPEN` el valor por defecto. La transición al estado `CLOSED` es irreversible y solo puede ser ejecutada por la mesa mediante la transacción `CloseBallotBox`, que se puede ver en el fragmento de código 6.3.

Dado que en Hyperledger Fabric cada chaincode gestiona su propio espacio de nombres en el *world state*, los demás chaincodes no pueden acceder directamente a esta variable: durante el desarrollo se comprobó que si un chaincode como `VoteEmition` intentaba invocar `GetBallotBoxStatus` de forma directa, la variable era visible pero su valor no reflejaba el estado actualizado, comportándose efectivamente como una variable independiente. Por este motivo, el resto de los chaincodes consultan el estado mediante *cross-chaincode invocations* hacia `TokenEmition` (fragmento de código 6.4): la emisión de votos y tokens queda bloqueada una vez cerrada la urna, mientras que el escrutinio, implementado en `VoteCounting`, solo es posible cuando la variable se encuentra en estado `CLOSED`.

Este comportamiento fue validado experimentalmente junto con los mecanismos de control de permisos organizacionales. Durante los experimentos se comprobó empíricamente que cualquier intento de invocar dichas operaciones desde una organización no habilitada es rechazado de forma determinista por el chaincode, garantizando el cumplimiento del modelo de separación de responsabilidades definido en la arquitectura del sistema. Asimismo, se verificó que la transición del estado de la urna desde `OPEN` a `CLOSED` impide correctamente la emisión de nuevos votos una vez finalizado el acto electoral, habilitando únicamente las operaciones correspondientes a la etapa de conteo.

Fragmento de código 6.3: Transacción CloseBallotBox, TokenEmition.ts

```
@Transaction()
public async CloseBallotBox(ctx: Context): Promise<void> {
  const callerMspId = ctx.clientIdentity.getMSPID();
  // Se verifica que sea la Org2 que invoca la operacion
  if (callerMspId !== 'Org2MSP') {
    throw new Error(`Only Org2MSP (Election Board) can
        close ballot box. Caller: ${callerMspId}`);
  }

  // Se actualiza el valor de la variable y se guarda
  await ctx.stub.putState('ballotBoxStatus', Buffer.from(
      BallotBoxStatus.CLOSED));
  await ctx.stub.putState('ballotBoxClosedAt', Buffer.from
      (new Date().toISOString()));

  console.log("Ballot box closed by Election Board.");
}
```

Fragmento de código 6.4: Operacion GetBallotBoxStatus, TokenEmition.ts

```
@Transaction(false)
public async GetBallotBoxStatus(ctx: Context): Promise<
    string> {
  const statusBytes = await ctx.stub.getState('
      ballotBoxStatus');
  return statusBytes.toString() || BallotBoxStatus.OPEN;
}
```

6.2. Blockchain: desafíos y comparación con tecnologías tradicionales

6.2.1. Desafios y limitaciones

Desde el punto de vista técnico, el uso de tecnologías blockchain introduce desafíos adicionales en comparación con arquitecturas tradicionales centralizadas. En particular, los entornos de desarrollo y prueba presentan una mayor complejidad inicial, ya que requieren la configuración y coordinación de múltiples componentes, como nodos, servicios de ordenamiento, canales y contenedores. Esta complejidad contrasta con el despliegue más directo de sistemas convencionales, donde las herramientas y flujos de trabajo están ampliamente estandarizados y cuentan con una curva de aprendizaje más accesible.

En el caso específico de *Hyperledger Fabric*, la experiencia de desarrollo demanda un entendimiento profundo de su arquitectura y del uso de sus herramientas de línea de comandos antes de poder implementar lógica de negocio de manera efectiva. Si bien el ecosistema provee utilidades robustas para el monitoreo y la gestión de contenedores, estas suelen generar un volumen elevado de información, lo que dificulta el análisis de logs y la identificación de errores relevantes. A esto se suma que, en comparación con tecnologías más difundidas, la comunidad y el soporte disponibles son más acotados, lo que puede impactar en los tiempos de desarrollo y en la resolución de problemas, especialmente en etapas tempranas del proyecto.

