Prototipo para apoyar el registro y trazabilidad de estados en el proceso de fotocomparendos aplicando tecnologías de redes distribuidas

Laura Catalina Preciado Ballen

Cristian Stiven Guzman Tovar

Proyecto de Monografía para Optar por el Título de

Ingeniero(a) de Sistemas

Universidad Distrital Francisco José De Caldas

Facultad de Ingeniería

Colombia, Bogotá D.C.

Julio de 2025

Índice

Introducción 1

Formulación del problema . . . 2
Objetivos . . . 9
Impacto esperado . . . 10

Justificación 11

Pertinencia del proyecto . . . 13
Originalidad e innovación . . . 13
Impacto y objetivos . . . 14

Marco teórico 16

Confianza descentralizada . . . 16
Sistemas distribuidos y redes descentralizadas . . . 16
Almacenamiento de evidencias digitales . . . 17
Blockchain: un registro distribuido, inmutable y transparente . . . 17
IPFS: almacenamiento verificable mediante direccionamiento por contenido . . . 18
Arquitectura híbrida blockchain-IPFS . . . 19
Criptografía aplicada . . . 19

Estado del arte 21

Blockchain en gestión gubernamental . . . 21
Integración blockchain-IPFS . . . 21
Cadena de custodia digital . . . 22
Casos de implementación gubernamental . . . 23
Registro de propiedad en Suecia . . . 23
Integridad de registros clínicos en Estonia . . . 24
Síntesis y relevancia para el proyecto . . . 24

Aplicaciones en tránsito 24
Avances significativos 28
Limitaciones identificadas 28
Relevancia del prototipo 29
La relevancia del prototipo se justifica por su potencial para: 30
Tendencias internacionales 30
Sistema de fotocomparendo en Bogotá 35
Funcionamiento de los fotocomparendos en Bogotá: mecanismos, regulación e impacto 35
Análisis crítico del sistema FÉNIX: limitaciones estructurales identificadas . 38

Metodología **41**
Enfoque metodológico de investigación 41
Selección de la pila tecnológica 41
Justificación del uso de blockchain 41
Capa privada: Hyperledger Fabric 44
Capa pública: Ethereum 46
Metodología de desarrollo 46
Justificación de la elección 46
Fases del proceso de desarrollo 47
Ventajas y limitaciones del enfoque 49
Artefactos técnicos del diseño 50

Alcance **51**
Delimitación geográfica 51
Componentes del prototipo 51
Fuera del alcance 52
Entregables 52

Criterios de éxito 52

Limitaciones del prototipo **53**

Entorno de validación 53

Integración y comparación con sistemas existentes 53

Aspectos técnicos y de escalabilidad 54

Seguridad y robustez 54

Diseño del prototipo **57**

Definición de requisitos 57

Diagrama de casos de uso 59

Diagrama de clases 60

Diagramas de actividades 62

Proceso de creación de multa 62

Proceso de apelación de multa 62

Diagrama de despliegue 62

Interfaz de usuario 71

Implementación del prototipo **73**

Entorno de desarrollo y herramientas 73

Stack tecnológico implementado 73

Tecnologías backend 73

Tecnologías blockchain 74

Almacenamiento de evidencias 74

Tecnologías frontend 75

Frameworks de testing 75

Capa pública Ethereum 75

Desarrollo del smart contract 76

Despliegue y configuración 82

Configuración de infraestructura 83
Instalación y despliegue 84
Capa privada Hyperledger Fabric 85
Configuración de la red 85
Desarrollo del chaincode 86
Sincronización entre blockchains 87
Arquitectura del servicio 87
Flujo de sincronización 87
Manejo de errores y reintentos 88
Implementación de IPFS dual 88
IPFS privado 88
IPFS público 89
Desarrollo del backend 89
Arquitectura de servicios 89
Endpoints principales 90
Middleware de seguridad 90
Documentación con Swagger 91
Interfaz de usuario 91
Arquitectura de componentes 91
Gestión de estado 92
Interacción con backend 92
Integración con sistemas externos 92
Simulación de APIs gubernamentales 92
Consideraciones para integración real 93
Desafíos técnicos 93
Compatibilidad de Módulos ESM 93
Optimización de Gas en Ethereum 94

Sincronización Asíncrona 94
Manejo de Archivos Grandes en IPFS 94
Estrategia de validación 94

Plan de pruebas **95**
Concepto de prueba 95
Propósito del plan 95
Alcance de las pruebas 95
Fuera de alcance 96
Entorno de pruebas 96
Tipos de pruebas y casos 97
Pruebas de inmutabilidad 98
Pruebas de rendimiento 100
Casos de prueba funcionales 100
Pruebas de interfaz de usuario 101

Resultados de las pruebas de inmutabilidad y verificabilidad del prototipo **103**
Pruebas de inmutabilidad en blockchain 103
Verificación de integridad con IPFS 103
Verificabilidad del registro 103
Casos de prueba funcionales 104
Casos de prueba de inmutabilidad 104
Pruebas de Rendimiento Básico 105
Cumplimiento de objetivos 106
Pruebas del backend 107
Cobertura de estados del proceso de fotocomparendos 111
Cobertura de estados y extensibilidad del prototipo 114
Estados pendientes de implementación 115

Extensibilidad y trabajo futuro 115
Validación técnica de los estados implementados 116

Discusión y análisis **117**

Arquitectura híbrida 118
Ventajas sobre arquitecturas monolíticas 118
Comparación con el sistema actual (FÉNIX) 118
Comparación con estado del arte 120
Implicaciones y contribuciones 121
Impacto técnico 121
Impacto social 121
Impacto económico 122
Del éxito técnico a la validación operativa: Interpretación crítica de resultados . . 122
Significado técnico de .Exitoso.en blockchain 122
Verification vs. Validation 122
Matriz de Brechas: Entorno de Prueba vs. Producción 122
Interpretación: Lo que el éxito técnico demuestra y lo que no 123
Conclusión: Éxito como condición necesaria pero no suficiente 124
Limitaciones y restricciones 124
Limitaciones Técnicas del Prototipo 124
Limitaciones Metodológicas 125
Limitaciones de Seguridad 125
Lecciones aprendidas 126
Complejidad de Hyperledger Fabric 126
Trade-off Descentralización vs Rendimiento 126
Importancia de UX en sistemas blockchain 126
Diseño Modular para Evolución 126
Despliegue en producción 127

Infraestructura 127
Seguridad 127
Operaciones 127

Conclusiones y trabajo futuro **128**
Viabilidad de implementación 128
Gobernanza de la red blockchain 129
Resolución de debilidades del sistema actual 131
Cierre de ciclo: Oráculos para notificación en la cadena de custodia 135
Trabajo futuro 136

Anexos **138**
Anexo A: repositorios del proyecto 138
Enlaces a los Repositorios 138
Descripción técnica de cada repositorio 138
Instrucciones de acceso 139
Anexo B: manual de usuario 140
Manual para Agentes de Tránsito 140
Manual para ciudadanos 141
Anexo C: glosario de términos 142

Referencias **146**

Índice de figuras

Figura 1. Estadísticas de comparendos emitidos en Bogotá entre enero de 2018 y agosto de 2024 . . . 3
Figura 2. Distribución global de la producción científica sobre blockchain y gestión de infracciones . . . 31
Figura 3. Evolución anual de publicaciones en los países líderes del tema (Brasil, México, Colombia, España y Perú) . . . 32
Figura 4. Nube de palabras de los términos más recurrentes en la literatura sobre blockchain e infracciones . . . 33
Figura 5. Mapa temático de los principales temas de investigación en el área . . . 34
Figura 6. Diagrama de casos de uso del prototipo de gestión de infracciones de tránsito . . . 59
Figura 7. Diagrama de clases del sistema de gestión de multas . . . 61
Figura 8. Diagrama de actividades para el proceso de creación de multa . . . 62
Figura 9. Diagrama de actividades para el proceso de apelación de multa . . . 63
Figura 10. Diagrama de despliegue de la arquitectura del sistema . . . 64
Figura 11. Pantalla de login del sistema . . . 66
Figura 12. Pantalla de recuperación de contraseña . . . 67
Figura 13. Dashboard del agente de tránsito . . . 68
Figura 14. Pantalla de consulta del estado de multa . . . 69
Figura 15. Pantalla de consulta de detalle de multa . . . 70
Figura 16. Pantalla de consulta de multas para propietarios de vehículos . . . 71
Figura 17. Pantalla de detalle de multa para propietarios de vehículos . . . 72

Índice de tablas

1. Variables del problema de investigación 5
2. Comparación entre bases de datos tradicionales y blockchain para gestión de registros gubernamentales 6
3. Comparación entre un modelo centralizado y un modelo descentralizado . . . 11
4. Análisis Comparativo del Estado del Arte en Gestión de Infracciones con *blockchain* 25
5. Violaciones al Estatuto de Contratación Pública (Ley 80/1993) 38
6. Brechas en protección de datos personales 39
7. Deficits de habilitación sectorial 39
8. Comparación de blockchain con otras tecnologías de registro distribuido . . . 42
9. Comparativo de plataformas blockchain para selección de arquitectura híbrida 45
10. Fases del proceso de desarrollo del prototipo 47
11. Mapeo entre estados conceptuales y estados del *smart contract* 77
12. Endpoints principales de la *API REST* 90
13. Casos de prueba funcionales para validar operaciones básicas del sistema . . 97
14. Casos de prueba de inmutabilidad 98
15. Resultados de pruebas de inmutabilidad del sistema 99
16. Casos de prueba de inmutabilidad y verificabilidad del sistema 101
17. Resultados de pruebas funcionales del sistema 104
18. Resumen de casos de prueba de inmutabilidad ejecutados 104
19. Tiempos promedio de operaciones en el entorno de prueba 105
20. Relación entre objetivos específicos, técnicas de validación y resultados . . . 106
21. Resultados de pruebas del backend por módulo 108
22. Validaciones de seguridad implementadas y verificadas 109
23. Validación de cobertura de estados del proceso 111

24. Estados implementados en el prototipo y su relevancia para la validación de integridad. 115
25. Comparación Sistema FÉNIX vs Arquitectura Híbrida 119
26. Brechas Verification-Validation: Laboratorio vs. Producción. El éxito del prototipo se valida en la columna central; el riesgo en producción está en la columna derecha. 123
27. Mapeo de debilidades del sistema FÉNIX a soluciones del prototipo 132
28. Glosario de Términos Técnicos 142

Introducción

En Colombia, la gestión de fotocomparendos ha enfrentado cuestionamientos persistentes en materia de transparencia y confiabilidad de los registros administrativos (Contraloría de Bogotá, 2024). La evidencia disponible indica que el sistema actual, soportado en arquitecturas centralizadas como FÉNIX, presenta limitaciones en seguridad, trazabilidad y confianza ciudadana (Departamento Administrativo de la Función Pública (DAFP), 2021). Estas debilidades se manifiestan en indicadores críticos, como una tasa de impugnación del 34.1 % y más de 155.000 PQRSD tramitadas semestralmente, lo cual evidencia una carga administrativa difícilmente sostenible y un déficit de confianza institucional (Secretaría Distrital de Movilidad, 2024).

Las tecnologías de registro distribuido, particularmente *blockchain*, han demostrado ser efectivas para garantizar la inmutabilidad y verificabilidad de la información en contextos donde la seguridad jurídica y la transparencia son fundamentales, como la gestión gubernamental de datos y el tratamiento de evidencias digitales (Gamero Casado & Fernández Ramos, s.f.; *Sentencia C-112 de 2018*, 2018). A través de contratos inteligentes, es posible automatizar reglas de negocio y procesos de validación, reduciendo la intervención discrecional de los operadores humanos y mitigando riesgos de corrupción y errores administrativos asociados a la gestión de información sancionatoria (Semana, 2023).

En América Latina, estas tecnologías han comenzado a adoptarse en distintos dominios públicos. En Chile, se ha utilizado *blockchain* para certificar datos de la red eléctrica nacional, garantizando transparencia en la determinación de tarifas energéticas (Gálvez y col., 2018). Brasil explora el uso de *blockchain* para el manejo de registros médicos y la interoperabilidad entre instituciones de salud (Azaria y col., 2016). En Colombia, el marco regulatorio ha avanzado mediante iniciativas del Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC) orientadas a la transformación digital del Estado y a la adopción de tecnologías emergentes en el sector público (Ministerio de Tecnologías de la Información y las Comunicaciones, 2020).

A pesar de este potencial, la gestión de infracciones de tránsito ha recibido una atención limitada en la literatura sobre aplicaciones gubernamentales de *blockchain*. Algunas propuestas, como la de Yousfi y col. (2019), plantean sistemas de multas basados en *blockchain* pública (Ethereum), pero afrontan restricciones significativas de privacidad al exponer datos personales sensibles. Otros trabajos, como el de Chen y col. (2024), sugieren arquitecturas híbridas que combinan bases de datos tradicionales con *blockchain*; sin embargo, en estos esquemas la mutabilidad persiste en la capa de datos primaria, debilitando las garantías de integridad. Ninguna de estas aproximaciones aborda de forma específica el desafío de equilibrar transparencia pública y protección de datos personales en consonancia con marcos normativos como el Reglamento General de Protección de Datos (RGPD, 2016) y la Ley 1581 de 2012 en Colombia.

En este contexto, el presente proyecto de grado propone el diseño e implementación de un prototipo híbrido basado en *blockchain* e *InterPlanetary File System* (IPFS) para la gestión de fotocomparendos en Bogotá. El objetivo es fortalecer la integridad de los registros, garantizar la trazabilidad de los estados del proceso y ofrecer un modelo de confianza sustentado en garantías criptográficas más que en procedimientos administrativos. La relevancia de este trabajo radica en su potencial para mejorar la eficiencia operativa, prevenir fraudes y establecer un precedente técnico replicable en otros procesos gubernamentales que requieran altos niveles de seguridad, transparencia y protección de datos en la gestión de información.

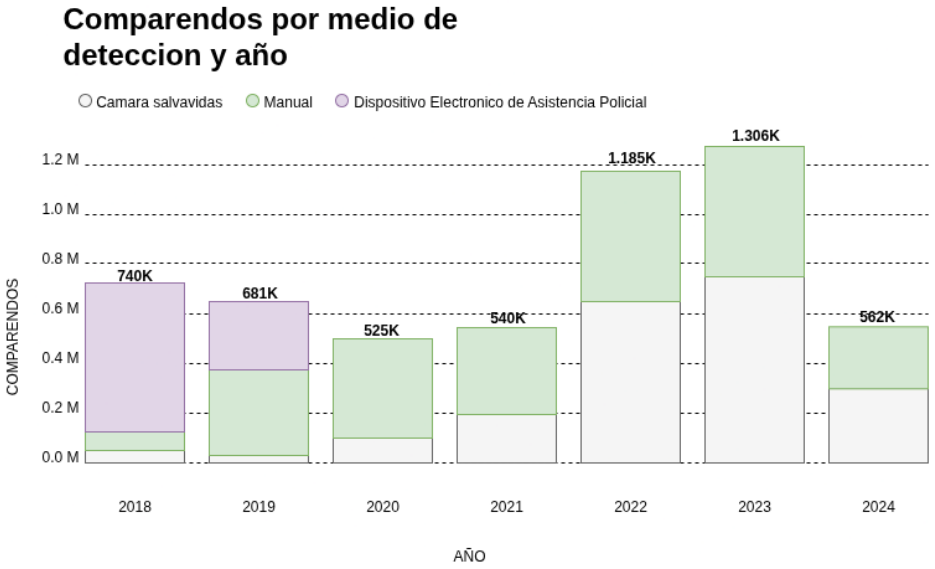
Formulación del problema

La gestión de comparendos en Bogotá es un proceso de gran escala. Según datos del Observatorio de Movilidad, entre enero de 2018 y agosto de 2024 se emitieron más de 1.9 millones de comparendos a través de cámaras salvavidas, evidenciando la importancia sistémica de este proceso para la regulación del tránsito en la ciudad, como se presenta en la Figura 1 se observa los diferentes métodos utilizados para crear los comparendos. Esta operación se apoya en el sistema FÉNIX, una aplicación con infraestructura en la nube,

cuya arquitectura de datos y control de acceso opera bajo un paradigma centralizado.

Figura 1

Estadísticas de comparendos emitidos en Bogotá entre enero de 2018 y agosto de 2024



Nota. Estadísticas de comparendos emitidos en Bogotá entre 2018 y 2024, mostrando los diferentes métodos de creación.

En el sistema actual, la validez e inmutabilidad de los registros de infracciones se fundamenta en los procedimientos administrativos y en la gestión de los funcionarios responsables del sistema (*Sentencia C-112 de 2018*, 2018). Los cambios en la información solo pueden ser detectados por las entidades autorizadas, lo que implica que el control sobre los registros depende directamente de la correcta aplicación de las políticas internas y del seguimiento realizado por dichas entidades («Sentencia No. 123 del 21 de junio de 2019», 2019).

La evidencia generada se conserva bajo un modelo centralizado, en el cual la confianza en la integridad de los datos se sostiene en mecanismos administrativos y controles internos, más que en garantías técnicas accesibles públicamente (Departamento Administrativo de la Función Pública (DAFP), 2021). La potestad sancionatoria y el debido procedimiento

administrativo aseguran la validez de los actos administrativos y la correcta motivación en la imposición de sanciones (Corte Constitucional, 2022; Gamero Casado y Fernández Ramos, s.f.).

De acuerdo con la Auditoría de Cumplimiento de la Contraloría de Bogotá (2024), en el proceso de desarrollo del sistema FÉNIX se identificaron dificultades relacionadas con la supervisión contractual, lo que derivó en retrasos, duplicidad de sistemas y un presunto detrimento patrimonial estimado en más de $8.000 millones de pesos. Estos hallazgos reflejan que, desde su implementación, la plataforma ha enfrentado retos significativos en materia de gobernanza y gestión, los cuales han tenido impacto en la eficiencia administrativa y en la sostenibilidad financiera del proyecto.

Estas debilidades se manifiestan en la operación técnica actual. A nivel operativo, el riesgo de integridad se materializa en una fricción a gran escala con la ciudadanía. Un análisis correlacional de fuentes oficiales para el primer semestre de 2025 revela la magnitud de esta fricción: frente a 457.000 comparendos impuestos [Observatorio de Movilidad, 2025], se gestionaron 155.854 PQRSD [Informe de Gestión PQRSD, 2025].

De estos datos se deduce una Tasa de Impugnación general del 34.1 %, un indicador cuantitativo que sugiere que al menos uno de cada tres actos administrativos del sistema genera una disputa formal, reflejando una carga administrativa insostenible y un déficit de confianza.

La desconfianza generada por estas opacidades y dificultades procesales crea un vacío que es explotado por terceros, afectando directamente al ciudadano. Reportajes de prensa documentan cómo la ausencia de canales oficiales percibidos como confiables ha fomentado la aparición de redes de fraude, como el caso de Juzto.co, donde miles de ciudadanos fueron estafados con promesas de impugnaciones garantizadas, resultando en trámites inconclusos y mayores deudas (Semana - Redacción Nación, 2023).

La identificación de estas limitaciones permite estructurar el problema en torno a variables que reflejan tanto el modelo de confianza actual como sus impactos técnicos, operativos y

financieros. La Tabla 1 sintetiza las variables del problema de investigación y los indicadores asociados, mostrando cómo el paradigma centralizado de gestión condiciona la integridad de los datos, la eficiencia administrativa, la confianza ciudadana y la sostenibilidad del sistema.