Desafíos durante el desarrollo del prototipo

Red personalizada: Al inicio del desarrollo se planteó la creación de una red personalizada de *Hyperledger Fabric*, diseñada específicamente para reflejar la arquitectura organizacional del sistema de votación propuesto. Sin embargo, este enfoque enfrentó dificultades significativas derivadas, en gran parte, de la limitado soporte disponible para la configuración manual de redes desde cero. La complejidad de definir correctamente los archivos de configuración base, gestionar los certificados de cada organización y establecer las políticas de consenso adecuadas, sumado a la falta de recursos y ejemplos detallados, resultó en un obstáculo que amenazaba con desviar el foco del proyecto. Ante esta situación, se optó por utilizar la test-network preconfigurada disponible en el repositorio oficial de *Hyperledger Fabric*, que proporciona una topología funcional lista para usar. Esta decisión permitió concentrar los esfuerzos en el desarrollo de la lógica de negocio del sistema electoral, los chaincodes de emisión de tokens, registro de votos y conteo, garantizando la entrega de un prototipo funcional dentro de los plazos establecidos. La implementación de una red personalizada queda identificada como una línea prioritaria de trabajo futuro.

Incorporación dinámica de organizaciones: Durante el desarrollo del prototipo, se intentó implementar la incorporación dinámica de la Autoridad de Escrutinio (Org3MSP) al canal una vez cerrada la urna, aprovechando las capacidades de *Hyperledger Fabric* para actualizar la configuración del canal en

tiempo de ejecución. Sin embargo, este enfoque introdujo complejidades significativas en el proceso de debugging y diagnóstico de errores. La dificultad para identificar la causa raíz de los problemas, ya sea que se originaran en la lógica de los chaincodes, en la configuración del canal, o en el proceso de actualización de la topología de red, llevó a optar por una solución más simple: mantener la organización presente desde el inicio con permisos restrictivos hasta el cierre de la urna. Esta decisión permitió avanzar con la validación funcional del prototipo, dejando la incorporación dinámica como una línea de trabajo futura una vez se cuente con mayor experiencia en la administración y troubleshooting de redes *Hyperledger Fabric*.

6.2.2. Comparación con tecnologías tradicionales

El desarrollo de sistemas basados en *Blockchain* presenta una experiencia de desarrollo menos cómoda en comparación con tecnologías tradicionales. Las herramientas, bibliotecas y entornos de prueba suelen ser más limitados, y la comunidad técnica es considerablemente más reducida que la de plataformas consolidadas en el ámbito del desarrollo web o de aplicaciones empresariales. Esto impacta en la disponibilidad de recursos, documentación y soluciones a problemas comunes, lo que puede incrementar el esfuerzo de implementación y mantenimiento.

Por otro lado, las tecnologías *Blockchain* ofrecen características que resultan ventajosas para el contexto del voto electrónico, en particular la inmutabilidad de los registros y los mecanismos de validación distribuida. En el caso de *Hyperledger Fabric*, el modelo de endoso (*endorsement*) por parte de los *peers* permite que las transacciones sean verificadas de manera descentralizada antes de ser incorporadas al registro, aportando mayor transparencia y trazabilidad respecto a los sistemas tradicionales basados en bases de datos centralizadas.

Una característica fundamental de blockchain que resulta especialmente relevante para sistemas electorales es la redundancia inherente de los datos. En *Hyperledger Fabric*, cada peer de las organizaciones participantes mantiene una copia completa del ledger correspondiente a los canales en los que participa. Esta replicación automática ofrece múltiples ventajas en el contexto del voto electrónico:

- En primer lugar, la disponibilidad del sistema se ve significativamente reforzada. A diferencia de una base de datos centralizada, donde la falla del servidor principal puede comprometer el acceso a toda la información, en una red blockchain la caída de uno o varios nodos no impide que el sistema continúe operando. Los peers restantes mantienen copias idénticas del registro, garantizando la continuidad del servicio durante todo el proceso electoral.

- En segundo lugar, esta redundancia proporciona protección contra la pérdida de datos. En el escenario de un sistema electoral tradicional centralizado, un fallo de hardware, un error de configuración o un ataque malicioso

podrían resultar en la pérdida irrecuperable de votos registrados. En cambio, en la arquitectura propuesta, la destrucción o corrupción de los datos en un nodo no compromete la integridad del registro electoral, ya que múltiples copias verificables permanecen distribuidas entre los participantes de la red.