Tabla 1. *Variables del problema de investigación*

Variable	**Definición**	**Medición Actual**	**Meta con Prototipo**
Tasa de Impugnación	Porcentaje de comparendos que generan PQRSD por parte de ciudadanos que cuestionan su validez o evidencia	34.1 % (155,854 PQRSD de 457,000 comparendos semestrales)	Reducción esperada por mayor confianza en integridad de evidencia
Detrimento Patrimonial	Pérdida económica estimada para el Distrito por comparendos impugnados exitosamente o declarados nulos	$8,000+ millones de pesos semestrales	Cuantificación de reducción mediante trazabilidad verificable
Carga Operativa PQRSD	Cantidad de solicitudes de petición, queja, reclamo y denuncia que deben procesarse administrativamente	155,854 solicitudes por semestre (2024-I)	Reducción por transparencia y verificabilidad autónoma

Continúa en la siguiente

Tabla 1. *(Continuación)*

Variable	**Definición**	**Medición Actual**	**Meta con Prototipo**
Vulnerabilidad Ciudadana	Exposición del ciudadano a fraudes o manipulación de registros de comparendos (ej. casos Juzto.co)	Casos documentados de suplantación y modificación irregular	Mitigación por inmutabilidad criptográfica y registro distribuido

Nota. Elaboración propia basada en datos de Secretaría Distrital de Movilidad (2024) y Contraloría de Bogotá

Para comprender las implicaciones técnicas de estas variables, la Tabla 2 contrasta las características fundamentales entre el modelo de base de datos convencional actualmente utilizado y una arquitectura basada en blockchain, evidenciando las diferencias en los mecanismos de confianza, inmutabilidad y trazabilidad que motivan la propuesta de este trabajo.

Tabla 2. *Comparación entre bases de datos tradicionales y blockchain para gestión de registros gubernamentales*

Característica	**Base de Datos Convencional**	**Blockchain**
Modelo de confianza	Se basa en un administrador central (entidad de TI)	Confianza distribuida entre múltiples nodos
Inmutabilidad	Registros pueden ser modificados o eliminados por administradores	Los registros son inmutables por diseño

Continúa en la siguiente página

Tabla 2. *(Continuación)*

Característica	**Base de Datos Convencional**	**Blockchain**
Trazabilidad / Auditoría	Depende de la implementación y control interno	Historial completo e inalterable disponible
Riesgo de corrupción interna	Alto, si hay privilegios indebidos o colusión	Bajo, no se puede alterar sin consenso de la red
Seguridad criptográfica	Opcional, no siempre integrada nativamente	Integrada (firmas digitales, hashes, cifrado)
Disponibilidad / tolerancia a fallos	Riesgo de puntos únicos de falla	Alta disponibilidad por replicación descentralizada
Velocidad de operación	Alta velocidad en lectura/escritura	Menor velocidad, prioriza integridad y consenso

Nota. Elaboración propia.

En síntesis, el problema se formula como un Riesgo de Integridad de Datos inherente al paradigma de confianza centralizada del sistema de fotocomparendos. Este riesgo se encuentra documentado por debilidades fundacionales en la gobernanza del proyecto y se manifiesta en consecuencias medibles: (i) una Tasa de Impugnación del 34.1 %; (ii) una carga operativa superior a 155 mil PQRSD semestrales; (iii) un presunto detrimento patrimonial por más de $8.000 millones; y (iv) la vulnerabilidad de la ciudadanía a esquemas fraudulentos derivados de la falta de transparencia institucional.

Para comprender el alcance del problema de integridad, es fundamental identificar los estados por los que transita un comparendo desde su emisión hasta su resolución final. El proceso de fotocomparendos en Bogotá contempla los siguientes estados principales:

1. **GENERADA:** Estado inicial del comparendo, creado por un agente de tránsito o sistema automático de fotodetección al registrar una infracción. En este estado se capturan los metadatos esenciales: placa del vehículo, tipo de infracción, fecha, hora, ubicación y evidencia fotográfica.

2. **NOTIFICADA:** El comparendo ha sido oficialmente notificado al ciudadano propietario o conductor del vehículo, según lo establecido en el Código Nacional de Tránsito. La notificación puede realizarse de forma electrónica, por correo certificado o personalmente. Este estado habilita los términos legales para respuesta ciudadana.

3. **PENDIENTE_RESPUESTA:** El ciudadano ha sido notificado y se encuentra dentro del plazo legal para tomar una acción: pagar voluntariamente con descuento, solicitar acuerdos de pago, o presentar una apelación mediante PQRSD. Este es un estado crítico donde la integridad de la evidencia es fundamental para la toma de decisiones.

4. **EN_APELACION:** El ciudadano ha presentado formalmente una Petición, Queja, Reclamo, Sugerencia o Denuncia (PQRSD) cuestionando la validez del comparendo. Durante este estado, la autoridad de tránsito debe revisar la evidencia, verificar el cumplimiento del debido proceso, y emitir una resolución motivada. La trazabilidad completa del comparendo es esencial en esta etapa.

5. **RESUELTA_APELACION:** La autoridad competente ha tomado una decisión sobre la apelación presentada, que puede ser: (a) confirmar el comparendo, (b) revocarlo total o parcialmente, o (c) declararlo nulo por defectos procedimentales. La decisión debe estar debidamente fundamentada y ser notificada al ciudadano.

6. **PAGADA:** El comparendo ha sido pagado, ya sea voluntariamente por el ciudadano, mediante acuerdo de pago, o tras agotar el proceso de apelación con resultado confirmatorio. El pago liquida la obligación económica pero el registro del comparendo permanece en el sistema con fines estadísticos y de antecedentes.

7. **CANCELADA:** El comparendo ha sido cancelado administrativamente por razones como: anulación judicial, revocación por defectos procedimentales, o corrección de errores en la

emisión (ej. placa incorrecta). Este estado requiere la mayor trazabilidad posible para prevenir cancelaciones irregulares.

8. **CERRADA:** Estado final y definitivo del comparendo. Todas las acciones administrativas, judiciales o de pago han concluido. El registro se mantiene de forma permanente en el sistema con fines de auditoría, estadística y consulta histórica.

Cada transición entre estados representa un punto crítico donde la alteración, pérdida o manipulación de datos puede comprometer la validez del acto administrativo y generar las consecuencias descritas anteriormente: impugnaciones, detrimento patrimonial y vulnerabilidad ciudadana. En el desarrollo del proyecto se identifican explícitamente como puntos de intervención técnica: (i) el registro inicial de metadatos y evidencia en los estados **GENERADA** y **NOTIFICADA**; (ii) la trazabilidad de cambios y decisiones durante **PENDIENTE_RESPUESTA**, **EN_APELACION** y **RESUELTA_APELACION**, con especial énfasis en **CANCELADA**; y (iii) la verificación independiente y permanente del cierre del caso en **CERRADA**.

Ante este panorama, surge la necesidad de explorar arquitecturas que permitan sustituir la confianza administrativa por garantías criptográficas. La pregunta central que guía este trabajo es: **¿Cómo mitigar el riesgo de pérdida o alteración de la integridad de los datos asociados a todos los estados en el proceso de fotocomparendos en Bogotá mediante el uso de tecnologías de redes distribuidas que garanticen el registro, la trazabilidad, la autenticidad y la confidencialidad de la información?**

Objetivos

Objetivo General. Desarrollar un prototipo software tecnológico que facilite el registro y la trazabilidad de los estados en el proceso de fotocomparendos en Bogotá, mediante la aplicación de tecnologías de redes distribuidas, para el fortalecimiento de la integridad y autenticidad de la información reduciendo los riesgos asociados a su confidencialidad.

Objetivos específicos.

- Analizar el proceso actual de registro de fotocomparendos en Bogotá, a partir del marco jurídico y regulatorio que lo rige y de los informes de auditoría emitidos por la secretaria

distrital de movilidad sobre la gestión de comparendos, para identificar vulnerabilidades, requisitos funcionales y no funcionales.

- Desarrollar un prototipo con arquitectura híbrida basado en *blockchain* permisionado (*Hyperledger Fabric*) y *blockchain* público (*Ethereum*), integrando almacenamiento distribuido mediante *IPFS*, asegurando que cada transacción incorpore los metadatos del comparendo y disponiendo de una interfaz de programación de aplicaciones (API *REST*) que permita operaciones de registro, consulta y verificación de estados.
- Evaluar la viabilidad técnica y funcional del prototipo a través de un plan de pruebas que incluya: pruebas de inmutabilidad ante intentos de modificación de registros, verificación de trazabilidad mediante validación de *hashes* criptográficos en cada transición de estado, análisis de rendimiento para validar tiempos de respuesta $\leq$3 segundos, y evaluación de integridad de documentos mediante *Content Identifiers* (*CID*) de *IPFS*.

Impacto esperado

El desarrollo de este prototipo tiene como propósito demostrar la viabilidad técnica de una arquitectura descentralizada para la gestión de fotocomparendos, con el potencial de:

- **Fortalecer la confianza ciudadana:** Mediante la verificación independiente de infracciones y el acceso transparente a la información, sin intermediarios.
- **Reducir la fricción operativa:** Disminuir los recursos destinados a la gestión de PQRSD y disputas administrativas, actualmente estimados en más de 155.000 casos semestrales.
- **Prevenir fraudes:** Mitigar la vulnerabilidad de los ciudadanos ante esquemas de estafa derivados de la falta de canales oficiales confiables.
- **Establecer un precedente técnico:** Servir como referencia para la implementación de soluciones similares en otros procesos gubernamentales que requieran alta integridad de datos.

Nota: Para la especificación detallada del alcance del proyecto, los componentes del prototipo, criterios de éxito y limitaciones metodológicas, consultar la sección Alcance.

Justificación

La gestión de registros públicos, como los fotocomparendos, en arquitecturas centralizadas presenta debilidades en materia de seguridad, transparencia y auditabilidad. En Bogotá, el sistema FÉNIX ilustra estos desafíos, según auditorías de la Contraloría (Contraloría General de la República de Colombia, 2024) (Contraloría General de la República de Colombia, 2023), que destacan limitaciones en la integridad de los datos y una fricción operativa reflejada en más de 153.000 PQRSD en un semestre. Esta situación subraya la necesidad de modelos arquitectónicos alternativos que fortalezcan la confianza pública, independizándola de la dependencia exclusiva en procedimientos y administradores internos. Se transita así de un sistema donde la integridad se presume y se audita retrospectivamente, a uno donde es intrínseca y verificable criptográficamente desde el origen.

El propósito de este proyecto no es modificar el sistema actual, sino diseñar y evaluar un prototipo autocontenido que demuestre un modelo de confianza diferente. Para ilustrar las diferencias estructurales entre el modelo convencional y el propuesto, la Tabla 3 compara sus características clave:

Tabla 3. *Comparación entre un modelo centralizado y un modelo descentralizado*

Característica	**Modelo Centralizado**	**Modelo Descentralizado**	**Relevancia Contextual**
Modelo de Confianza	Basado en la confianza en los administradores del sistema y en la robustez de los controles internos definidos.	Basado en un consenso criptográfico distribuido, donde la confianza reside en el protocolo y no en un intermediario.	La correcta asignación de roles es fundamental. La auditoría observó “ausencia de un profesional responsable de Seguridad de la Información” (págs. 20–25), subrayando la criticidad de los factores de gobernanza.

Tabla 3. *(Continuación)*

Característica	**Modelo Centralizado**	**Modelo Descentralizado**	**Relevancia Contextual**
Integridad de Datos	La integridad se asegura mediante controles de acceso y logs de auditoría internos gestionados por la entidad.	La integridad es una propiedad intrínseca de la estructura de datos; los registros son inmutables por diseño.	La efectividad de los controles internos es fundamental. La auditoría documentó "Falta de control sobre la integridad y calidad de los datos migrados" (págs. 38–40) como punto de atención.
Gestión de Seguridad	Dependiente de políticas y procedimientos de seguridad definidos y ejecutados por la institución.	La seguridad es una propiedad inherente a la capa de protocolo, auditada de forma continua y global por la comunidad.	La formalización de procedimientos es clave. La auditoría identificó "falta de gestión formal de riesgos y controles" y "ausencia de un plan de seguridad para la infraestructura en la nube" (págs. 25–30).
Auditabilidad y Trazabilidad	La auditoría se realiza a través de logs internos, con acceso gestionado por la entidad y sujeto a sus políticas de retención y seguridad.	La traza de auditoría es transparente, inalterable por diseño y públicamente verificable por cualquier actor autorizado.	La consistencia de los registros internos es un factor de éxito. La auditoría observó "retrasos y baja velocidad de desarrollo" (págs. 15–20), subrayando la importancia de una gobernanza rigurosa.

Nota. Elaboración propia, con hallazgos basados en la Auditoría de Cumplimiento No. 90 de la Contraloría de Bogotá D.C. (octubre de 2023) y la Auditoría de Cumplimiento 170100-0054-24.

Pertinencia del proyecto

La pertinencia de este proyecto se enmarca en tres dimensiones complementarias que justifican la necesidad de explorar arquitecturas descentralizadas para la gestión de registros públicos críticos:

- **Social y ciudadana:** En un contexto donde la desconfianza en los procesos administrativos genera una tasa de impugnación del 34.1 % (más de 155.000 PQRSD semestrales según se detalla en el Estado del Arte, subsección sobre el contexto de Bogotá), este proyecto ofrece un modelo alternativo que responde a la necesidad de transparencia, permitiendo la verificación independiente y empoderando al ciudadano con herramientas de auditoría directa sobre la autenticidad de las sanciones.
- **Tecnológica:** Demuestra cómo la integración de *blockchain* (para registros inmutables) e *IPFS* (para evidencias con contenido direccionable) puede abordar los desafíos de seguridad y trazabilidad documentados en sistemas centralizados, respondiendo específicamente a las limitaciones estructurales del sistema FÉNIX identificadas por la Contraloría de Bogotá (ver análisis detallado en el Estado del Arte).
- **Legal e institucional:** El prototipo se alinea con los principios de eficiencia, transparencia y rendición de cuentas exigidos por los organismos de control. Frente a los incumplimientos normativos y brechas de protección de datos identificados en el sistema actual, esta propuesta sirve como caso de estudio sobre cómo garantías técnicas intrínsecas pueden fortalecer el cumplimiento normativo y reducir los riesgos de detrimento patrimonial asociados a modelos centralizados.

Para un análisis detallado del contexto operativo y legal del sistema actual de fotocomparendos en Bogotá, así como de las limitaciones críticas identificadas en FÉNIX que motivan esta propuesta, consultar la subsección correspondiente en el Estado del Arte.

Originalidad e innovación

La innovación de esta monografía radica en la concepción del prototipo como un laboratorio experimental para un nuevo modelo de confianza aplicado a la gestión de fotocomparendos en Bogotá. A diferencia del sistema FÉNIX actual, que depende de controles administrativos centralizados y ha generado más de 155.000 PQRSD semestrales debido a disputas por falta de transparencia, esta propuesta implementa un modelo distribuido resistente a la manipulación mediante:

- **Inmutabilidad criptográfica:** Cada fotocomparendo queda registrado en blockchain con un hash único que impide alteraciones posteriores, resolviendo el problema de integridad que ha generado una tasa de impugnación del 34.1 % en el sistema actual.
- **Gobernanza automatizada:** Contratos inteligentes (*smart contracts*) ejecutan las reglas de tránsito de manera predecible, eliminando la discrecionalidad administrativa que ha derivado en un detrimento patrimonial estimado en más de $8.000 millones.
- **Almacenamiento descentralizado:** *IPFS* garantiza que las evidencias fotográficas sean inalterables y accesibles sin intermediarios, abordando las vulnerabilidades de confidencialidad identificadas en auditorías de la Contraloría.

La DApp funciona como una prueba de concepto que integra estas tecnologías para demostrar una solución a problemas específicos de gestión pública que las bases de datos centralizadas no pueden resolver de manera nativa: la verificación independiente de más de 1.9 millones de comparendos emitidos entre 2018 y 2024, sin depender de la confianza en instituciones centralizadas.

Impacto y objetivos

Este prototipo responde directamente a la problemática documentada en el sistema de fotocomparendos de Bogotá y se posiciona como una contribución pionera al GovTech colombiano. El impacto esperado se materializa en dimensiones cuantificables que abordan los desafíos específicos identificados por la Contraloría de Bogotá:

- **Confianza por Diseño:** La verificación independiente mediante *blockchain* fortalece la legitimidad de los procesos públicos, permitiendo que ciudadanos y autoridades verifiquen la autenticidad de cualquier comparendo sin intermediarios. Esto cumple con el objetivo específico de garantizar integridad y autenticidad, potencialmente reduciendo la tasa de impugnación del 34.1 % actual mediante evidencia técnica irrefutable.
- **Gobernanza Automatizada:** Los contratos inteligentes (*smart contracts*) ejecutan automáticamente las reglas del Código Nacional de Tránsito, desde la detección hasta la resolución de apelaciones, reduciendo la dependencia de supervisión humana que ha generado sobrecargas administrativas superiores a 155.000 PQRSD semestrales. Esta automatización se alinea con el objetivo de desarrollar un sistema transparente y confiable que minimice riesgos de corrupción inherentes a procesos manuales.
- **Escalabilidad en GovTech:** Este caso de uso de fotocomparendos es directamente transferible a otros procesos críticos como permisos de construcción, licencias ambientales o registros civiles. El

prototipo establece un precedente técnico replicable que puede extenderse a más de 20 procesos administrativos identificados en el Plan Distrital de Gobierno Digital, cumpliendo con el objetivo de crear un modelo escalable para la administración pública colombiana.

La adopción de *blockchain* e *IPFS* en esta propuesta no representa una preferencia tecnológica, sino una respuesta técnica deliberada y fundamentada a los desafíos específicos de integridad identificados en el sistema FÉNIX: proponiendo una arquitectura donde la veracidad no se presume ni se audita retrospectivamente, sino que es una propiedad intrínseca y verificable criptográficamente desde el momento de registro de cada comparendo.

Marco teórico

El marco conceptual y técnologico que sustenta la propuesta del prototipo, presentan las teorías y modelos clave que justifican la selección de blockchain e IPFS como componentes centrales, evidencian los principios inherentes de integridad, transparencia, resiliencia y auditabilidad en la gestión de evidencia digital crítica como los fotocomparendos.

Confianza descentralizada

Los sistemas de información tradicionales suelen depender de intermediarios centralizados o autoridades certificadoras para validar transacciones y garantizar la fiabilidad de los registros. La teoría de los modelos de confianza descentralizada, en cambio, analiza cómo establecer y mantener la confianza en entornos distribuidos donde tales autoridades centrales están ausentes (Swan, 2015).

La relevancia de este modelo es fundamental para justificar el uso de la tecnología blockchain en la gestión de fotocomparendos, ya que su propósito es precisamente reemplazar la necesidad de depositar confianza exclusiva en una única entidad para la custodia, validación e integridad de los registros. Blockchain habilita un cambio de paradigma: en lugar de confiar en un actor central, la confianza se distribuye y se deposita en la robustez del protocolo criptográfico subyacente (Nakamoto, 2008), en la transparencia de las reglas del sistema y en el consenso mayoritario de los participantes de la red (Antonopoulos & Harding, 2023). Este enfoque reduce drásticamente los puntos únicos de fallo y los vectores de corrupción asociados a la dependencia de intermediarios centralizados, quienes podrían ser comprometidos, cometer errores o actuar de manera malintencionada.

Sistemas distribuidos y redes descentralizadas

El paradigma de la confianza descentralizada se sustenta en la teoría de los sistemas distribuidos, donde múltiples entidades autónomas, denominadas nodos, colaboran a través de una red para alcanzar un objetivo común, compartiendo tanto la carga computacional como el almacenamiento de datos (van Steen & Tanenbaum, 2017). Estos sistemas se fundamentan en principios como la distribución de recursos, la comunicación inter-nodo y mecanismos de coordinación que prescinden de intermediarios centrales (Coulouris y col., 2011).

La relevancia de esta teoría para el presente proyecto es primordial, ya que tanto blockchain como el InterPlanetary File System (IPFS) son implementaciones nativas de sistemas distribuidos. Su adopción conjunta promueve inherentemente:

- **Resiliencia:** Al eliminar puntos únicos de fallo (Single Points of Failure - SPOF).
- **Alta Disponibilidad:** Al permitir el acceso a datos y servicios desde múltiples nodos.

- **Resistencia a la Censura:** Dado que ninguna entidad individual posee control absoluto sobre la red o los datos almacenados (Antonopoulos & Harding, 2023).