- Finalmente, la redundancia facilita los mecanismos de auditoría y verificación. Cada organización participante posee su propia copia sincronizada del libro mayor, lo que permite realizar auditorías independientes sin depender de un único custodio de la información. Esta característica resulta particularmente valiosa en un contexto electoral, donde la confianza en el proceso depende de la posibilidad de verificación por parte de múltiples actores con intereses diversos.

Capítulo 7

Conclusiones y Trabajo Futuro

7.1. Conclusiones

El presente trabajo tuvo como objetivo analizar la viabilidad del uso de tecnologías blockchain en el contexto del voto electrónico, abordando el problema desde una perspectiva teórica y práctica. En primer lugar, se realizó un estado del arte que permitió examinar distintas iniciativas y sistemas existentes, identificando sus principales enfoques arquitectónicos y mecanismos de verficiación. Este análisis evidenció que no existe una solución única al problema del voto electrónico, sino diferentes compromisos entre seguridad, transparencia, complejidad técnica y usabilidad.

A partir de dicho análisis se llevó a cabo la especificación, diseño y posterior implementación de un prototipo de urna electrónica basado en Hyperledger Fabric. Este desarrollo permitió validar, en un entorno controlado, aspectos fundamentales del proceso electoral tales como la emisión y validación de tokens, la separación de responsabilidades entre organizaciones, el cierre de urna y el conteo posterior de votos. La implementación constituyó una contribución central del trabajo, ya que permitió no solo estudiar el problema desde el plano conceptual, sino también experimentar con las decisiones de diseño, las restricciones técnicas y los mecanismos de seguridad propios de una red blockchain permisionada.

Durante el desarrollo se constató que la adopción de una plataforma como Hyperledger Fabric implica una complejidad técnica considerable en comparación con tecnologías tradicionales. La configuración de la red, la gestión de identidades, la definición de canales y contratos inteligentes, y la comprensión de la arquitectura subyacente representan una barrera inicial significativa. No obstante, dicha complejidad se ve atenuada con funcionalidades particularmente adecuadas y relevantes para contextos electorales, tales como el control fino de permisos, la trazabilidad de las transacciones, la auditabilidad del sistema y la

inmutabilidad de los registros; características que resultan centrales en procesos donde la confianza y la transparencia son requisitos esenciales.

Asimismo, el trabajo permitió discutir ventajas y limitaciones del uso de blockchain en este dominio. Desde el punto de vista tecnológico, la implementación de un sistema de voto electrónico basado en blockchain es viable y permite diseñar arquitecturas con alto grado de aislamiento, por ejemplo mediante instancias independientes por urna. Sin embargo, la introducción de este tipo de soluciones no depende exclusivamente de su factibilidad técnica. En el caso uruguayo, donde el sistema electoral vigente es ampliamente percibido como sólido y confiable, la adopción de una tecnología disruptiva enfrenta también desafíos institucionales, legales y culturales que deben ser cuidadosamente considerados.

En consecuencia, este trabajo no solo aporta un análisis comparativo y una revisión del estado del arte, sino también la especificación e implementación concreta de un prototipo funcional, junto con una discusión sobre las dificultades encontradas y posibles pasos futuros en el camino hacia un (eventual) sistema de voto electrónico. Más que proponer una sustitución inmediata del sistema actual, se buscó generar evidencia técnica y conceptual que contribuya al debate informado sobre el rol que tecnologías como blockchain podrían desempeñar en eventuales transformaciones del proceso electoral.

7.2. Trabajo futuro

A partir del desarrollo del prototipo y los resultados obtenidos, se identifican varias líneas de trabajo que permitirían avanzar hacia una implementación más robusta del sistema propuesto:

- **Red personalizada de *Hyperledger Fabric*:** Implementar una red propia en lugar de la *test-network* utilizada en el prototipo, con un *Membership Service Provider* (MSP) configurado específicamente y parámetros totalmente adaptables a las necesidades de la Corte Electoral. Esto permitiría un entorno más controlado y representativo de una infraestructura institucional, con políticas de identidad, autenticación y gobernanza acordes a la realidad nacional.
- **Gestión de claves criptográficas:** Mejorar el manejo del par de claves pública y privada empleadas para el cifrado y descifrado de los votos. Actualmente, las claves se generan por ejecución del flujo de votación, sin persistencia ni accesibilidad compartida. Se propone diseñar un mecanismo seguro que garantice su disponibilidad para los miembros de la mesa y su protección a lo largo del ciclo de vida del proceso electoral.
- **Token de habilitación en medio físico:** Incorporar el almacenamiento del token de votación en un soporte físico que contenga el identificador generado por la mesa, habilitando al votante a utilizar la urna. Se plantea explorar la compatibilidad con el módulo *PKCS#11*, empleado por la cédula de identidad electrónica uruguaya, para aprovechar la infraestructura criptográfica ya existente [AGE25].