Una característica esencial de estos sistemas es su arquitectura de red **Peer-to-Peer (P2P)**, donde los participantes se conectan y comparten recursos directamente entre sí, sin necesidad de un servidor central. En una red P2P, cada nodo puede actuar simultáneamente como cliente y servidor, lo que posibilita que el registro distribuido (ledger) se mantenga sincronizado y que los archivos puedan ser recuperados desde múltiples fuentes, garantizando la integridad de la información sin depender de una autoridad central.

Redes distribuidas vs. redes descentralizadas. En la literatura especializada se distingue entre *sistemas distribuidos* y *sistemas descentralizados*, aunque en muchos trabajos los términos se usen de forma intercambiable. Un sistema distribuido, en sentido amplio, es aquel en el que múltiples nodos cooperan para ofrecer un servicio único, compartiendo estado y comunicación a través de la red (Coulouris y col., 2011; van Steen & Tanenbaum, 2017). La descentralización, por su parte, hace referencia al grado en que el control y la toma de decisiones se reparten entre los participantes, reduciendo o eliminando la existencia de un único punto de autoridad o de fallo (Antonopoulos & Harding, 2023; Swan, 2015). En este trabajo se utiliza el término **"tecnologías de redes distribuidas"** en el título porque enfatiza la dimensión arquitectónica (distribución de nodos, datos y procesos) que comparten tanto Hyperledger Fabric como Ethereum e IPFS, mientras que el análisis conceptual reconoce que la solución propuesta implementa **mecanismos de descentralización de la confianza** —particularmente en la verificación ciudadana y la inmutabilidad de los registros— sobre dicha infraestructura distribuida. Esta elección terminológica se alinea con la definición amplia de *Distributed Ledger Technologies (DLT)* como categoría paraguas para blockchains públicas y permisionadas (Narayanan y col., 2016; Ruan y col., 2021), y permite dialogar con la literatura que utiliza indistintamente los términos "red distribuida" y "red descentralizada" mantenido un uso riguroso en el cuerpo del documento.

Almacenamiento de evidencias digitales

Para materializar un sistema de gestión de fotocomparendos descentralizado, se requiere la sinergia de dos tipos de tecnologías: una para el registro inmutable de transacciones y otra para el almacenamiento verificable de la evidencia.

Blockchain: un registro distribuido, inmutable y transparente

Blockchain es un tipo específico de Tecnología de Ledger Distribuido (DLT), un sistema de registro digital caracterizado por ser distribuido, sincronizado y asegurado criptográficamente entre múltiples participantes (Narayanan y col., 2016). Su estructura fundamental se compone de **transacciones** —operaciones firmadas digitalmente que modifican el estado del ledger de forma permanente (Antonopoulos & Harding,

2023)— agrupadas en bloques. Cada bloque contiene un hash criptográfico que lo vincula al anterior, formando una cadena cronológica e inmutable.

La **inmutabilidad** y la **transparencia** son los beneficios centrales que esta tecnología aporta (Antonopoulos & Harding, 2023; Swan, 2015). La primera se logra mediante la estructura encadenada y los mecanismos de consenso distribuido (ej., Proof-of-Work (Nakamoto, 2008) o Proof-of-Stake (King & Nadal, 2012)), que hacen que la modificación de un bloque pasado sea computacionalmente prohibitiva. La segunda se habilita por la naturaleza replicada del ledger, permitiendo que actores autorizados puedan consultar y verificar la información de forma independiente. Dentro de este ecosistema, los **Smart Contracts** (Contratos Inteligentes) actúan como programas autoejecutables cuyo código define e impone automáticamente los términos de un proceso, permitiendo automatizar la gestión del ciclo de vida del comparendo (Buterin, 2014; Szabo, 1997; Wood, 2014).

Modelos arquitectónicos y elección para el prototipo. La tecnología blockchain no es monolítica; existen diferentes arquitecturas:

- **Públicas (Permissionless):** Abiertas a cualquier participante, priorizan la descentralización radical (ej. Bitcoin, Ethereum) (Nakamoto, 2008).
- **Privadas:** Controladas por una única entidad, ofrecen alta eficiencia pero son centralizadas.
- **De Consorcio/Permisionadas (Permissioned):** Operadas por un grupo selecto de participantes autorizados. Ofrecen un equilibrio entre descentralización, rendimiento y confidencialidad, siendo la opción ideal para contextos gubernamentales y empresariales (Cachin, 2018; Vukolic', 2015).

Para este prototipo, se opta por una **implementación permisionada** (simulada con Hyperledger Fabric), permitiendo que solo entidades autorizadas operen nodos y registren transacciones, con un mecanismo de consenso eficiente (ej. Raft) adecuado para un sistema de gestión de registros.

IPFS: almacenamiento verificable mediante direccionamiento por contenido

Los ledgers de blockchain no están optimizados para almacenar grandes volúmenes de datos (blobs), como las imágenes de los fotocomparendos (Xu y col., 2019). Para resolver esto, se utiliza un sistema de almacenamiento descentralizado. La elección de IPFS sobre alternativas centralizadas como AWS S3 es crucial para la integridad del sistema. Mientras que en un sistema centralizado el propietario puede modificar o eliminar unilateralmente un archivo (Vogels, 2008), IPFS opera bajo el paradigma del **direccionamiento por contenido (Content Addressing)** (Benet, 2014; Voigt & von dem Bussche, 2017).

En este modelo, la identidad única de un archivo, su Content Identifier (CID), es un **hash criptográfico**

derivado directamente de su contenido. Esto establece un vínculo intrínseco e inmutable: si el contenido del archivo cambia, incluso mínimamente, su CID también cambiará. IPFS es un protocolo y red P2P que utiliza este principio: divide los archivos en bloques, calcula sus hashes y permite su recuperación a través de su CID, utilizando mecanismos como DHT para localizar los nodos que los poseen (Benet, 2014; Maymounkov & Mazieres, 2002).

Arquitectura híbrida blockchain-IPFS

La integración de ambas tecnologías en una arquitectura híbrida sigue el siguiente flujo de trabajo:

1. La imagen probatoria del comparendo se carga a un nodo IPFS, obteniendo su CID único.
2. Se crea una transacción en la blockchain (on-chain) que contiene este CID junto con los metadatos esenciales del comparendo (fecha, hora, lugar, placa).
3. Esta transacción se valida y registra de forma inmutable en el ledger.

Este enfoque crea un enlace criptográfico inalterable entre el registro oficial (en blockchain) y la evidencia visual original (en IPFS). Cualquier intento de manipulación de la imagen almacenada en IPFS resultaría en un CID diferente, rompiendo explícitamente la cadena de custodia digital y haciendo que la alteración sea detectable de forma inmediata y algorítmica. La combinación de blockchain e IPFS no solo sigue los principios de descentralización (van Steen & Tanenbaum, 2017), sino que refuerza activamente los objetivos de inmutabilidad verificable y transparencia del sistema.

Criptografía aplicada

La criptografía proporciona los pilares matemáticos que garantizan la seguridad, integridad y autenticidad en todo el ecosistema del prototipo (Katz & Lindell, 2020).

- **Funciones Hash Criptográficas:** Son algoritmos que transforman datos en una huella digital de tamaño fijo. Sus propiedades (unidireccionalidad, resistencia a colisiones, efecto avalancha) son vitales (Menezes y col., 1996; Schneier, 2007). En este proyecto, se utilizan para: generar el CID en IPFS, asegurar la integridad de la cadena de bloques y crear identificadores únicos para las transacciones (Benet, 2014; Nakamoto, 2008).
- **Criptografía Asimétrica y Firmas Digitales:** Basada en pares de claves (pública y privada), habilita las firmas digitales (Diffie & Hellman, 2022; Rivest y col., 1978). Cuando un usuario autorizado registra un comparendo, utiliza su clave privada para firmar la transacción. Cualquier participante puede usar la clave pública correspondiente para verificar la firma, garantizando así la **autenticidad** y el **no repudio** de la acción (Katz & Lindell, 2020).

A partir de estos fundamentos, se analizan a continuación las principales investigaciones y proyectos aplicados que exploran estas tecnologías en contextos gubernamentales.

Estado del arte

Blockchain en gestión gubernamental

La aplicación de la tecnología blockchain y DLT (Distributed Ledger Technology) en la administración pública ha sido un área de creciente interés, impulsada por las promesas teóricas de Inmutabilidad, Transparencia y Auditoría mejorada, fundamentales para la Confianza Descentralizada. La investigación sugiere que blockchain puede transformar la gestión de registros oficiales, como licencias, títulos de propiedad y, potencialmente, multas o sanciones como los fotocomparendos.

Análisis de aplicación. La capacidad de crear un registro de Transacciones criptográficamente asegurado y distribuido permite generar una pista de auditoría fiable y resistente a la manipulación. Cada registro de sanción, incluyendo sus Metadatos (fecha, hora, ubicación, tipo de infracción) y el Hash de la evidencia asociada, puede ser anclado a la cadena, proporcionando una fuente única de verdad verificable por las partes autorizadas, lo que potencialmente reduce disputas sobre la validez de registros. Esto se alinea con los principios de Sistemas Distribuidos aplicados a la gobernanza.

Blockchains públicas vs. permisionadas. En el contexto gubernamental, la literatura y los estudios piloto tienden a favorecer las blockchains permisionadas (o de consorcio). Si bien las blockchains públicas ofrecen máxima transparencia, las permisionadas permiten a las entidades gubernamentales controlar quién puede participar en la red (validar transacciones, acceder a datos), gestionar mejor la privacidad (crucial para datos ciudadanos) y, a menudo, ofrecer mayor rendimiento y escalabilidad. La elección impacta directamente en el modelo de Confianza Descentralizada implementado.

Madurez y barreras. Aunque existen numerosos estudios y proyectos piloto (ej., registros de tierras en Suecia o Georgia, identidad digital en Estonia), las implementaciones a gran escala para la gestión integral de sanciones administrativas aún son limitadas. La madurez es variable. Las barreras reconocidas incluyen la complejidad técnica, la necesidad de marcos legales y regulatorios adaptados, la interoperabilidad con sistemas heredados, los costos iniciales de implementación y la adopción tanto por parte de las instituciones como de los ciudadanos. La percepción pública de la tecnología blockchain también juega un rol significativo.

Integración blockchain-IPFS

El almacenamiento directo de datos voluminosos (como imágenes o vídeos de alta resolución) en una blockchain es ineficiente y costoso. La literatura técnica y diversos prototipos exploran la integración de blockchain con sistemas de Almacenamiento Direccionable por Contenido como IPFS (InterPlanetary File System) para abordar este desafío.

Estado actual. El enfoque predominante consiste en almacenar el dato voluminoso (la imagen del fotocomparendo) en IPFS, obteniendo un Hash único basado en su contenido. Este Hash IPFS, junto con otros Metadatos relevantes, se almacena en una Transacción blockchain. Este modelo aprovecha la eficiencia de IPFS para el almacenamiento distribuido y la fortaleza de blockchain para el registro inmutable y verificable del puntero (el Hash) y los metadatos asociados.

Ventajas y desafíos. Las ventajas logradas incluyen la verificabilidad (cualquier cambio en el archivo IPFS cambiaría su hash, invalidando el enlace en la blockchain), la resiliencia potencial (si múltiples nodos almacenan el archivo) y el direccionamiento por contenido inherente a IPFS. Sin embargo, persisten desafíos persistentes significativos:

Persistencia de datos (pinning). Los datos en IPFS solo persisten mientras algún nodo los esté "pineando"(almacenando activamente). Garantizar la persistencia a largo plazo de la evidencia requiere mecanismos o servicios de pinning fiables, que pueden tener costos asociados.

Disponibilidad. La recuperación del archivo depende de que los nodos que lo almacenan estén en línea y accesibles.

Costos a largo plazo. El almacenamiento distribuido no es necesariamente gratuito, especialmente si se requieren garantías de disponibilidad y persistencia.

Gestión de la privacidad. Los datos en IPFS son típicamente accesibles públicamente si se conoce el hash. Para evidencia sensible, se requerirían capas adicionales de encriptación antes de la subida a IPFS, añadiendo complejidad.

Cadena de custodia digital

La integridad y la cadena de custodia de la evidencia digital son cruciales en procesos sancionatorios. Blockchain/DLT ofrece mecanismos basados en Criptografía Aplicada para fortalecer estos aspectos.

Fortalecimiento de la integridad y trazabilidad. Al registrar el Hash de la evidencia digital (imagen del fotocomparendo) en una transacción blockchain, se crea un sello de tiempo (timestamping) inmutable y verificable. Cualquier intento posterior de modificar la evidencia original resultaría en un hash diferente, lo que permitiría detectar fácilmente la manipulación. La secuencia de transacciones en la blockchain proporciona una trazabilidad auditable del ciclo de vida de la evidencia (captura, registro).

Comparación y valor añadido. En comparación con los sistemas tradicionales (bases de datos centralizadas, logs de servidor), que pueden ser susceptibles a alteraciones internas o fallos únicos, la DLT aporta un valor añadido significativo al distribuir la confianza y hacer que la manipulación sea computacionalmente inviable (principio de Inmutabilidad). Esto refuerza la Confianza Descentralizada en la validez de la evidencia presentada, reduciendo potenciales disputas, un factor crítico en contextos donde la percepción de manipulación de evidencia genera altas tasas de impugnación ciudadana.

Estándares emergentes. En el ámbito de la tecnología blockchain, observamos la consolidación de estándares emergentes en diversas áreas, que representan un consenso práctico y técnico en ausencia de normas formales universalmente ratificadas.

Un área clave es la Seguridad de Smart Contracts. Para construir confianza y fiabilidad en las aplicaciones descentralizadas (dApps) y mitigar vulnerabilidades, se están adoptando ampliamente prácticas que funcionan como estándares de facto:

- **Auditorías y Listas de Chequeo:** Metodologías promovidas por firmas especializadas como ConsenSys Diligence, Trail of Bits y OpenZeppelin se han vuelto habituales.
- **Patrones de Diseño Seguro:** Se aplican convenciones como Checks-Effects-Interactions y el uso de proxies actualizables (UUPS, Transparent Proxy), aunque estos patrones continúan evolucionando.
- **Estándares de Reporte de Vulnerabilidades:** Propuestas como las EIPs (Ethereum Improvement Proposals) relacionadas con la seguridad ayudan a estandarizar la comunicación de fallos.

Otra área fundamental donde emergen estándares es la Gestión de Evidencia Digital y Cadena de Custodia mediante blockchain/DLT. Aunque todavía no existe una norma global única (como un estándar ISO específico para esta aplicación), sí se está formando un fuerte consenso técnico sobre los principios tecnológicos clave para asegurar la integridad y fiabilidad:

- **Hashing Criptográfico:** El uso de funciones hash para generar una huella digital única e infalsificable de la evidencia (como el CID en IPFS) es la práctica estándar para garantizar la integridad y detectar manipulaciones (Benet, 2014).
- **Timestamping Inmutable:** Registrar el hash de la evidencia y sus metadatos en una transacción blockchain proporciona una marca de tiempo segura e inalterable, estableciendo una prueba fehaciente del momento del registro (Nakamoto, 2008).
- **Registro en Ledger Distribuido (DLT):** Utilizar la DLT como el libro contable distribuido para estos registros es el mecanismo reconocido para lograr inmutabilidad, transparencia controlada y auditabilidad (Swan, 2015), superando las limitaciones de las bases de datos centralizadas.

Casos de implementación gubernamental

Registro de propiedad en Suecia

La autoridad catastral sueca *Lantmäteriet* realizó entre 2016 y 2017 un piloto con una cadena permisionada y contratos inteligentes para registrar transacciones inmobiliarias. El proyecto buscó reducir

la manipulación documental y agilizar los trámites que involucran a bancos, agentes inmobiliarios y entidades estatales. Los resultados mostraron que el tiempo de compraventa podría reducirse de 4–7 meses a tan solo unos días, con un ahorro estimado de ~100 millones de euros anuales gracias a la eliminación de procesos en papel y la automatización parcial (Haaramo, 2017; Suberg, 2017; Young, 2017).
Durante la segunda fase se incorporaron contratos inteligentes que ejecutaban automáticamente pasos como la firma digital de documentos y el registro de hipotecas cuando se cumplían condiciones predefinidas, demostrando la viabilidad técnica y la interoperabilidad entre actores.

Integridad de registros clínicos en Estonia

Desde 2016 la autoridad nacional de salud de Estonia, en colaboración con Guardtime, emplea la infraestructura KSI (*Keyless Signature Infrastructure*) para asegurar la integridad de los expedientes médicos de más de un millón de ciudadanos. En lugar de almacenar datos sensibles en la cadena, el sistema registra huellas *hash* de cada operación sobre la historia clínica, posibilitando auditorías en tiempo real y la detección inmediata de accesos o modificaciones no autorizadas (Guardtime & e-Health Authority, 2016; Reddit user, 2021). Esta aproximación cumple los requisitos de privacidad y refuerza la confianza pública en el manejo de datos sanitarios.

Síntesis y relevancia para el proyecto

Estos casos evidencian que:

- La tecnología blockchain coordina procesos complejos con múltiples partes interesadas, garantizando un registro único y verificable, con ahorros documentados de ~100 millones de euros anuales en el caso sueco.
- Permite verificar de forma irrefutable la integridad de datos sensibles sin exponer su contenido, manteniendo el cumplimiento normativo (ejemplo de Estonia).

Las lecciones aprendidas refuerzan la pertinencia de aplicar una arquitectura basada en blockchain e IPFS para gestionar evidencias de fotocomparendos en Bogotá, buscando niveles de transparencia, inmutabilidad y confianza comparables.

Aplicaciones en tránsito

Al revisar las aplicaciones de blockchain en la gestión de tráfico, infracciones y fotocomparendos, como se observa en la Tabla 4, se observa que, si bien hay discusiones teóricas (Yousfi y col., 2022), propuestas conceptuales (Chen y col., 2024) y hasta una joven PYME española, las implementaciones prácticas que integren el flujo completo descrito en el prototipo (captura -> IPFS -> Blockchain -> Verificación -> Pago

Automatizado con Billetera Digital) son escasas y se encuentran en fase de propuesta o son parciales (Choquevilca Quispe & Morales Valencia, 2024; Omar y col., 2024).

Tabla 4. *Análisis Comparativo del Estado del Arte en Gestión de Infracciones con blockchain*

Trabajo	**Ámbito**	**Tecnologías**	**Limitaciones Identificadas**	**Aporte Relevante para el Prototipo**
Yousfi et al. (2022)	Gestión de tráfico urbano	*Blockchain* pública, *Smart Contracts*	Alto costo de transacciones (*gas*), privacidad limitada para datos personales	Modelo conceptual de integración *blockchain*-tráfico, solución a la transparencia y trazabilidad
Chen et al. (2024)	Sistema de multas electrónicas	Base de datos centralizada + *Blockchain*	Falta de inmutabilidad completa, dependencia del servidor central	Propuesta de registrar *hash* de actas en *blockchain* para mayor integridad y transparencia
Joseph (2023)	Registros vehiculares gubernamentales	*Hyperledger Fabric*, *IPFS*	Complejidad para escalar, gestión de identidades	Arquitectura permisionada para manejo seguro de datos sensibles

Continúa en la página siguiente...

Tabla 4. *(Continuación)*

Trabajo	**Ámbito**	**Tecnologías**	**Limitaciones Identificadas**	**Aporte Relevante para el Prototipo**
Dutta et al. (2023)	Seguros automotrices	*Ethereum*, *Smart Contracts*	Latencia en transacciones, costos operativos	Automatización de procesos mediante contratos inteligentes
Omar et al. (2024)	Gestión de infracciones de tránsito	*Blockchain* híbrida, base de datos	Integración parcial, falta de flujo completo	Aproximación hacia una gestión descentralizada con uso mixto de tecnologías
Choquevilca Quispe & Morales Valencia (2024)	Fotocomparendos en Latinoamérica	Análisis conceptual	Falta de implementaciones prácticas en la región	Identificación de brechas y oportunidades para implementación *blockchain*
Proyectos e-gov (Suecia, Estonia)	Registros gubernamentales	*Blockchain* permisionada, *KSI*	Limitado a registros específicos, no multas	Validación técnica y mejora en eficiencia para registros oficiales

Continúa en la página siguiente...

Tabla 4. *(Continuación)*

Trabajo	**Ámbito**	**Tecnologías**	**Limitaciones Identificadas**	**Aporte Relevante para el Prototipo**
Anand & Singh (2024)	Gestión de documentos oficiales	*IPFS* + *Blockchain*	Persistencia en *IPFS*, costos de almacenamiento	Almacenamiento distribuido para evidencias con verificación en *blockchain*
JUIT Research Group (2024)	Sistema de pagos gubernamentales	*Stablecoins*, *Smart Contracts*	Adopción de criptomonedas, entorno regulatorio restringido	Automatización de pagos en ecosistemas *blockchain*

Nota. Elaboración propia.

Análisis de existencia. La literatura existente se centra más en componentes aislados (Yousfi y col., 2022), uso de blockchain para registros vehiculares (Mani Joseph, 2023), seguros (Dutta y col., 2023), o gestión genérica de multas (Omar y col., 2024), pero raramente combinando el almacenamiento de evidencia en IPFS con la automatización del pago vía billetera digital específicamente para fotocomparendos.

Arquitecturas y resultados. Dada la escasez de implementaciones completas reportadas (Anand & Singh, 2024; JUIT Research Group, 2024), es difícil generalizar sobre arquitecturas dominantes o resultados concretos para este caso de uso tan específico. Los estudios existentes a menudo se limitan a explorar la viabilidad teórica o a implementar módulos parciales (Choquevilca Quispe & Morales Valencia, 2024).

Lecciones aprendidas. La principal lección aprendida de áreas adyacentes es la importancia de abordar no solo los desafíos técnicos (Zheng y col., 2018) sino también los regulatorios, de gobernanza y de adopción (Tan y col., 2022). La ausencia de soluciones integrales reportadas para el flujo completo de gestión de fotocomparendos representa una brecha significativa, particularmente en contextos donde la ciudadanía carece de mecanismos para verificar independientemente la validez de las sanciones impuestas.

El análisis del estado del arte revela avances significativos, pero también limitaciones claras:

Avances significativos

La tecnología blockchain ha demostrado su potencial para crear registros gubernamentales más inmutables, transparentes y auditables. (Balcerzak y col., 2022; Meroni y col., 2023)

La integración blockchain + IPFS es una solución técnicamente viable y reconocida para gestionar datos voluminosos referenciados desde una cadena de bloques, mejorando la verificabilidad (Adel y col., 2023; Mishra y col., 2024).

DLT ofrece mejoras sustanciales para la integridad y trazabilidad de la evidencia digital (Thanasas y col., 2025).

Existen mecanismos (billeteras digitales, smart contracts, stablecoins) para habilitar pagos digitales automatizados en ecosistemas blockchain (Antonopoulos & Harding, 2023).

Limitaciones identificadas

Madurez e integración. Muchas aplicaciones gubernamentales de blockchain son pilotos aislados (Li y col., 2021; Zheng y col., 2018). Falta integración entre sistemas y con procesos completos.

Desafíos técnicos. La persistencia y gestión a largo plazo de datos en IPFS (pinning), la escalabilidad de algunas blockchains y la seguridad/fiabilidad de los oráculos para Smart Contracts siguen siendo áreas de desarrollo activo (Zheng y col., 2018).

Adopción y regulación. La adopción de billeteras digitales para pagos gubernamentales, el uso de criptoactivos/stablecoins y la claridad legal sobre smart contracts en el sector público son obstáculos importantes (Tan y col., 2022).

Política de reserva de información. Bogotá mantiene una política de reserva de información que restringe la divulgación completa de los datos almacenados en su base de datos. Esta política limita el acceso y la difusión de ciertos datos, lo que puede afectar la transparencia y la capacidad de realizar un análisis exhaustivo (Choquevilca Quispe & Morales Valencia, 2024).

Actualización de datos. La actualización de los datos almacenados en la base de datos es un proceso que requiere tiempo. Dada la naturaleza progresiva de este proceso, que depende de la cantidad de datos que se agregan diariamente, la actualización completa de los datos con el sistema que se desarrollará puede llevar un tiempo considerable (Choquevilca Quispe & Morales Valencia, 2024).

Aplicación específica. Existe una notable ausencia de soluciones documentadas que implementen el flujo completo e integrado (captura de imagen -> subida a IPFS -> registro en blockchain -> verificación vía app -> pago automático desde billetera digital) específicamente para la gestión de fotocomparendos. (Chen y col., 2024; Yousfi y col., 2022)

Verificación participativa. Los sistemas actuales raramente permiten que el ciudadano verifique independientemente la autenticidad e integridad de la evidencia presentada contra ellos, limitando los beneficios de transparencia inherentes a blockchain. Esta opacidad genera un vacío de confianza que, en algunos contextos, ha sido explotado por terceros mediante esquemas fraudulentos.

Adopción en Bogotá. La literatura muestra una escasez notable de implementaciones o estudios piloto en contextos latinoamericanos, donde factores como confianza institucional, infraestructura tecnológica y marcos regulatorios presentan desafíos particulares. (Choquevilca Quispe & Morales Valencia, 2024; Rezabala Loor & Alci'var Cevallos, 2025)

Relevancia del prototipo

Las brechas específicas que este prototipo busca abordar es precisamente la falta de una solución integrada y de extremo a extremo para la gestión de fotocomparendos utilizando la sinergia de blockchain, IPFS y pagos automatizados, como se evidencia en la revisión de la literatura existente (Anand & Singh, 2024; Yousfi y col., 2022) La novedad principal radica en la integración holística de todo el flujo propuesto. Mientras que los componentes individuales han sido explorados por separado (Adel y col., 2023; Mishra y col., 2024) o en otros contextos (Dutta

y col., 2023; Mani Joseph, 2023), este prototipo propone conectarlos en una secuencia lógica y automatizada para este caso de uso particular.

Aborda la brecha de aplicación específica, llevando los conceptos teóricos (Antonopoulos & Harding, 2023; Swan, 2015) y las soluciones parciales existentes (Choquevilca Quispe & Morales Valencia, 2024) a un dominio concreto (fotocomparendos en Bogotá) con un proceso completo.

La relevancia del prototipo se justifica por su potencial para:

- Mejorar la transparencia y confianza en el proceso de fotocomparendos, evidencia verificable e inmutable (Meroni y col., 2023; Thanasas y col., 2025).
- Aumentar la eficiencia operativa mediante la automatización del registro, verificación y pago.
- Reducir disputas y costos asociados a la gestión manual y a la falta de confianza en la evidencia, factores que en sistemas centralizados generan cargas operativas significativas.
- Explorar un modelo innovador de pago automatizado condicional basado en la verificación en blockchain.

Tendencias internacionales

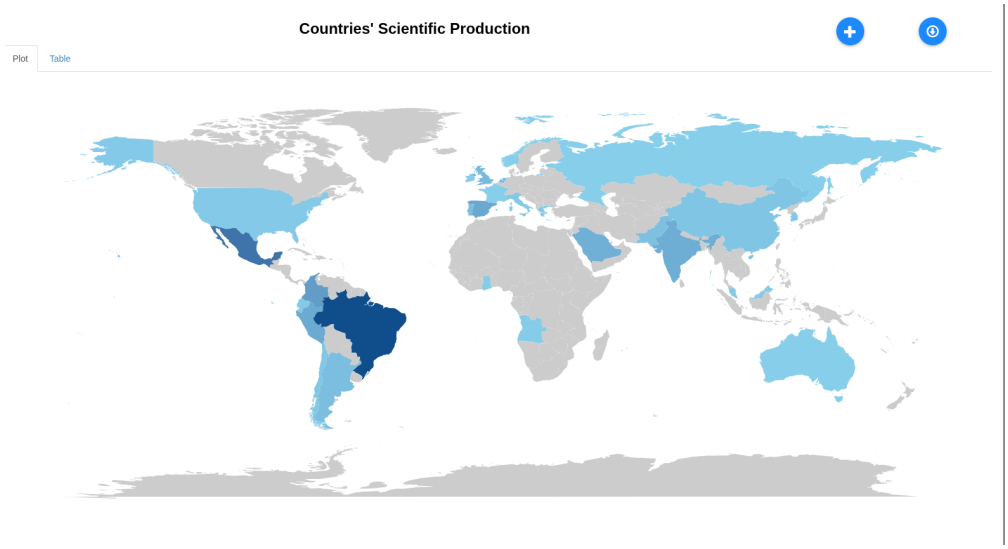
Producción científica por países (mapa y gráfico de líneas). Descripción General: Esta gráfica se compone de dos partes. La primera es un mapa mundial que utiliza una escala de color para representar la cantidad de producción científica por país. Las tonalidades más oscuras generalmente indican una mayor producción. La segunda parte es un gráfico de líneas que muestra la evolución de la producción científica (en artículos) a lo largo de los años para un conjunto específico de países.

Mapa mundial. El mapa muestra la distribución global de la producción científica en el área de estudio. Se observa una concentración significativa de publicaciones en países como Brasil, lo que sugiere un interés y actividad investigadora importante en Latinoamérica. Otros países con una producción notable incluyen México y España. Es importante notar que algunas regiones

muestran una menor actividad, lo que podría indicar diferencias en el enfoque de investigación, financiamiento o acceso a recursos.

Figura 2

Distribución global de la producción científica sobre blockchain y gestión de infracciones



Nota. Mapa mundial y gráfico de líneas mostrando la producción científica sobre blockchain por países.

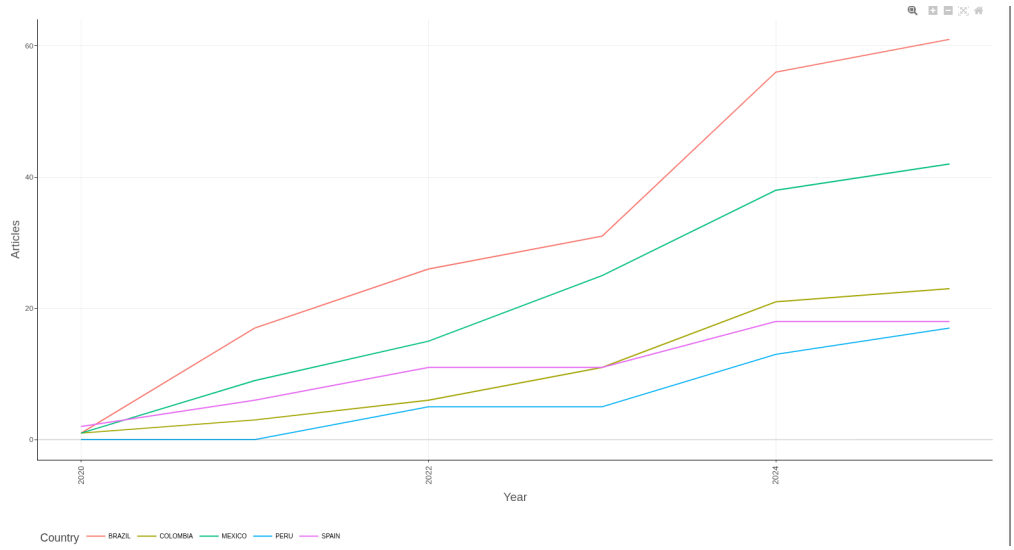
Nube de palabras. La Figura 4 muestra los términos más frecuentes en la literatura analizada. Destacan conceptos como *blockchain*, *challenges*, *management* y *secure*, lo que refleja el énfasis de la comunidad académica en los retos de seguridad y gestión al aplicar tecnologías DLT en contextos gubernamentales.

Mapa temático. El mapa temático representa la distribución de los principales temas de investigación en el área, organizados según su grado de desarrollo (densidad) y relevancia (centralidad). En el cuadrante superior derecho se ubican los "temas motores", como *challenges*, *framework*, *model*, que son altamente desarrollados y centrales en la literatura.

En el cuadrante inferior derecho, los "temas básicos" como *management*, *secure*, *network* e

Figura 3

Evolución anual de publicaciones en los países líderes del tema (Brasil, México, Colombia, España y Perú)



Nota. Evolución anual de publicaciones científicas sobre blockchain en los países líderes.

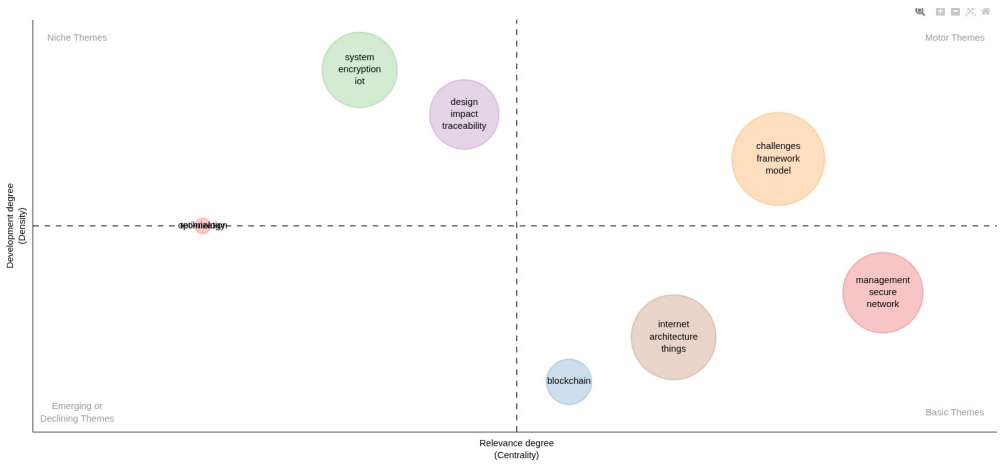
Figura 4

Nube de palabras de los términos más recurrentes en la literatura sobre blockchain e infracciones

Nota. Nube de palabras con los términos más recurrentes en la literatura sobre blockchain e infracciones.

Figura 5

Mapa temático de los principales temas de investigación en el área



Nota. Mapa temático de los principales temas de investigación en blockchain y gestión de infracciones.

internet, *architecture*, *things* son fundamentales pero menos desarrollados. El cuadrante superior izquierdo agrupa "temas nicho" como *system*, *encryption*, *IoT* y *design*, *impact*, *traceability*, que presentan alta especialización pero menor centralidad.

Finalmente, en el cuadrante inferior izquierdo se encuentran los "temas emergentes o en declive", como *blockchain* y *optimization*, *technology*, que muestran baja densidad y centralidad, indicando áreas de reciente aparición o menor desarrollo. Esta visualización permite identificar las tendencias, vacíos y oportunidades de investigación en el campo.

Sistema de fotocomparendo en Bogotá

Esta subsección contextualiza el problema abordado en este trabajo, describiendo el funcionamiento operativo y regulatorio del sistema de fotocomparendos en Bogotá, así como las limitaciones críticas identificadas en el sistema FÉNIX que justifican la necesidad de explorar arquitecturas alternativas basadas en tecnologías de registro distribuido.

Funcionamiento de los fotocomparendos en Bogotá: mecanismos, regulación e impacto

El sistema de fotocomparendos en Bogotá (FENIX) representa un modelo tecnológico y regulatorio diseñado para mejorar la seguridad vial mediante la detección automatizada de infracciones de tránsito (Ministerio de Transporte de Colombia, 2023). Basado en cámaras de fotodetección ubicadas en zonas autorizadas por el Ministerio de Transporte, este sistema combina vigilancia electrónica, validación humana y marcos legales específicos para sancionar conductas de riesgo (Superintendencia de Transporte, 2021). Desde su implementación, ha logrado reducir siniestros en puntos críticos, aunque enfrenta desafíos técnicos y jurídicos. A continuación, se detalla su operación, criterios de aplicación y marco legal que lo regula (Ministerio de Transporte de Colombia, 2023).

Marco legal y regulatorio. La implementación de fotocomparendos en Bogotá se sustenta en la Ley 1843 de 2017 (Congreso de Colombia, 2017) y su reglamentación mediante resoluciones como la Resolución 20203040011245 (Ministerio de Transporte de Colombia, 2020). Estos instrumentos establecen cuatro criterios para instalar cámaras:

1. **Siniestralidad:** Ubicación en zonas con alto índice de accidentes.

2. **Prevención:** Disuasión de conductas peligrosas.
3. **Movilidad:** Optimización del flujo vehicular.
4. **Historial de infracciones:** Enfoque en corredores con recurrentes violaciones.

Adicionalmente, las autoridades deben garantizar la visibilidad de los dispositivos, señalizando su presencia al menos 500 metros antes de su ubicación (Congreso de Colombia, 2017), y cumplir con planes de seguridad vial alineados con políticas distritales. La Secretaría Distrital de Movilidad (SDM) ha enfrentado cuestionamientos legales, como los señalados por la Personería en 2018 (Secretaría Distrital de Movilidad, 2023a).

Proceso operativo de los fotocomparendos.

Detección y captura de infracciones.

Las cámaras de fotodetección en Bogotá se clasifican en dos tipos (Ministerio de Transporte de Colombia, 2023; Superintendencia de Transporte, 2021):

- **Automáticas:** Monitorean velocidades, semáforos en rojo y restricciones como pico y placa.
- **Semiautomáticas:** Vigilan bloqueos de calzadas, paradas prohibidas y recolección irregular de pasajeros.

Estos dispositivos, instalados en corredores de alta accidentalidad como la Avenida NQS o la Calle 100, capturan imágenes o videos que incluyen matrícula, fecha, hora y ubicación GPS. Por ejemplo, en 2025, un conductor que exceda el límite de 50 km/h en la Avenida Boyacá será registrado por cámaras previamente señalizadas.

Validación y notificación.

Una vez detectada una presunta infracción, las pruebas se envían a un centro de análisis de la SDM, donde agentes de tránsito verifican:

- Legibilidad de la matrícula.
- Contexto de la violación (ejemplo: si un semáforo en rojo fue respetado).
- Datos del vehículo en el RUNT (Registro Único Nacional de Tránsito).

Tras validar la infracción, se genera un comparendo electrónico notificado al propietario del vehículo mediante correo certificado o plataformas digitales. El plazo máximo para emitir la sanción es de 10 días hábiles desde la detección, seguido de 3 días para su notificación. Si el domicilio registrado está desactualizado, el infractor podría no recibir la notificación, lo que no exime el pago (Congreso de Colombia, 2017).

Tecnología y transparencia.

El sistema combina:

- **Cámaras de última generación:** Equipadas con sensores de velocidad Lidar y visión nocturna.
- **Plataforma de análisis IA:** Algoritmos que descartan falsos positivos (ejemplo: ambulancias en emergencia).
- **Integración con RUNT:** Verificación instantánea de documentos como SOAT o tarjeta de operación.

Los datos se almacenan en servidores con cifrado AES-256, accesibles solo para funcionarios autorizados mediante autenticación biométrica.

Problemas operativos identificados.

- **Notificaciones fallidas:** Errores en direcciones del RUNT causan sanciones no recibidas, acumulando intereses moratorios.
- **Latencia en validaciones:** En horas pico, el volumen de infracciones puede retrasar procesamientos hasta 72 horas.

Cuestionamientos legales.

En 2018, la Personería de Bogotá identificó que el 15 % de las cámaras operaban sin autorización ministerial durante un periodo de transición legal. La SDM rectificó esta situación en 2019, regularizando todos los dispositivos bajo la Resolución 20203040011245 de 2020 (Secretaría Distrital de Movilidad, 2023b).

Análisis crítico del sistema FÉNIX: limitaciones estructurales identificadas

La necesidad de explorar arquitecturas alternativas se sustenta en las deficiencias críticas identificadas en el sistema FÉNIX, el cual, a pesar de su implementación tecnológica, presenta limitaciones documentadas por los organismos de control en tres dimensiones: cumplimiento normativo, protección de datos y capacidad técnica.