- **Interfaz de usuario inclusiva:** Desarrollar un *frontend* abarcativo, accesible y equitativo para todos los rangos etarios y sociales. Aunque Uruguay presenta un buen nivel de digitalización, la experiencia de usuario debe ser simple, comprensible y justa, evitando favorecer a usuarios con mayor familiaridad tecnológica.

- **Análisis de requerimientos de infraestructura:** Investigar en profundidad la capacidad de hardware y red necesaria para el despliegue del sistema a gran escala. Este análisis debería contemplar aspectos de disponibilidad, escalabilidad y resiliencia del entorno durante una jornada electoral completa.

- **Integración con la Corte Electoral:** Explorar posibles integraciones con los sistemas institucionales existentes, incluyendo la lista de votantes habilitados y las opciones de voto (candidatos). En el prototipo actual, la validación se realiza a nivel de mesa y los votos son representados como cadenas de texto. Una integración real permitiría aproximarse al funcionamiento operativo de una elección nacional.

- **Estructura organizacional y exportación del *ledger*:** Evaluar la posibilidad de exportar el *ledger* del sistema y adaptar la solución hacia un esquema de *urna única por centro de votación* en lugar de por circuito. Esto implicaría que cada mesa dentro del centro pertenezca a la organización *Mesa* y cada urna a la organización *Urna*. Según la documentación actual de *Hyperledger Fabric*, esta arquitectura resulta viable y representa un paso hacia un despliegue de menor costo.

- **Auditoría del proceso**: A modo de mitigar el desafío de la desconfianza del público en general, podría implementarse un mecanismo de auditoría. En una primera aproximación, este mecanismo podría materializarse como una nueva organización participante del sistema, con acceso limitado a cierta información, como por ejemplo el número de votos registrados, sin intervenir en el proceso de votación en sí.

- **Agregar dinamicamente la organizacion de conteo**: Como línea de trabajo futuro, se propone implementar la incorporación dinámica de Org3MSP al canal una vez finalizada la votación, aprovechando las capacidades de *Hyperledger Fabric* para actualizar la configuración del canal en tiempo de ejecución. Este enfoque presentaría ventajas significativas sobre la implementación actual: reforzaría los principios de separación de responsabilidades al garantizar que la autoridad de escrutinio no tenga ningún tipo de presencia en la red durante el período de votación, eliminaría la necesidad de gestionar permisos restrictivos previos al cierre, y reflejaría con mayor fidelidad el flujo operativo del proceso electoral real. Además, esta arquitectura fortalecería la confianza en el sistema al demostrar de manera técnicamente verificable que el acceso a los datos de votación solo es posible una vez concluido el acto electoral.

Bibliografía

[AGE25] AGESIC. *Autenticación y Firma Avanzada a través de PKCS#11*. Último acceso: 3 de marzo de 2026. 2025. URL: https://www.gub.uy/agencia-gobierno-electronico-sociedad-informacion-conocimiento/comunicacion/publicaciones/documentacion-tecnica-cedula-identidad-chip/documentacion-tecnica-3.

[AM16] Syed Taha Ali y Judy Murray. "An Overview of End-to-End Verifiable Voting Systems". En: *Real-World Electronic Voting: Design, Analysis and Deployment*. Ed. por Feng Hao y Peter Y. A. Ryan. Auerbach Publications, 2016, págs. 171-218. ISBN: 9781315371290. URL: https://arxiv.org/abs/1605.08554.

[Cla25] Clarín. *Elecciones 2025 en Salta: cómo es la boleta y el paso a paso para votar*. Último acceso: 3 de marzo de 2026. 2025. URL: https://www.clarin.com/informacion-general/elecciones-2025-salta-boleta-paso-paso-votar_0_wru2fHho82.html.