Incumplimientos normativos en contratación. El proceso de contratación del sistema FÉNIX presenta desviaciones respecto a la normativa de contratación pública colombiana. La Tabla 5 sintetiza las principales violaciones identificadas por la Contraloría de Bogotá, evidenciando falencias en la planeación, supervisión y gestión contractual del proyecto.

Tabla 5. *Violaciones al Estatuto de Contratación Pública (Ley 80/1993)*

Artículo	**Disposición**	**Hallazgo en FÉNIX**	**Fuente**
3	Principios de transparencia y economía	Adendas sin justificación técnica ni económica	Auditoría 90-2023, p. 18
40	Cumplimiento contractual	Modificaciones de alcance sin cláusula de variación	Contraloría 170100-0054-24

Nota. Elaboración propia.

Brechas en protección de datos personales. La gestión de información sensible de los ciudadanos presenta deficiencias en el cumplimiento de la Ley 1581 de 2012 (Protección de Datos Personales). La Tabla 6 detalla las brechas identificadas en materia de privacidad y tratamiento de datos.

Tabla 6. *Brechas en protección de datos personales*

Norma	**Exigencia**	**Brecha detectada**	**Riesgo**
Ley 1581/2012	Registro ante la SIC	Base de datos no registrada hasta 2022	Sanción hasta 2.000 SMMLV

Nota. Elaboración propia.

Déficits en habilitación y capacidad técnica. La capacidad operativa y técnica del sistema muestra limitaciones documentadas en auditorías de cumplimiento. La Tabla 7 presenta los principales déficits en términos de infraestructura, recursos y capacidad institucional.

Tabla 7. *Deficits de habilitación sectorial*

Requisito	**Fuente**	**% incumplimiento**	**Consecuencia**
Certificado técnico por cámara	Res. 11245/2020	15 %	Comparendos nulos

Nota. Elaboración propia.

Síntesis de la problemática. Estos hallazgos, documentados por la Contraloría de Bogotá y otras entidades de control, evidencian que las limitaciones del sistema FÉNIX no son solo operativas sino estructurales, requiriendo un replanteamiento arquitectónico que incorpore garantías criptográficas y descentralización como fundamentos de diseño. El presunto detrimento patrimonial de más de $8.000 millones de pesos y la tasa de impugnación del 34.1 % (más de 155.000 PQRSD semestrales) reflejan la magnitud del problema y justifican la exploración de arquitecturas basadas en tecnologías de registro distribuido.

Esta problemática contextualiza la necesidad del prototipo propuesto en este trabajo, el cual busca demostrar cómo una arquitectura descentralizada puede mitigar estos riesgos mediante

garantías técnicas intrínsecas de inmutabilidad, trazabilidad y verificabilidad.

Metodología

Este proyecto combina investigación aplicada con desarrollo tecnológico innovador. A continuación se describen el enfoque metodológico, la selección tecnológica y el modelo de desarrollo utilizado.

Enfoque metodológico de investigación

Este trabajo se enmarca en una investigación aplicada que aborda las deficiencias de integridad, transparencia y confianza en el sistema actual de gestión de fotocomparendos en Bogotá. Adopta un enfoque descriptivo al detallar las características de un sistema descentralizado basado en blockchain e IPFS, constituyendo un caso de estudio sobre su aplicación en el sector público.

Selección de la pila tecnológica

La selección de tecnologías de registro distribuido (DLT) fue crítica, impactando directamente en:

- Privacidad de datos personales (Ley 1581 de 2012).
- Escalabilidad ante ∼457.000 comparendos semestrales.
- Costos operativos predecibles (sin criptomonedas volátiles).
- Modelo de gobernanza institucional.

Justificación del uso de blockchain

Para el caso de fotocomparendos, donde la **integridad irrefutable** y la **verificación ciudadana independiente** son requisitos no negociables, blockchain resulta la tecnología más apropiada frente a otras alternativas de registro distribuido.
La Tabla 8 compara blockchain con otras tecnologías emergentes.

Tabla 8. *Comparación de blockchain con otras tecnologías de registro distribuido*

Criterio	**Blockchain**	**Hashgraph**	**BD Distribuida**	**BD Centralizada**
Inmutabilidad	Alta (criptográfica)	Media (consenso virtual)	Baja (config.)	Ninguna (admin.)
Resistencia manipulación	Alta (prohibitiva)	Media	Baja (permisos)	Ninguna (admin)
Auditabilidad	Alta (completa)	Media (parcial)	Baja (logs modificables)	Baja (logs centralizados)
Descentralización	Alta (real)	Alta (real)	Baja (réplicas)	Ninguna
Verificación indep.	Alta (sin confianza)	Media (requiere nodos)	Ninguna (acceso BD)	Ninguna (API controlada)
Estándares	Alta (maduros)	Baja (emergente)	Media (SQL/NoSQL)	Alta (SQL)
Rendimiento (TPS)	Media (15-20.000)	Alta (>10.000)	Alta (>100.000)	Alta (>100.000)
Costo operativo	Alto	Moderado	Moderado	Bajo
Precedente legal	Alto (eIDAS UE)	Bajo (sin precedente)	Medio (aceptado)	Alto (estándar)

Continúa en la siguiente página

Tabla 8. *(Continuación)*

Criterio	Blockchain	Hashgraph	BD Distribuida	BD Centralizada
Apto evidencia legal	**SÍ**	Parcial	NO	NO

Nota. Elaboración propia.

Justificación de la elección de blockchain. La selección de blockchain se fundamenta en los siguientes argumentos técnicos y legales:

1. **Inmutabilidad criptográfica verificable:** A diferencia de bases de datos donde los logs pueden ser alterados por administradores con privilegios elevados, blockchain garantiza que modificar un registro requeriría alterar toda la cadena desde ese punto, lo cual es computacionalmente prohibitivo (Nakamoto, 2008). Esta propiedad es crítica para evidencia que puede ser objeto de litigio.

2. **Verificación sin confianza (trustless):** Un ciudadano puede verificar la autenticidad de un fotocomparendo sin necesidad de confiar en la institución emisora, simplemente validando la cadena de hashes. Esto no es posible con bases de datos tradicionales donde la verificación depende de APIs controladas por la misma entidad (Antonopoulos & Harding, 2023). Esta característica aborda directamente la crisis de confianza reflejada en la tasa de impugnación del 34.1 %.

3. **Precedente legal reconocido:** Existen marcos regulatorios emergentes que reconocen la validez legal de registros blockchain. El Reglamento eIDAS de la Unión Europea (European Parliament and Council, 2014) establece un marco para la identificación electrónica y servicios de confianza que incluye tecnologías de registro distribuido. Tecnologías más recientes como Hashgraph (Baird, 2016) aún no han establecido precedentes legales comparables.

4. **Auditabilidad completa e inmutable:** Cada transacción queda registrada con timestamp inmutable, creando una cadena de custodia digital irrefutable para procesos sancionatorios (Swan, 2015). Esta trazabilidad es esencial para cumplir con los requisitos de debido proceso administrativo.

5. **Madurez del ecosistema:** Blockchain cuenta con implementaciones probadas en producción (Hyperledger Fabric (Cachin, 2018), Ethereum (Wood, 2014)), herramientas de desarrollo consolidadas y comunidades activas. Si bien tecnologías como Hashgraph ofrecen mayor rendimiento teórico (Baird y col., 2020), o IOTA Tangle (Popov, 2018) promete eliminación de fees, ninguna ha demostrado la robustez operativa de blockchain en entornos gubernamentales críticos.

Estudios comparativos recientes (Karlsson y col., 2019; Ruan y col., 2021) confirman que, si bien bases de datos distribuidas como Cassandra tienen menor costo operativo y mayor rendimiento bruto, ninguna proporciona el nivel de **confianza descentralizada** y **resistencia a manipulación** que requiere un sistema de sanciones gubernamentales donde la percepción de imparcialidad es crítica.

Con esta fundamentación establecida, la siguiente decisión crítica es determinar qué implementación específica de blockchain utilizar y cómo estructurar la arquitectura del sistema.

Arquitectura híbrida: balance entre privacidad y transparencia. Dado que ninguna blockchain cumple simultáneamente con todos los requisitos (privacidad de datos sensibles + transparencia pública + rendimiento + costos controlados), se optó por una **arquitectura híbrida**:

- **Capa privada (permisionada):** gestión interna y datos sensibles.
- **Capa pública (blockchain):** verificación ciudadana sin intermediarios.

Capa privada: Hyperledger Fabric

Para la capa privada se seleccionó Hyperledger Fabric tras un análisis comparativo de plataformas blockchain.

Tabla 9. *Comparativo de plataformas blockchain para selección de arquitectura híbrida*

Criterio	**Hyperledger Fabric**	**Ethereum**	**Corda**	**Solana**	**Polygon**
Tipo de red	Permisionada	Pública	Permisionada	Pública	Pública
Consenso	Raft / BFT	PoS	Notario	PoH + PoS	PoS
TPS	2.000–20.000	∼30	∼1.000	65.000+	7.000+
Privacidad	Alta[(1)]	Nula	Alta (P2P)	Nula	Nula
Smart contracts	Go, Java, Node.js	Solidity	Kotlin/Java	Rust/C	Solidity
Control de acceso	PKI / Roles	Abierto	Identidad	Abierto	Abierto
Moneda nativa	No	ETH	No	SOL	MATIC
Costo / tx	Sin gas	Gas variable	Sin gas	Muy bajo	Muy bajo
Madurez Gob.	Alta	Media	Alta (banca)	Baja (DeFi)	Media

Razones de elección de Hyperledger Fabric:

- **Privacidad y confidencialidad:** canales y colecciones privadas permiten segmentar la información, garantizando que solo entidades autorizadas (agentes, auditores) accedan a datos sensibles, cumpliendo la Ley 1581 de 2012.
- **Rendimiento:** 2 000–20 000 TPS, suficiente para el volumen de Bogotá sin cuellos de botella.
- **Sin costos de gas:** elimina volatilidad y complejidad, crítico para presupuestos

gubernamentales.

- **Control de acceso granular:** PKI + roles definidos internamente (admin, agente, auditor, ciudadano).

Descarte de alternativas:

- **Ethereum / Solana / Polygon:** públicas ⇒ exposición total de datos y costos variables.
- **Corda:** orientada a finanzas; menor flexibilidad para evidencias fotográficas heterogéneas.

Capa pública: Ethereum

Para la verificación ciudadana se eligió Ethereum (testnet Sepolia) por:

- **Máxima transparencia:** cualquier persona puede verificar metadatos sin permisos.
- **Ecosistema maduro:** mayor comunidad, herramientas (Ethers.js, Hardhat) y estándares (ERC-20, ERC-721).
- **Costo controlado:** solo se publican hashes y metadatos no sensibles, minimizando gastos de gas.

Metodología de desarrollo

Para la construcción del sistema, se seleccionó el **Modelo de Desarrollo por Prototipos** (*Prototyping Model*). Esta elección metodológica fue estratégica y se fundamenta en las características inherentes al proyecto.

Justificación de la elección

La adopción de este modelo iterativo responde a tres factores cruciales:

1. **Naturaleza Innovadora y Riesgo Tecnológico:** El proyecto combina tecnologías emergentes como blockchain (Hyperledger Fabric y Ethereum) e IPFS en un dominio gubernamental donde no existían precedentes locales de una integración similar. La alta incertidumbre sobre el rendimiento, la seguridad de los contratos inteligentes y la viabilidad de la sincronización entre redes heterogéneas requería una validación temprana para mitigar riesgos técnicos fundamentales.

2. **Requisitos Evolutivos:** Los requisitos funcionales y no funcionales de un sistema de esta naturaleza están sujetos a cambios, tanto por la evolución de la tecnología como por posibles ajustes en el marco normativo de las sanciones de tránsito. El enfoque por prototipos ofrece la flexibilidad necesaria para adaptar la solución de forma ágil a medida que se profundiza el entendimiento del problema.

3. **Validación Temprana de Conceptos:** Era imperativo demostrar la hipótesis central del proyecto —que la combinación de blockchain e IPFS puede garantizar la inmutabilidad y verificabilidad de la evidencia digital— antes de invertir recursos en el desarrollo de una plataforma completa. El prototipo sirvió como una prueba de concepto funcional para validar esta premisa.

Fases del proceso de desarrollo

El ciclo de vida del desarrollo siguió las fases iterativas del modelo de prototipos, adaptadas a los objetivos específicos del proyecto, como se describe en la Tabla 10.

Tabla 10. *Fases del proceso de desarrollo del prototipo*

Fase	**Descripción**	**Aplicación en el Proyecto**
1. Requisitos Iniciales	Recopilación de los requisitos funcionales básicos y esenciales del sistema.	Se definieron las funcionalidades mínimas viables: registro inmutable de multas, almacenamiento de evidencia en IPFS, consulta pública y un mecanismo para la verificación de integridad.

Continúa en la siguiente página

Tabla 10. *(Continuación)*

Fase	Descripción	Aplicación en el Proyecto
2. Construcción del Prototipo	Desarrollo rápido de una versión funcional reducida que implementa los requisitos iniciales.	Se implementó un prototipo funcional que incluía un Smart Contract en una red local de Ethereum, una API REST para la comunicación y un frontend básico para la interacción del usuario.
3. Evaluación del Prototipo	Validación del prototipo mediante pruebas internas para evaluar su funcionalidad y alineación con los objetivos.	Se ejecutó un plan de pruebas exhaustivo (detallado en la sección Plan de pruebas) para validar la inmutabilidad de los registros, la integridad de la evidencia y la usabilidad de la interfaz con datos simulados.
4. Refinamiento e Iteración	Ajuste y mejora del prototipo basándose en los hallazgos de la evaluación.	Con base en los resultados, se optimizó el consumo de gas del Smart Contract, se mejoraron las validaciones de la API y se refinó la arquitectura para incorporar la capa privada con Hyperledger Fabric.

Continúa en la siguiente página

Tabla 10. *(Continuación)*

Fase	Descripción	Aplicación en el Proyecto
5. Documentación Final	Una vez validado el concepto, se documenta la arquitectura final y se proponen los siguientes pasos.	Se consolidó el diseño de la arquitectura híbrida final y se elaboró un *roadmap* detallado para una eventual implementación en un entorno de producción.

Ventajas y limitaciones del enfoque

La metodología por prototipos ofreció ventajas estratégicas determinantes para el éxito del proyecto, entre las que destacan la **validación temprana de la arquitectura híbrida**, la **mitigación de riesgos técnicos** relacionados con el rendimiento de IPFS y la **reducción de costos** al permitir ajustes antes de la fase final de desarrollo.

No obstante, es importante reconocer las limitaciones inherentes a este enfoque en el contexto de este trabajo:

- **Rendimiento no representativo:** El prototipo fue evaluado en un entorno de laboratorio controlado, por lo que su rendimiento no refleja las condiciones de una red pública con alta carga transaccional.
- **Gestión de expectativas:** Una versión funcional puede generar expectativas en los usuarios de que el sistema está casi terminado, cuando aún requiere fases críticas de seguridad y optimización.
- **Disciplina de desarrollo:** Se requirió una disciplina estricta para asegurar que el código del prototipo, concebido para validación, no se promoviera a un entorno de producción sin pasar por procesos formales de auditoría y refactorización.

En conclusión, la metodología por prototipos fue fundamental para navegar la complejidad e

incertidumbre del proyecto. Permitió demostrar de manera empírica que una arquitectura descentralizada es una solución técnica viable y socialmente pertinente para fortalecer la confianza en la gestión de fotocomparendos en Bogotá.

Artefactos técnicos del diseño

Con el fin de estructurar de manera clara el desarrollo de la solución propuesta, en esta sección se presentan los principales artefactos utilizados durante la etapa de diseño. Estos elementos permiten representar gráficamente tanto la lógica de funcionamiento como la arquitectura del sistema, sirviendo como guía para la implementación y posterior validación del prototipo.

El conjunto de diagramas que se incluye responde a la necesidad de modelar distintos aspectos del sistema. Por un lado, se usan diagramas de casos de uso para identificar las funcionalidades clave desde la perspectiva del usuario. Por otro, los diagramas de clases permiten definir la estructura del software, mientras que los diagramas de despliegue muestran cómo se distribuyen los componentes en el entorno tecnológico. Además, se incluyen diagramas de flujo que describen el comportamiento del sistema ante eventos específicos, facilitando la comprensión de su dinámica interna.

Cada uno de estos artefactos está alineado con los objetivos del proyecto y fue elaborado considerando tanto las necesidades funcionales como las características propias de las tecnologías involucradas, en particular el uso de Blockchain e IPFS. De esta forma, se busca garantizar coherencia técnica en el diseño y establecer una base sólida para el desarrollo e implementación de la solución.

Alcance

Delimitación geográfica

Este trabajo se circunscribe al proceso de generación, gestión y verificación de **multas de tránsito automatizadas (fotomultas)** emitidas por la Secretaría Distrital de Movilidad de Bogotá. Se excluyen deliberadamente:

- Multas impuestas de forma presencial por agentes de tránsito.
- Procesos sancionatorios de otras ciudades o entidades territoriales.
- Funcionalidades de recaudo y pasarelas de pago (solo se registra el estado del pago, no se procesa el pago en sí).

Componentes del prototipo

El prototipo aborda los siguientes módulos funcionales:

1. **Registro inmutable de la infracción** Captura de metadatos (placa, fecha, hora, ubicación y tipo de infracción) y publicación del identificador de la evidencia en la *blockchain* (*Ethereum* local con *Hardhat* para desarrollo, con arquitectura preparada para *Hyperledger Fabric* en producción).
2. **Almacenamiento descentralizado de evidencias** Carga de la imagen o video de la fotomulta en *IPFS* y obtención de su *hash*.
3. **Verificación pública** Servicio de consulta que permite contrastar el *hash* guardado en la cadena con el archivo almacenado en *IPFS*.
4. **Gestión del ciclo de vida de la multa** El prototipo implementa la gestión de transiciones entre estados del proceso de fotocomparendos. El modelo conceptual completo, descrito en la Formulación del problema (sección 1.1), contempla ocho estados: GENERADA → NOTIFICADA → PENDIENTE_RESPUESTA → EN_APELACION → RESUELTA_APELACION → PAGADA/CANCELADA → CERRADA. Como prueba de concepto, el *smart contract* implementa cinco de estos estados (ver Tabla 11 en sección 8

para mapeo detallado). Cada transición queda registrada mediante eventos de *smart contract* con *timestamp* inmutable.

5. **Interfaz mínima** Panel Web para: (i) agentes que registran la infracción y (ii) ciudadanos que consultan la autenticidad y el estado de su fotomulta.

Fuera del alcance

- Integración completa con sistemas legados del RUNT o SIMIT; se simula mediante datos de prueba.
- Implementación de un modelo económico (tarifas de gas, costos operativos reales).
- Implementación de algoritmos de detección automática de infracciones (visión por computador). Se parte de que la cámara ya detectó la infracción y generó la evidencia.

Entregables

- **Contrato inteligente en Solidity** con 80 pruebas automatizadas (97.5 % de éxito).
- Script de despliegue de red *Ethereum* local (*Hardhat*) e instalación de *IPFS* local.
- Aplicación Web de demostración (*frontend* ligero) conectada a los servicios anteriores.
- Manual técnico que documenta la arquitectura y el flujo de datos.
- Informe de resultados de las pruebas funcionales y de rendimiento básico.

Criterios de éxito

1. Tiempo medio de publicación de una infracción ≤ 3 s en entorno de laboratorio.
2. Coincidencia 100 % entre el *hash* almacenado en la cadena y la evidencia recuperada desde *IPFS*.
3. Trazabilidad completa del historial de estados para al menos 50 multas de prueba.
4. Ausencia de fallos críticos en pruebas de carga con 10 transacciones concurrentes.

Limitaciones del prototipo

Es fundamental reconocer que, como prototipo desarrollado en un contexto académico, el presente estudio presenta ciertas limitaciones que definen el alcance de sus conclusiones y delinean claras oportunidades para futuras investigaciones. Las principales limitaciones son:

1. **Entorno de validación**

   - **Validación en Entorno de Laboratorio:** El prototipo fue diseñado, desplegado y evaluado en un entorno de simulación controlado. No se sometió a pruebas en una infraestructura productiva real con la carga de transacciones y el volumen de usuarios que gestiona actualmente la Secretaría de Movilidad. Por lo tanto, su rendimiento, estabilidad y escalabilidad bajo condiciones de estrés real aún no han sido cuantificados.
   - **Uso de Datos Simulados:** Debido a estrictas normativas de privacidad y protección de datos personales que impiden el acceso a información real de ciudadanos y vehículos, todas las pruebas se realizaron con datos sintéticos. Esto implica que el prototipo no fue expuesto a la variabilidad, inconsistencias y casos atípicos que caracterizan a los datos del mundo real, lo cual podría influir en la lógica de negocio y en el manejo de errores en un entorno de producción.
   - **Suposiciones sobre la Calidad de la Evidencia:** El sistema asume que las evidencias fotográficas (imágenes de fotocomparendos) son capturadas con una calidad suficiente para su procesamiento. No se implementaron ni probaron mecanismos para manejar escenarios con imágenes de baja resolución, borrosas o con obstrucciones, que son comunes en la operación real.

2. **Integración y comparación con sistemas existentes**

   - **Integración Simulada con Sistemas Externos:** La interacción con plataformas gubernamentales clave como el RUNT y el SIMIT fue simulada a través de APIs de prueba (mocks). No se abordaron los desafíos técnicos y burocráticos de una

integración real, como los protocolos de comunicación, los tiempos de respuesta, la disponibilidad de los servicios y los posibles cuellos de botella.

- **Ausencia de Benchmarking Directo con el Sistema Actual (Fénix):** La falta de acceso al código fuente y a la arquitectura interna del sistema Fénix impidió realizar una comparación cuantitativa y directa en términos de rendimiento, costos operativos o eficiencia de procesos. El análisis comparativo se basó en las características conceptuales de ambas arquitecturas (centralizada vs. descentralizada).

3. **Aspectos técnicos y de escalabilidad**

- **Proyección de Costos como Escenario de Referencia:** Los costos de infraestructura y desarrollo estimados corresponden a un escenario de referencia. Los costos reales en un despliegue a gran escala podrían variar considerablemente dependiendo de factores como el número de nodos en la red, el volumen de almacenamiento en IPFS, el tráfico de red y la estrategia de persistencia de datos (pinning) que se adopte.
- **Estrategia de Persistencia en IPFS:** Para que la evidencia digital permanezca disponible a largo plazo en IPFS, es necesario que al menos un nodo la mantenga "pineada". El prototipo no implementa una política de pinning distribuida y resiliente, lo cual sería un requisito crítico para garantizar la cadena de custodia digital en un sistema de producción.

4. **Seguridad y robustez**

- **Limitaciones en Pruebas de Seguridad Avanzadas:** Si bien el prototipo implementa validaciones básicas de entrada (XSS, SQL injection, path traversal) y manejo de errores a nivel de aplicación, validadas mediante 26 pruebas automatizadas con 100 % de éxito, el alcance del proyecto **no contempló auditorías de seguridad exhaustivas** como las siguientes:
  - **Análisis estático de contratos inteligentes**: No se emplearon herramientas especializadas como *Slither*, *Mythril* o *MythX* para detectar vulnerabilidades en el

código Solidity del contrato `FineRegistry`.

- **Pruebas de penetración (pentesting)**: No se realizaron ataques simulados avanzados sobre la API REST más allá de las validaciones básicas implementadas.
- **Auditoría de permisos y autenticación**: El prototipo actual implementa validaciones básicas de entrada pero no incluye un sistema robusto de autenticación y autorización (JWT, OAuth2, RBAC) necesario para producción.
- **Validación exhaustiva de límites de archivos IPFS**: Aunque se validan formatos de imagen (JPG, PNG, WEBP) y límites de tamaño (10MB), no se implementaron validaciones avanzadas contra contenido malicioso embebido (steganografía, malware).
- **Protección contra ataques de denegación de servicio (DoS)**: No se implementaron mecanismos de *rate limiting*, *throttling* o *CAPTCHA* para prevenir abusos en los endpoints públicos.

Trabajo Futuro en Seguridad. Se recomienda que, en fases posteriores de desarrollo hacia producción, se incorporen las siguientes medidas:

a) Integración de herramientas de análisis estático como *Slither* para contratos Solidity.

b) Implementación de un sistema de autenticación y autorización basado en roles (RBAC) con tokens JWT.

c) Auditoría de seguridad externa realizada por especialistas en *blockchain security*.

d) Pruebas de penetración automatizadas utilizando herramientas como *OWASP ZAP* o *Burp Suite*.

e) Implementación de validación de archivos mediante *magic numbers* y análisis de contenido.

f) Configuración de límites de tasa (*rate limiting*) en la API REST.

Estas limitaciones **no comprometen la validez de la prueba de concepto**, dado que el objetivo principal es demostrar la viabilidad técnica de un modelo de confianza

descentralizado basado en inmutabilidad y verificabilidad, no el despliegue de un sistema en producción listo para operar en un entorno real con amenazas activas.

Diseño del prototipo

Se hace mención de que, aunque la documentación para elaborar el software está en español, es un estándar escribir código en inglés y, por tanto, para mantener la coherencia, los diagramas mostrados a continuación usarán este idioma para los nombres de las variables, funciones y clases.

Definición de requisitos

1. **Datos sobre infracciones de tráfico:** La captura de datos detallados sobre infracciones de tráfico, como la hora de la infracción, las coordenadas GPS, el tipo de infracción, los datos de identificación del vehículo e imágenes o vídeos, garantiza que cada incidente se documenta exhaustivamente. Este registro exhaustivo proporciona transparencia y responsabilidad, ya que los datos son inmutables y a prueba de manipulaciones una vez almacenados en la cadena de bloques. La inclusión de pruebas mediáticas refuerza aún más la credibilidad y verificabilidad de cada infracción, haciendo que los registros sean sólidos a efectos legales y administrativos.

2. **Información sobre el conductor:** Asociar las infracciones de tráfico a conductores concretos utilizando su dirección Ethereum (clave pública), los datos KYC si es necesario, y los números de identificación del conductor permite un seguimiento y una rendición de cuentas precisos. Esta vinculación permite al sistema personalizar el seguimiento y la verificación de las sanciones, garantizando que las sanciones se atribuyan correctamente a las personas adecuadas. El uso de datos KYC garantiza que las identidades de los conductores puedan verificarse de forma fiable, lo que resulta esencial para mantener la integridad y fiabilidad del sistema.

3. **Datos de la sanción:** Registrando los datos de la sanción, incluyendo el tipo de sanción, el importe de la sanción y el estado del pago de la sanción facilita la ejecución automatizada de las sanciones a través de contratos inteligentes. Esta automatización reduce la carga administrativa de personal y garantiza que las sanciones se apliquen de forma coherente y transparente. El registro inmutable de las sanciones y su estado de pago en la blockchain garantiza que el proceso sea justo y responsable, proporcionando una pista de auditoría clara para todas las transacciones financieras relacionadas con las infracciones de tráfico.

4. **Eventos de contratos inteligentes:** El registro de eventos de contratos inteligentes, como el registro de nuevas infracciones de tráfico o la ejecución de sanciones, con datos relevantes y marcas de tiempo, garantiza que todas las acciones significativas se documenten de forma transparente. Este registro de eventos mejora la trazabilidad y la rendición de cuentas, proporcionando un registro cronológico de las actividades importantes del sistema. Esta transparencia es crucial para las auditorías y revisiones, ya que ayuda a generar confianza en las operaciones del sistema.

5. **Datos de las transacciones de la cadena de bloques:** El seguimiento de los datos de las transacciones de la cadena de bloques, incluido el hash de la transacción, las direcciones del remitente/receptor y las tarifas del gas, proporciona un registro detallado de todas las interacciones dentro del sistema. Estos datos permiten supervisar y auditar las transacciones, garantizando la transparencia y la trazabilidad. Además, hacer un seguimiento de las tarifas de gas ayuda a gestionar y optimizar los costes asociados a la ejecución de transacciones en la blockchain, que es importante para mantener la rentabilidad del sistema.

6. **Dispositivos de datos IoT:** La integración de datos de dispositivos IoT, como sensores o cámaras, junto con marcas de tiempo e identificación del dispositivo, puede mejorar las pruebas recopiladas para infracciones de tráfico. Estos datos en tiempo real proporcionan contexto adicional y pruebas corroborativas, haciendo que los registros de infracciones sean más sólidos y fiables. El uso de dispositivos IoT también puede automatizar la detección y el registro de infracciones, aumentando la eficiencia y la precisión del sistema.

7. **Opiniones de los usuarios:** La recopilación de opiniones de los usuarios, incluidos el tipo de opinión, los comentarios y las valoraciones de los usuarios, ayuda a los administradores del sistema a comprender las experiencias y percepciones de los usuarios. Esta información es valiosa para identificar áreas de mejora como la usabilidad y funcionalidad del sistema. Involucrar a los usuarios de esta manera puede conducir a un diseño del sistema más centrado en el usuario, mejorando la satisfacción y la eficacia general.

8. **Datos de cumplimiento:** El registro de los datos de cumplimiento, incluido el estado de

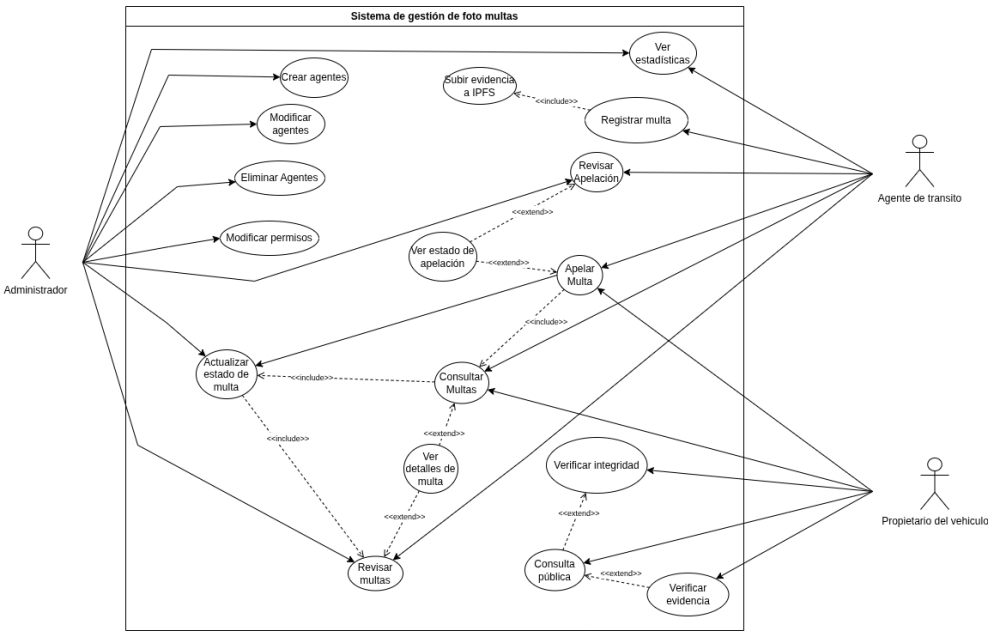
cumplimiento y los detalles normativos, garantiza que el sistema se adhiere a las leyes y normativas de tráfico locales. Este seguimiento es vital para demostrar el cumplimiento de la normativa y evitar problemas legales. El mantenimiento de registros de cumplimiento detallados también facilita las auditorías reglamentarias en, proporcionando pruebas transparentes de que el sistema funciona dentro de las normas legales, lo que es esencial para generar confianza y credibilidad entre las partes interesadas.

Diagrama de casos de uso

Este diagrama presenta las funcionalidades principales del sistema desde la perspectiva de los actores involucrados: agentes de tránsito, ciudadanos y administradores.

Figura 6

Diagrama de casos de uso del prototipo de gestión de infracciones de tránsito



Nota. Diagrama de casos de uso del sistema de gestión de infracciones de tránsito.

Diagrama de clases

La arquitectura de clases del sistema implementa el patrón Controller-Service-Repository, adaptado para soportar la arquitectura híbrida blockchain. Se distinguen tres capas principales de lógica de negocio:

Primera capa - servicios de blockchain híbrida. Gestiona la interacción con ambas blockchains de forma independiente:

- **HyperledgerService:** Coordina operaciones con la red privada de Hyperledger Fabric, incluyendo registro completo de infracciones, gestión de apelaciones y control de acceso.
- **EthereumService:** Maneja la publicación de metadatos en la blockchain pública de Ethereum y proporciona interfaces de consulta ciudadana.
- **SyncService:** Implementa la lógica de sincronización entre blockchains, extrayendo metadatos públicos de Hyperledger y publicándolos en Ethereum con hashes de integridad.

Segunda capa - almacenamiento distribuido dual. Separa el almacenamiento de evidencias según su nivel de sensibilidad:

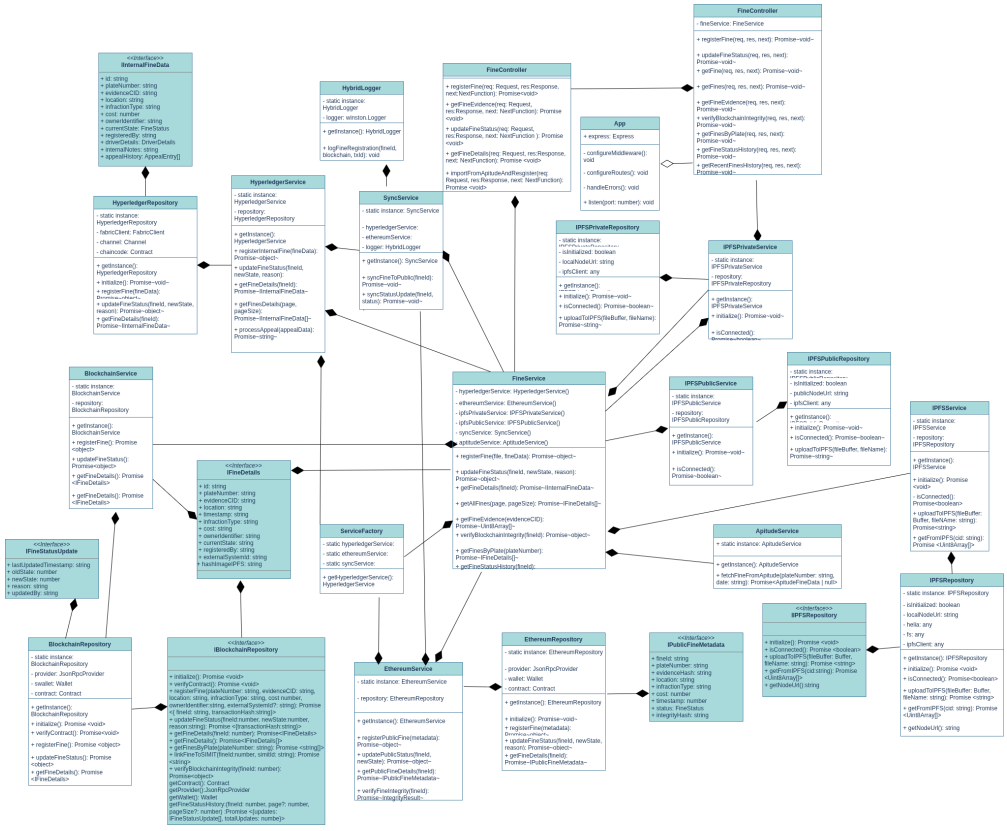
- **IPFSPrivateService:** Gestiona el almacenamiento de evidencias completas en IPFS privado, accesible solo para usuarios autorizados.
- **IPFSPublicService:** Maneja la publicación de hashes de evidencias en IPFS público para verificación ciudadana.

Tercera capa - orquestación y administración. Coordina las operaciones entre todas las capas:

- **FineService:** Orquesta el flujo completo de registro de infracciones, coordinando el almacenamiento en IPFS privado, registro en Hyperledger Fabric y sincronización a Ethereum.
- **FineController:** Expone endpoints REST para las operaciones del sistema, diferenciando entre operaciones internas (requieren autenticación) y consultas públicas.

Figura 7

Diagrama de clases del sistema de gestión de multas



Nota. Diagrama de clases mostrando la arquitectura híbrida blockchain del sistema.

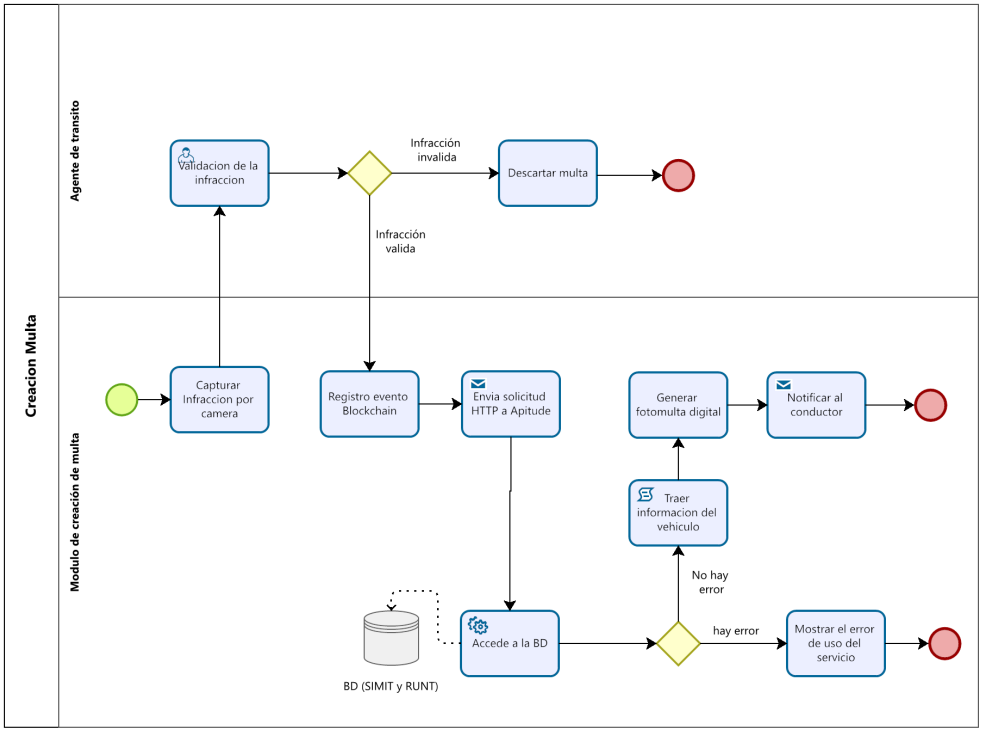
Diagramas de actividades

Los siguientes diagramas describen los flujos de proceso principales del sistema, mostrando la secuencia de operaciones y decisiones en la creación de multas y gestión de apelaciones.

Proceso de creación de multa

Figura 8

Diagrama de actividades para el proceso de creación de multa



Nota. Diagrama de actividades del proceso de creación y registro de multas.