[e-E23] e-Estonia. *How did Estonia carry out the world's first mostly online national elections*. Último acceso: 3 de marzo de 2026. 2023. URL: https://e-estonia.com/how-did-estonia-carry-out-the-worlds-first-mostly-online-national-elections/.

[Eth26a] Ethereum Foundation. *¿Qué es Ethereum?* Último acceso: 3 de marzo de 2026. 2026. URL: https://ethereum.org/es/what-is-ethereum/.

[Eth26b] Ethereum Foundation. *Introducción a los contratos inteligentes*. Último acceso: 3 de marzo de 2026. 2026. URL: https://ethereum.org/es/what-is-ethereum/.

[Har10] Harvard Magazine. "Secret Ballots, Verifiable Votes". En: *Harvard Magazine* (mayo de 2010). Último acceso: 3 de marzo de 2026. URL: https://www.harvardmagazine.com/2010/05/secret-ballots-verifiable-votes.

[Hyp17a] Hyperledger. *Introduction to Hyperledger Fabric*. Último acceso: 3 de marzo de 2026. 2017. URL: https://hyperledger-fabric.readthedocs.io/en/release-1.2/blockchain.html.

[Hyp17b] Hyperledger. *Writing Your First Application*. Último acceso: 3 de marzo de 2026. 2017. URL: https://hyperledger-fabric.readthedocs.io/en/release-2.2/write_first_app.html.

[Hyp25a] Hyperledger. *Channels*. Último acceso: 3 de marzo de 2026. 2025. URL: https://hyperledger-fabric.readthedocs.io/en/latest/channels.html.

[Hyp25b] Hyperledger. *Hyperledger Privacy and Confidentiality*. Último acceso: 3 de marzo de 2026. 2025. URL: https://hyperledger-fabric.readthedocs.io/en/release-2.2/whatis.html#privacy-and-confidentiality.

[Hyp25c] Hyperledger. *Hyperledger Private data*. Último acceso: 3 de marzo de 2026. 2025. URL: https://hyperledger-fabric.readthedocs.io/en/latest/private-data/private-data.html.

[Hyp25d] Hyperledger. *Ledger*. Último acceso: 3 de marzo de 2026. 2025. URL: https://hyperledger-fabric.readthedocs.io/en/latest/ledger/ledger.html.

[Hyp25e] Hyperledger. *Ledger*. Último acceso: 3 de marzo de 2026. 2025. URL: https://hyperledger-fabric.readthedocs.io/en/release-2.2/ledger/ledger.html#world-state-database-options.

[Hyp25f] Hyperledger. *Membership Service Providers (MSP)*. Último acceso: 3 de marzo de 2026. 2025. URL: https://hyperledger-fabric.readthedocs.io/en/latest/msp.html.

[Hyp25g] Hyperledger Fabric. *Gateways*. Último acceso: 3 de marzo de 2026. 2025. URL: https://hyperledger-fabric.readthedocs.io/en/latest/gateway.html#fabric-gateway.

[Hyp25h] Hyperledger Fabric. *Hyperledger Fabric Documentation*. Último acceso: 3 de marzo de 2026. 2025. URL: https://hyperledger-fabric.readthedocs.io/en/release-2.5.

[IMP25] IMPO. *Ley N.º 7.812: Ley Orgánica de Elecciones*. Aprobada el 16 de enero de 1925. Último acceso: 3 de marzo de 2026. 1925. URL: https://www.impo.com.uy/bases/leyes/7812-1925.

[Kra25] Kraken Learn team. *What Is a Blockchain Consensus Mechanism?* Último acceso: 3 de marzo de 2026. Feb. de 2025. URL: https://www.kraken.com/learn/what-is-blockchain-consensus-mechanism.

[MF25] Juan Medina y Bruno Ferrando. *Estado del arte Voto Electrónico*. Último acceso: 3 de marzo de 2026. 2025. URL: https://www.fing.edu.uy/~cluna/EdA_VotoElectronico.pdf.

[Mic24a] Microsoft. *Colaboraciones de ElectionGuard*. Último acceso: 3 de marzo de 2026. 2024. URL: https://www.microsoft.com/en-us/research/publication/electionguard-a-cryptographic-toolkit-to-enable-verifiable-elections.