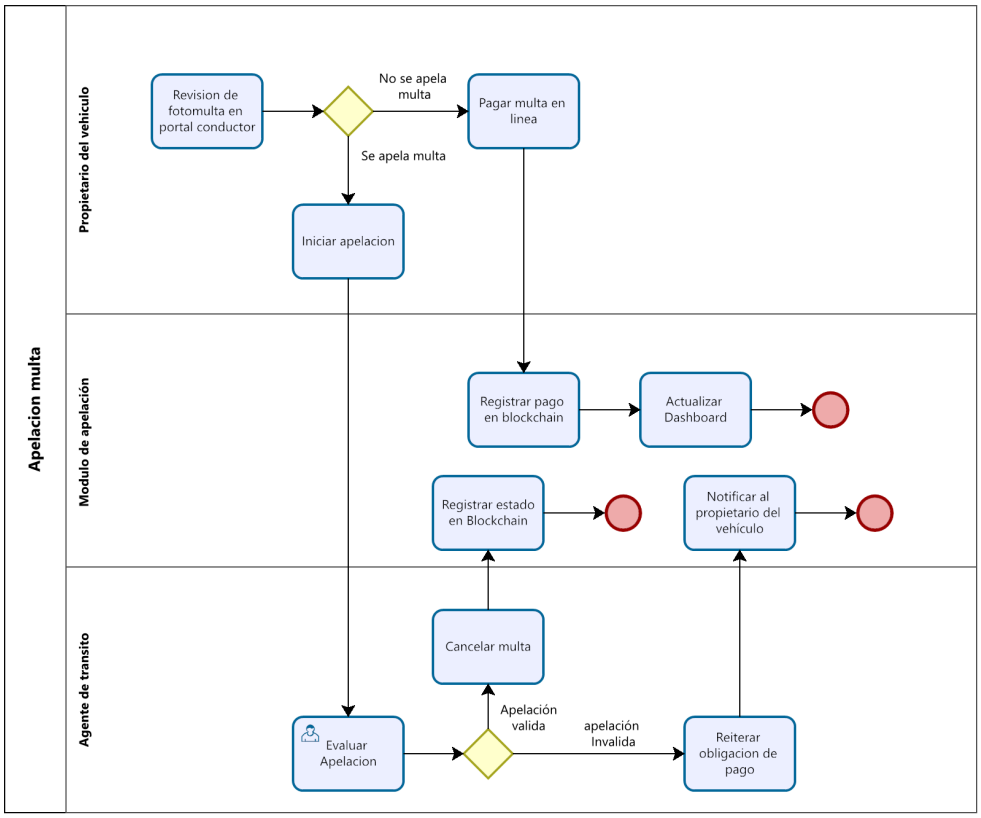
Proceso de apelación de multa

Diagrama de despliegue

La arquitectura se compone de los siguientes elementos:

Figura 9

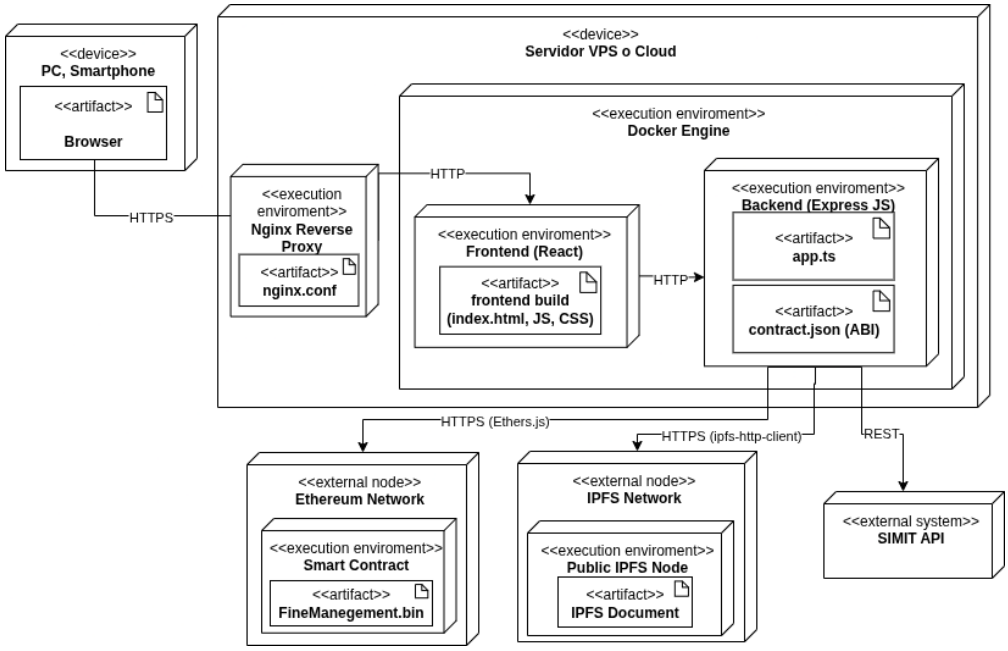
Diagrama de actividades para el proceso de apelación de multa



Nota. Diagrama de actividades del proceso de apelación de multas.

Figura 10

Diagrama de despliegue de la arquitectura del sistema



Nota. Diagrama de despliegue de la arquitectura híbrida blockchain propuesta.

Capa privada - Hyperledger Fabric.

- **Nodos Peer:** Mantienen el ledger privado y ejecutan *chaincode* (lógica de negocio). Estos nodos almacenan la información completa de las infracciones, incluyendo datos sensibles y evidencias completas.
- **Nodo Orderer:** Coordina el consenso entre peers utilizando el algoritmo *PBFT* (*Practical Byzantine Fault Tolerance*), garantizando la validación eficiente de transacciones.
- ***Certificate Authority* (*CA*):** Gestiona las identidades digitales y permisos de usuarios autorizados (administradores, agentes de tránsito).
- ***IPFS* Privado:** Almacena las evidencias fotográficas completas con sus metadatos, accesible solo para usuarios autorizados.

Capa pública - Ethereum.

- **Nodos *Ethereum*:** Ejecutan *Smart Contracts* que almacenan metadatos públicos de infracciones sin información personal sensible.
- ***IPFS* Público:** Almacena *hashes* de evidencias para verificación ciudadana, sin exponer imágenes completas.

Servicio de sincronización. Un servicio intermediario sincroniza los datos entre ambas blockchains, extrayendo metadatos no sensibles de *Hyperledger Fabric* y publicándolos en *Ethereum* junto con *hashes* de integridad. Este servicio garantiza la consistencia entre ambas capas mediante verificación cruzada de *hashes* criptográficos.

La arquitectura se conecta mediante servicios web a APIs externas como Apitude para acceder a información del Registro Único Nacional de Tránsito (RUNT) y del Sistema Integrado de Información sobre Multas y Sanciones por Infracciones de Tránsito (SIMIT), obteniendo datos de conductores, vehículos y el estado de multas. Esta integración permite validar la información de infracciones contra registros oficiales sin comprometer la privacidad de los datos almacenados en la capa privada.

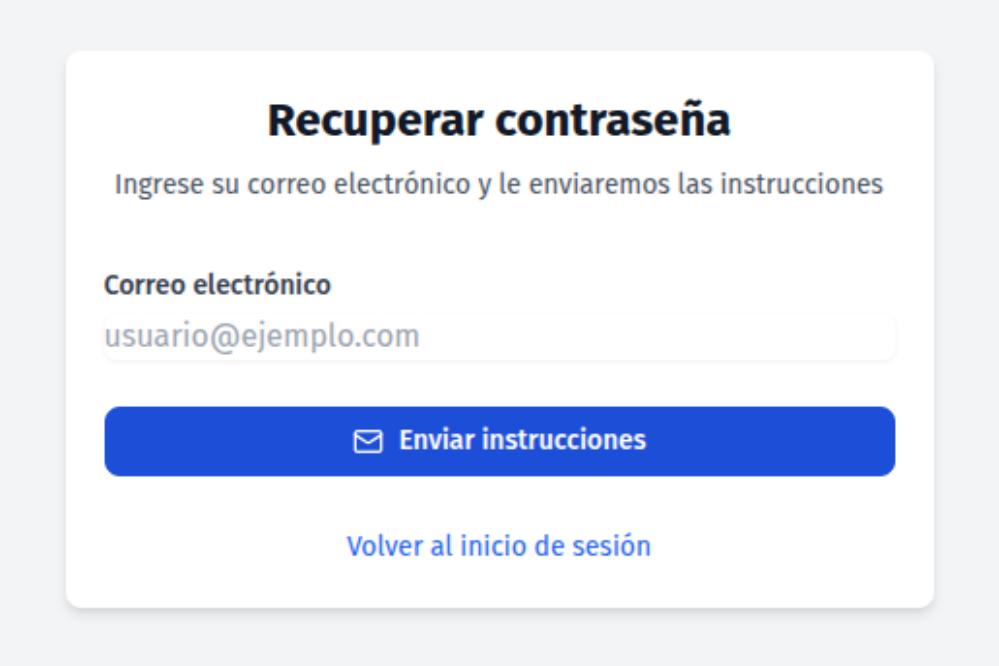
Figura 11

Pantalla de login del sistema

Nota. Pantalla de login del sistema de gestión de multas.

Figura 12

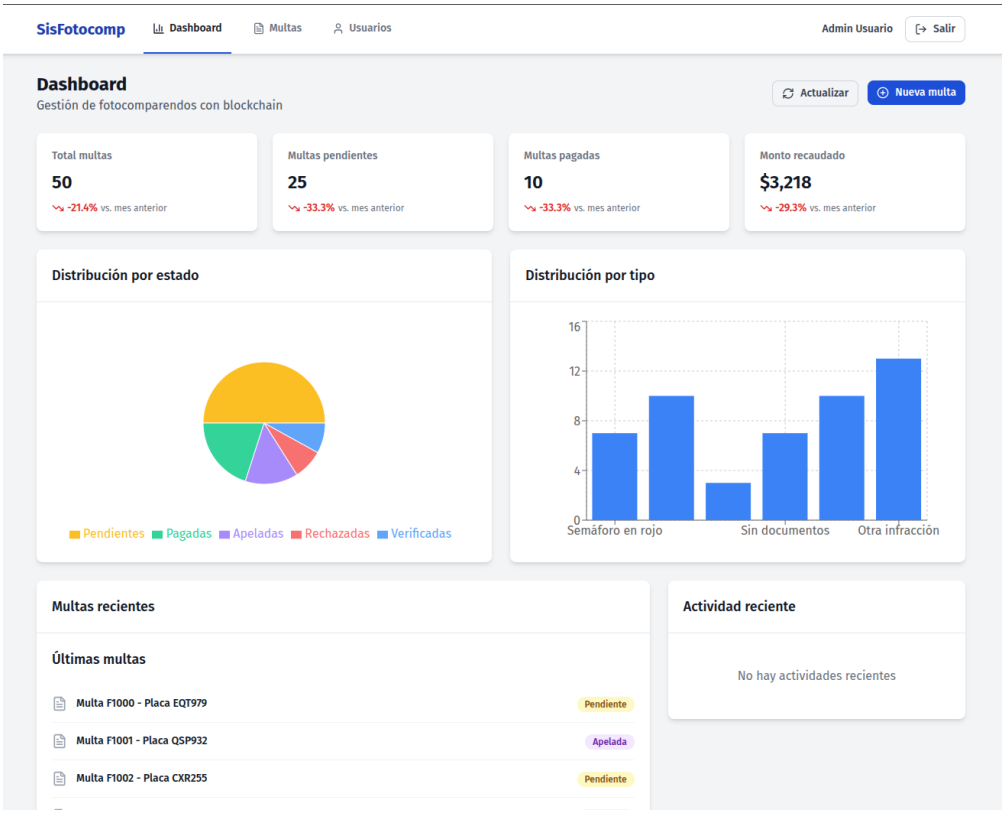
Pantalla de recuperación de contraseña



Nota. Pantalla de recuperación de contraseña.

Figura 13

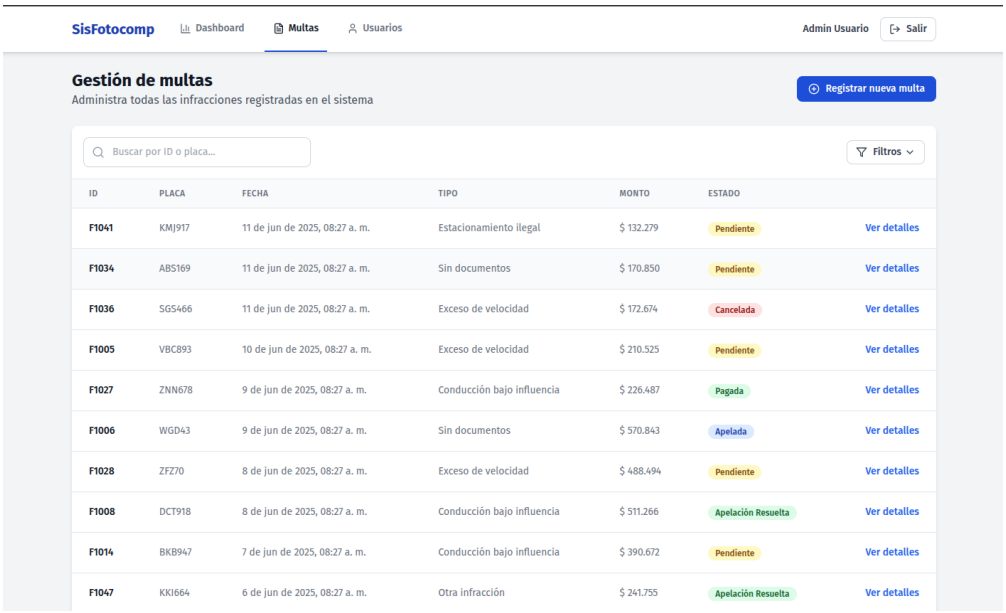
Dashboard del agente de tránsito



Nota. Dashboard principal del agente de tránsito.

Figura 14

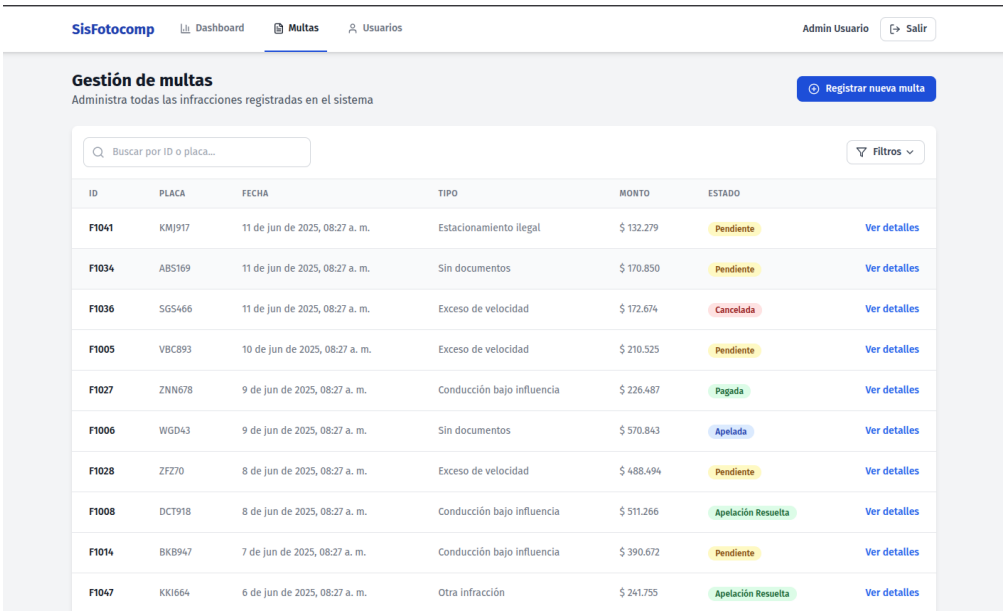
Pantalla de consulta del estado de multa



Nota. Pantalla de consulta del estado de multa para agentes.

Figura 15

Pantalla de consulta de detalle de multa



Nota. Pantalla de detalle completo de multa para agentes.

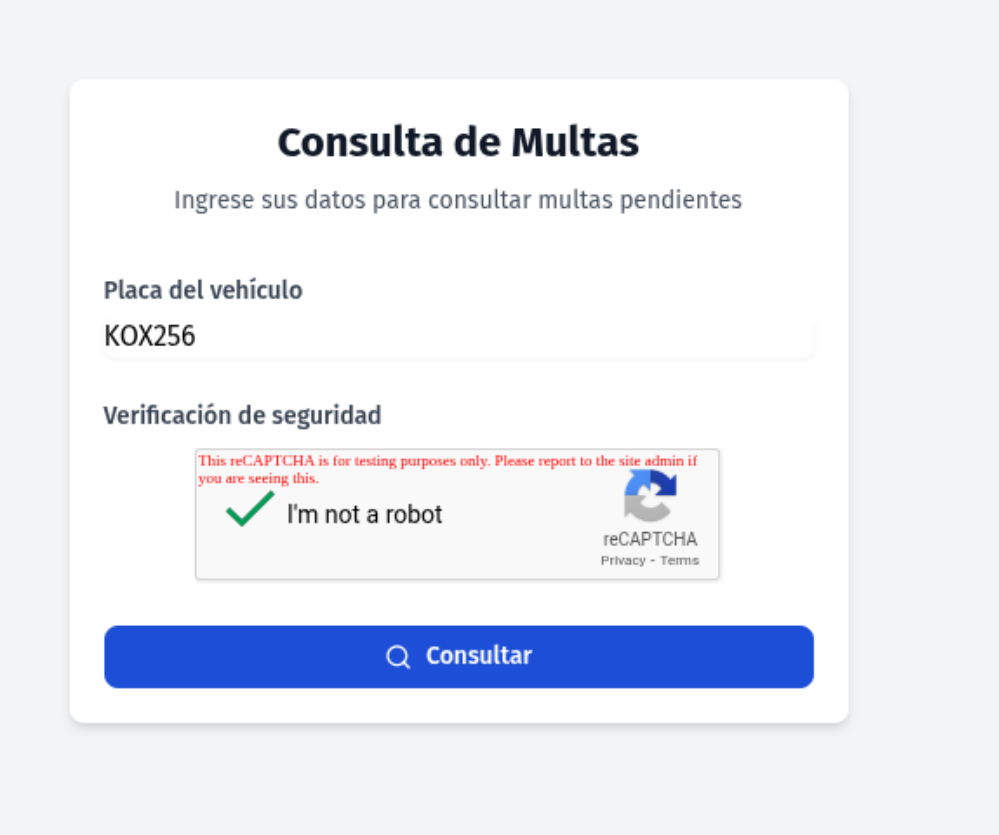
Interfaz de usuario

Compartidas.

Vista agente.

Figura 16

Pantalla de consulta de multas para propietarios de vehículos

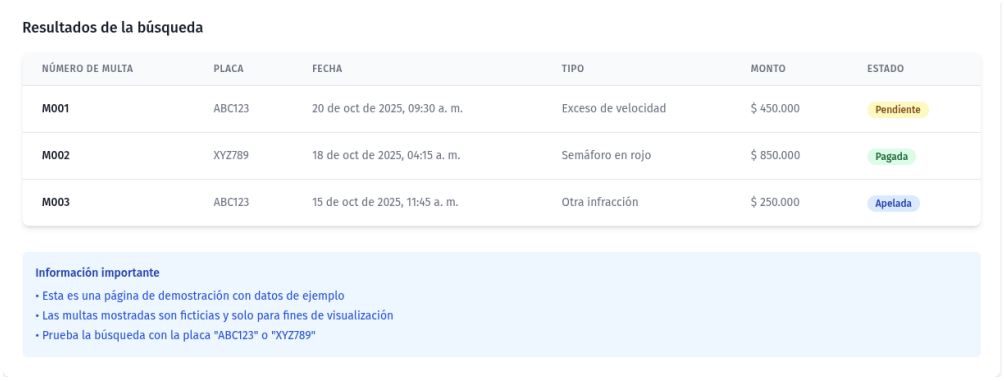


Nota. Pantalla de consulta de multas para propietarios de vehículos.

Vista propietario de vehículo.

Figura 17

Pantalla de detalle de multa para propietarios de vehículos



Nota. Pantalla de detalle de multa para propietarios de vehículos.

Implementación del prototipo

La implementación del prototipo se llevó a cabo siguiendo la arquitectura híbrida *blockchain* diseñada en la sección Diseño del prototipo, integrando *Hyperledger Fabric* para la gestión privada de datos sensibles, *Ethereum* para la transparencia pública e *IPFS* dual para el almacenamiento distribuido de evidencias.