[Mic24b] Microsoft. *ElectionGuard*. Último acceso: 3 de marzo de 2026. 2024. URL: https://www.electionguard.vote/.

[Mic24c] Microsoft. *End-to-End Verifiability in Real World Elections*. Último acceso: 3 de marzo de 2026. 2024. URL: https://www.electionguard.vote/Reports/E2EVerifiability/.

[Nak08] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. Inf. téc. bitcoin.org, 2008. URL: https://bitcoin.org/bitcoin.pdf.

[Nat18] National Academies of Sciences, Engineering, and Medicine. *Securing the Vote: Protecting American Democracy*. Washington, DC: National Academies Press, 2018. ISBN: 978-0-309-47647-8. URL: https://nap.nationalacademies.org/catalog/25120/securing-the-vote-protecting-american-democracy.

[OO14] Diego Ongaro y John Ousterhout. *In Search of an Understandable Consensus Algorithm*. Presentado en USENIX ATC 2014. Último acceso: 3 de marzo de 2026. 2014. URL: https://raft.github.io/raft.pdf.

[Per25a] Perfil. *Boleta Única Electrónica: dudas sobre su funcionamiento y cuánto cuesta cada voto*. Último acceso: 3 de marzo de 2026. 2025. URL: https://www.perfil.com/noticias/bravotv/boleta-unica-electronica-dudas-sobre-su-funcionamiento-y-cuanto-cuesta-cada-voto.phtml.

[Per25b] Perfil. *Elecciones en CABA 2025: habilitaron más de 120 puestos de capacitación para votar con Boleta Única Electrónica*. Último acceso: 3 de marzo de 2026. 2025. URL: https://www.perfil.com/noticias/politica/elecciones-en-caba-2025-se-habilitaron-mas-de-120-puestos-de-capacitacion-para-votar-con-boleta-unica-electronica.phtml.

[SAP24] SAP. *¿Qué es Blockchain?* Último acceso: 3 de marzo de 2026. 2024. URL: https://www.sap.com/latinamerica/products/technology-platform/what-is-blockchain.html.

[Sta24] State Electoral Office of Estonia. *Introduction to i-voting*. Último acceso: 3 de marzo de 2026. 2024. URL: https://www.valimised.ee/en/internet-voting/more-about-i-voting/introduction-i-voting.

[Sza94] Nick Szabo. *Smart Contracts*. Último acceso: 3 de marzo de 2026. 1994. URL: https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html.

[Sza96] Nick Szabo. *Smart Contracts: Building Blocks for Digital Markets*. Publicado originalmente en Extropy #16, 1996. Archivado en la Universidad de Amsterdam. Último acceso: 3 de marzo de 2026. 1996. URL: https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html.

[Tap09] Tapia, Carlos. *Basura electoral en infracción*. Último acceso: 3 de marzo de 2026. 2009. URL: https://www.elpais.com.uy/domingo/basura-electoral-en-infraccion.

[Tri24] Tribunal Superior Electoral. *Urna electrónica*. Último acceso: 3 de marzo de 2026. 2024. URL: https://international.tse.jus.br/es/urna-electronica/presentacion.

[Uni22a] Universidad de Chile. *Participa UChile*. Último acceso: 3 de marzo de 2026. 2022. URL: https://participa.uchile.cl/#/faq.

[Uni22b] Universidad de Chile. *Participa UChile: el nuevo sistema de votación para elecciones en la Universidad de Chile*. Último acceso: 3 de marzo de 2026. 2022. URL: https://participa.uchile.cl.

Anexo A

Código fuente

El código fuente del prototipo con todos sus componentes y la documentación asociada están disponibles en el siguiente repositorio GitLab: Urna Electrónica - https://gitlab.fing.edu.uy/juan.medina.cruz/urna-electronica.

Anexo B

Diagrama de secuencia del flujo de votación

B.1. Flujo de Votación

La figura B.1 presenta una visión general del flujo completo de funcionamiento del prototipo de votación electrónica. En el diagrama se muestran los distintos actores que participan en el proceso y cómo interactúan entre sí a lo largo de las diferentes etapas del sistema. Entre ellos se encuentran la autoridad electoral, la urna, la organización encargada del escrutinio y la red blockchain de Hyperledger Fabric, que actúa como infraestructura para registrar las transacciones de manera distribuida.

El proceso comienza con una fase de inicialización en la que se establecen las conexiones necesarias entre los distintos componentes del sistema y la red blockchain. Durante la etapa de votación, cada elector solicita emitir un voto para un candidato. Para ello, primero se genera y valida un token de votación emitido por la autoridad electoral. Si el token es válido, el sistema crea el voto, cifra la información del candidato utilizando criptografía de clave pública y lo registra en la red. Una vez registrado el voto, el token se marca como utilizado para evitar que pueda emplearse nuevamente.

Luego de finalizado el período de votación, se ejecuta el cierre de la urna electrónica, lo que habilita la etapa de conteo. En esta fase, los votos registrados son recuperados desde el ledger, descifrados utilizando la clave privada correspondiente y contabilizados por candidato para obtener los resultados de la elección. Finalmente, el sistema contempla una fase de auditoría en la que se pueden recuperar todos los votos registrados para su verificación, lo que permite reforzar la transparencia y la trazabilidad del proceso electoral.

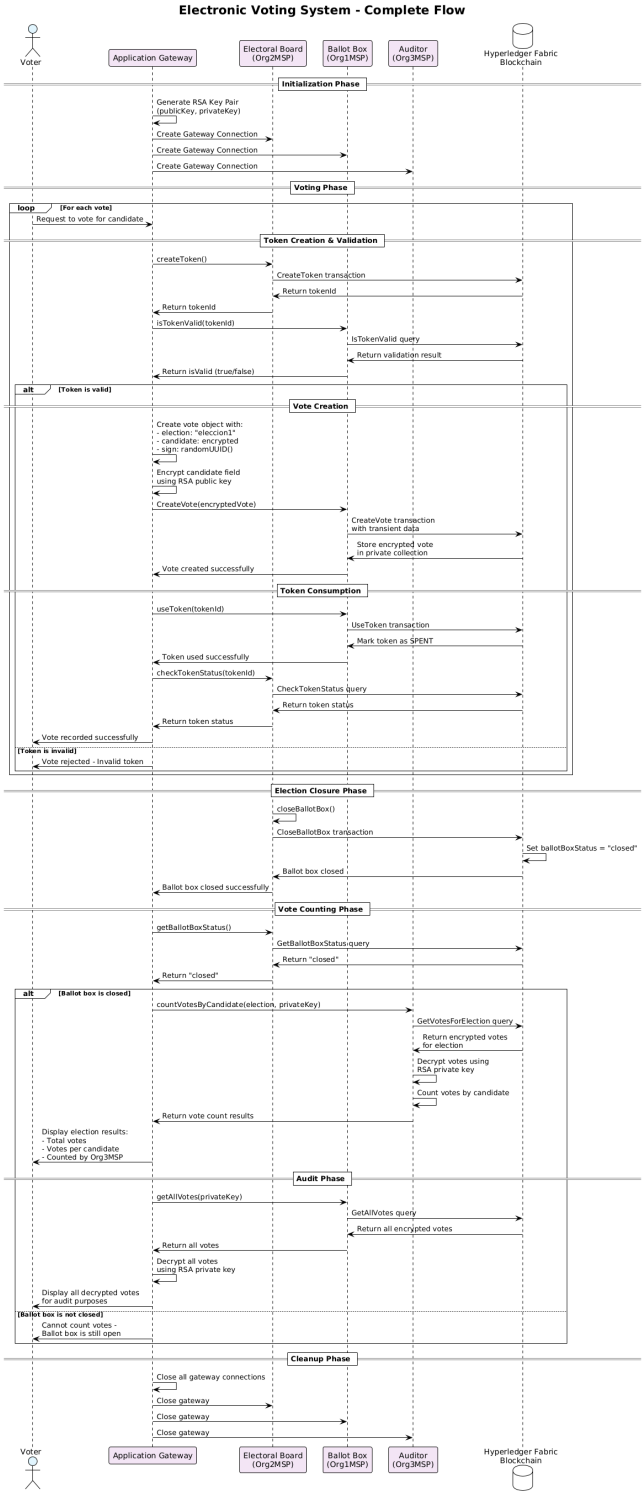

Figura B.1: Diagrama de flujo de votación