Entorno de desarrollo y herramientas

El desarrollo del prototipo se realizó en un entorno Unix (Linux) utilizando las siguientes herramientas:

- **Sistema Operativo:** Ubuntu 22.04 LTS
- **Control de Versiones:** Git 2.34+ para gestión de código fuente
- **Entorno de Ejecución:** *Node.js* v20.18.0 con npm v10.0.0
- **Gestión de Dependencias:** npm para paquetes *JavaScript*/*TypeScript*
- **IDE:** *Visual Studio Code* con extensiones para *Solidity*, *Go* y *TypeScript*

Stack tecnológico implementado

Tecnologías backend

El backend del sistema se implementó utilizando tecnologías modernas de JavaScript/TypeScript:

- **Framework Web:** *Express.js* v4.18.2 - Framework minimalista para *Node.js*
- **Lenguaje:** *TypeScript* v5.8.3 - Superset tipado de *JavaScript*
- **Validación:** Express-validator v7.2.1 y Joi v17.13.3 - Validación de datos de entrada
- **Documentación API:** Swagger-jsdoc v6.2.8 y Swagger-ui-express v5.0.1
- **Manejo de Archivos:** Multer v1.4.5-lts.2 - Procesamiento de uploads multipart
- **Cliente HTTP:** Axios v1.9.0 - Comunicación con APIs externas

Tecnologías blockchain

Capa pública - Ethereum. Para la implementación de la capa pública (justificada en la sección Metodología), se utilizó el ecosistema de Ethereum con las siguientes tecnologías:

- **Framework de Desarrollo:** *Hardhat* v2.24.0 - Entorno de desarrollo *Ethereum*
- **Biblioteca de Interacción:** *Ethers.js* v6.14.0 - Cliente para interactuar con *Ethereum*
- **Lenguaje de Contratos:** *Solidity* v0.8.28 - Lenguaje para *Smart Contracts*
- **Contratos Base:** *OpenZeppelin Contracts* v5.3.0 - Librería de contratos seguros y auditados
- **Generación de Tipos:** *TypeChain* v8.3.2 - Generación automática de tipos *TypeScript* desde *ABI*

Capa privada - Hyperledger Fabric. Para la implementación de la capa privada (justificada en la sección Metodología), se utilizó *Hyperledger Fabric* con las siguientes tecnologías:

- **Plataforma:** *Hyperledger Fabric* v2.5 - Blockchain permisionada empresarial
- **Lenguaje Chaincode:** *Go* v1.21+ - Lenguaje para desarrollo de *chaincode*
- **SDK:** *Fabric SDK* para *Node.js* - Interacción desde el backend
- **Consenso:** *Raft* - Algoritmo de consenso para tolerancia a fallas
- **Gestión de Identidades:** *Fabric CA* - *Certificate Authority* para control de acceso

Almacenamiento de evidencias

La implementación de IPFS se realizó en dos capas diferenciadas:

- **Implementación:** *Kubo* v0.34.1 - Implementación de referencia de *IPFS*
- **Cliente *JavaScript*:** *ipfs-http-client* v60.0.1 - *API HTTP* para *IPFS*
- ***IPFS* Privado:** Nodo local para almacenamiento de evidencias completas
- ***IPFS* Público:** Gateway público para *hashes* de verificación ciudadana
- **Protocolo de Contenido:** *Multiformats* v13.3.3 - Manejo de *CIDs*

Tecnologías frontend

El frontend se desarrolló con tecnologías modernas de React:

- **Framework:** *React* v18.3.1 - Biblioteca para interfaces de usuario
- **Bundler:** *Vite* v5.4.2 - Herramienta de build ultrarrápida
- **Lenguaje:** *TypeScript* v5.5.3 - Tipado estático
- **Estilos:** *Tailwind CSS* v3.4.1 - Framework de utilidades *CSS*
- **Enrutamiento:** *React Router DOM* v6.22.3 - Navegación entre vistas
- **Estado Global:** *Zustand* v4.5.2 - Gestión de estado ligera
- **Gráficos:** *Recharts* v2.12.3 - Librería de visualización de datos
- **Iconos:** *Lucide React* v0.344.0 - Iconos modulares

Frameworks de testing

Se implementaron pruebas automatizadas en múltiples capas:

- **Backend:** *Vitest* v3.2.3 - Framework de testing para *Vite*
- **Frontend:** *Jest* v30.0.3 - Framework de testing para *React*
- ***Smart Contracts*:** *Hardhat Testing* - Framework integrado de *Hardhat*
- **Aserciones:** *Chai* v4.5.0 - Librería de aserciones
- **Testing de *UI*:** *React Testing Library* v16.3.0 - Testing de componentes *React*

Capa pública Ethereum

La arquitectura de la capa pública se describe en detalle en la sección Diseño del prototipo. Esta sección se enfoca en los aspectos específicos de implementación técnica.

Desarrollo del smart contract

El *Smart Contract* FineManagement.sol implementa la lógica de negocio para la gestión pública de infracciones de tránsito. El contrato se desarrolló en *Solidity* v0.8.28 y hereda de *Ownable* (*OpenZeppelin*) para control de acceso.

Estructura de datos. El contrato define dos estructuras principales para modelar las multas y su historial de estados:

```
enum FineState {
    PENDING,
    PAID,
    APPEALED,
    RESOLVED_APPEAL,
    CANCELLED
}
```

Este enum define los cinco estados que el *smart contract* del prototipo implementa como prueba de concepto. Es importante destacar que el código utiliza nomenclatura en inglés siguiendo las mejores prácticas de la industria del desarrollo de *software blockchain*, donde *Solidity* y *Ethereum* tienen convenciones establecidas en este idioma.

La Tabla 11 presenta el mapeo sistemático entre estos estados implementados en el *smart contract* (inglés) y los estados conceptuales del proceso de fotocomparendos descritos en la Introducción (sección 1.2). Como se observa en la tabla, el prototipo implementa un subconjunto funcional de los ocho estados conceptuales completos, siendo suficiente para validar los principios de inmutabilidad y trazabilidad. Los estados NOTIFICADA y CERRADA no fueron implementados en esta versión pero pueden agregarse en una implementación de producción sin modificaciones arquitectónicas sustanciales.

Tabla 11. *Mapeo entre estados conceptuales y estados del smart contract*

Estado Conceptual (ES)	**Estado en *Smart Contract* (EN)**	**Descripción del Mapeo**
GENERADA / PENDIENTE_RESPUESTA	PENDING	Estado inicial tras registro del comparendo. Engloba tanto la generación inicial como el período de espera de respuesta ciudadana. El *smart contract* usa PENDING para representar cualquier comparendo que aún no ha sido resuelto mediante pago, apelación o cancelación.
PAGADA	PAID	Comparendo cuya obligación económica ha sido saldada. Mapeo directo 1:1 entre concepto y código. Representa el cierre exitoso del proceso mediante pago voluntario o forzoso.

Continúa en la siguiente página

Tabla 11. *(Continuación)*

Estado Conceptual (ES)	**Estado en *Smart Contract* (EN)**	**Descripción del Mapeo**
EN_APELACION	APPEALED	Comparendo bajo proceso de revisión por PQRSD ciudadana. Mapeo directo 1:1. Este estado es crítico para la trazabilidad de disputas y fundamenta la necesidad de inmutabilidad de evidencia.
RESUELTA_APELACION	RESOLVED_APPEAL	Apelación procesada con decisión administrativa (confirmación, revocación parcial/total, o anulación). Mapeo directo 1:1. Estado terminal para el flujo de apelaciones.
CANCELADA	CANCELLED	Comparendo cancelado administrativamente por anulación judicial, defectos procedimentales, o corrección de errores. Mapeo directo 1:1. Requiere máxima trazabilidad para prevenir cancelaciones irregulares.

Continúa en la siguiente página

Tabla 11. *(Continuación)*

Estado Conceptual (ES)	**Estado en *Smart Contract* (EN)**	**Descripción del Mapeo**
NOTIFICADA	*No implementado*	Este estado conceptual no tiene equivalente en el *smart contract* del prototipo. En una implementación de producción, se agregaría estado NOTIFIED entre PENDING y las transiciones subsecuentes. La notificación podría registrarse mediante evento *blockchain* con *timestamp* inmutable.
CERRADA	*No implementado*	Estado final conceptual que indica cierre definitivo del proceso. En el prototipo, los estados PAID, RESOLVED_APPEAL y CANCELLED funcionan como estados terminales. Una implementación completa podría agregar estado CLOSED explícito para uniformidad.

Nota. Elaboración propia. El *smart contract* implementa un subconjunto de estados como prueba de concepto.

```
struct Fine {
    uint256 id;
```

```
    string plateNumber;
    string evidenceCID;        // \textit{CID} de \textit{IPFS} público
    string location;
    uint256 timestamp;
    string infractionType;
    uint256 cost;
    string ownerIdentifier;
    FineState currentState;
    address registeredBy;
    string externalSystemId;   // ID del \textit{SIMIT}
}


struct FineStatusUpdate {
    uint256 lastUpdatedTimestamp;
    FineState oldState;
    FineState newState;
    string reason;
    address updatedBy;
}
```

Mapeos para consultas eficientes. Para optimizar las consultas se implementaron mapeos especializados:

```
mapping(uint256 => Fine) public fines;
mapping(uint256 => FineStatusUpdate[]) public fineStatusHistory;
mapping(string => uint256[]) public finesByPlate;
mapping(string => uint256[]) public finesByOwner;
mapping(address => bool) public operators;
```

Funciones principales. Las funciones críticas del contrato garantizan la inmutabilidad y trazabilidad:

- `registerFine()`: Registra una nueva multa con validaciones de entrada. Incrementa el contador de IDs, almacena la estructura Fine en el mapping, actualiza los índices de búsqueda por placa y propietario, y emite el evento FineRegistered.
- `updateFineStatus()`: Actualiza el estado de una multa existente. Valida que la multa exista y que el nuevo estado sea diferente al actual, registra el cambio en el historial y emite el evento FineStatusUpdated.
- `getFineDetails()`: Retorna los detalles completos de una multa dado su ID.
- `getFinesByPlate()`: Retorna un array de IDs de multas asociadas a un número de placa específico.
- `getPaginatedFines()`: Implementa paginación eficiente para consultas de múltiples multas, evitando problemas de límite de gas en consultas grandes.
- `getFineStatusHistory()`: Retorna el historial paginado de cambios de estado de una multa, permitiendo auditoría completa de su ciclo de vida.

Control de acceso. El contrato implementa un sistema de roles mediante el modificador `onlyOperator`, que restringe operaciones críticas (registro y actualización de multas) a direcciones autorizadas. El propietario del contrato puede agregar o remover operadores mediante las funciones `addOperator()` y `removeOperator()`.

Eventos para auditoría. Se definieron eventos para facilitar la auditoría externa:

```
event FineRegistered(
    uint256 indexed fineId,
    string indexed plateNumber,
    string evidenceCID,
    string ownerIdentifier,
    uint256 cost,
    uint256 timestamp
);
```

```
event FineStatusUpdated(
    uint256 indexed fineId,
    FineState indexed oldState,
    FineState indexed newState,
    string reason,
    uint256 timestamp
);
```

Estos eventos permiten que aplicaciones externas puedan suscribirse a cambios en tiempo real y mantener bases de datos sincronizadas sin necesidad de polling.

Despliegue y configuración

Configuración de Hardhat. El framework *Hardhat* se configuró para soportar despliegue en múltiples redes:

```
module.exports = {
  solidity: {
    version: "0.8.28",
    settings: {
      optimizer: {
        enabled: true,
        runs: 200
      }
    }
  },
  networks: {
    localhost: {
      url: "http://127.0.0.1:8545"
    },
    sepolia: {
```

```
      url: process.env.SEPOLIA_RPC_URL,
      accounts: [process.env.PRIVATE_KEY]
    }
  }
};
```

El optimizador de *Solidity* se habilitó con 200 runs, priorizando la eficiencia de ejecución sobre el tamaño del bytecode desplegado.

Script de despliegue. Se implementó un script automatizado para el despliegue del contrato:

```
// scripts/deploy.mjs
async function main() {
  const FineManagement = await ethers.getContractFactory(
    "FineManagement"
  );
  const fineManagement = await FineManagement.deploy();
  await fineManagement.waitForDeployment();

  const address = await fineManagement.getAddress();
  console.log(`FineManagement deployed to: ${address}`);
}
```

Red de despliegue. El prototipo se desplegó inicialmente en la red local de *Hardhat* para desarrollo y pruebas. Para demostración pública, se configuró el despliegue en *Sepolia Testnet*, una red de pruebas de *Ethereum* que permite validación externa sin costos reales.

Configuración de infraestructura

Configuración de Hyperledger Fabric. La red privada se orquestó mediante *Docker Compose* con tres organizaciones (Secretaría de Movilidad, Policía de Tránsito, Auditoría) y un nodo *orderer* con consenso *Raft*. La configuración incluye *Certificate Authorities* por organización y canales privados para separación de datos sensibles.

Configuración de IPFS. Se implementó un nodo *IPFS* local (*Kubo* v0.34.1) con *API HTTP* habilitado para almacenamiento de evidencias. La configuración incluye *pinning* automático y control de acceso restringido al backend.

Instalación y despliegue

Prerrequisitos del sistema. El prototipo requiere Ubuntu 22.04 LTS o superior con las siguientes dependencias:

- ***Node.js*:** v20.18.0+ con npm v10.0.0+
- ***Docker*:** v24.0+ con *Docker Compose* v2.20+
- ***IPFS Kubo*:** v0.34.1+ para almacenamiento descentralizado
- **Git:** Para clonado de repositorios

Proceso de instalación. La instalación se realiza mediante los siguientes pasos:

1. **Clonar repositorios:**

```
git clone https://github.com/CristianGT089/backend-multas
git clone https://github.com/k-delta/fotomultas-front
```

2. **Configurar backend:**

```
cd backend-multas
npm install
cp env.example .env
# Editar .env con configuración local
```

3. **Iniciar servicios:**

```
# Terminal 1: IPFS
ipfs daemon &
```

```
# Terminal 2: Hardhat
npm run dev:contracts

# Terminal 3: Backend
npm run dev
```

4. **Configurar frontend:**

```
cd ../fotomultas-front
npm install
npm run dev
```

Verificación del despliegue. El sistema estará disponible en:

- **Frontend:** http://localhost:5173
- **Backend API:** http://localhost:3000
- **Documentación Swagger:** http://localhost:3000/api-docs
- **IPFS Gateway:** http://localhost:5001

Capa privada Hyperledger Fabric

La arquitectura de la capa privada y su justificación técnica se detallan en la sección Metodología y la sección Diseño del prototipo. Esta sección describe la configuración técnica específica implementada.

Configuración de la red

La red de *Hyperledger Fabric* se configuró con la siguiente topología:

- **Organizaciones:** Tres organizaciones (Secretaría de Movilidad, Policía de Tránsito, Auditoría)

- **Peers:** Dos nodos *peer* por organización para redundancia
- ***Orderer*:** Un nodo *orderer* con consenso *Raft*
- **Canal:** Un canal llamado `fotomultas-channel` compartido por las tres organizaciones
- ***Certificate Authority*:** Una *CA* por organización para gestión de identidades

La configuración se definió mediante archivos *YAML* estándar de *Fabric*: `configtx.yaml` para la configuración del canal, `crypto-config.yaml` para la generación de certificados y `docker-compose.yaml` para la orquestación de contenedores.

Desarrollo del chaincode

El *chaincode* se implementó en *Go*, siguiendo la estructura de `contractapi.Contract` de *Hyperledger Fabric*. Las funciones principales incluyen:

- `RegisterInternalFine()`: Registra una multa completa con datos sensibles en la blockchain privada. Almacena información del conductor, detalles de la evidencia completa y notas internas.
- `UpdateFineStatus()`: Actualiza el estado de una multa y registra el cambio en el historial privado.
- `ProcessAppeal()`: Gestiona el proceso de apelación, almacenando las evidencias presentadas por el ciudadano y la resolución del agente.
- `GetFineDetails()`: Retorna los detalles completos de una multa, incluyendo información sensible accesible solo para usuarios autorizados.
- `AuditTrail()`: Proporciona un historial de auditoría completo de todas las operaciones realizadas sobre una multa específica.

Gestión de datos privados. Se utilizó la funcionalidad de *Private Data Collections* de *Fabric* para separar información altamente sensible (como datos de identificación del conductor) que solo debe ser accesible por la organización que la registró.

Control de acceso basado en atributos. El chaincode implementa validaciones basadas en los atributos del certificado del invocador, verificando roles (agente, administrador, auditor) antes de permitir operaciones sensibles.

Sincronización entre blockchains

Arquitectura del servicio

El servicio de sincronización se implementó como un proceso independiente en *Node.js* que escucha eventos de la blockchain privada (*Hyperledger Fabric*) y sincroniza metadatos públicos a la blockchain pública (*Ethereum*).

Componentes principales.

- ***Event Listener*:** Módulo que se suscribe a eventos del *chaincode* de *Fabric*
- ***Metadata Extractor*:** Componente que filtra datos sensibles y extrae solo metadatos públicos
- ***Hash Generator*:** Genera *hash SHA-256* de integridad del registro completo
- ***Ethereum Publisher*:** Publica los metadatos en el *Smart Contract* de *Ethereum*
- ***Consistency Validator*:** Verifica que los datos se sincronizaron correctamente

Flujo de sincronización

El proceso de sincronización sigue estos pasos:

1. El *chaincode* de *Fabric* emite un evento `FineRegistered` o `FineUpdated`
2. El *Event Listener* captura el evento y extrae el ID de la multa
3. Se consulta el registro completo desde *Fabric*
4. El *Metadata Extractor* genera la estructura pública:
    - ID de multa
    - Número de placa
    - *Hash* de evidencia (*CID* de *IPFS* público)

- Ubicación
- Tipo de infracción
- Costo
- Timestamp
- Estado actual

5. Se genera un *hash* de integridad del registro completo privado
6. Se publica el registro público en *Ethereum* mediante `registerPublicFine()`
7. Se valida que el *transaction hash* de *Ethereum* sea exitoso
8. Se registra la sincronización en un log de auditoría

Manejo de errores y reintentos

El servicio implementa un mecanismo de reintentos con backoff exponencial para manejar fallas temporales de red o gas insuficiente en Ethereum. Los eventos fallidos se encolan para reintento posterior, garantizando eventual consistencia.

Implementación de IPFS dual

IPFS privado

Se configuró un nodo IPFS local para el almacenamiento de evidencias completas:

- **Configuración:** Nodo *Kubo* v0.34.1 con *API HTTP* habilitado solo para localhost
- **Estrategia de *Pinning*:** *Pinning* automático de todas las evidencias subidas
- **Control de Acceso:** *API* accesible solo desde el backend, sin exposición pública
- **Persistencia:** Almacenamiento en disco local con respaldo periódico

El servicio IPFSPrivateService implementa las siguientes funciones:

- `uploadToIPFS(fileBuffer, fileName)`: Sube una evidencia completa y retorna su *CID*
- `getFromIPFS(cid)`: Recupera un archivo dado su *CID*
- `isConnected()`: Verifica la conectividad con el daemon de IPFS

IPFS público

Para la capa pública se utilizó un gateway público de IPFS que permite:

- Publicación de *hashes* de evidencias para verificación ciudadana
- Acceso sin autenticación a través de *HTTP*
- Verificación de integridad mediante comparación de *CIDs*

El IPFSPublicService gestiona la publicación de hashes en el nodo público, manteniendo la separación entre evidencias completas (privadas) y hashes verificables (públicos).

Desarrollo del backend

Arquitectura de servicios

El backend implementa el patrón Controller-Service-Repository adaptado para arquitectura híbrida:

Capa de controladores. `FineController` maneja las peticiones HTTP y delega la lógica de negocio a los servicios.

Capa de servicios.

- `FineService`: Orquestador principal que coordina operaciones entre blockchains
- `HyperledgerService`: Interacción con la red privada de *Fabric*
- `EthereumService`: (Implementado como `BlockchainService`) Interacción con *Ethereum*
- `IPFSPrivateService`: Gestión de evidencias en IPFS privado
- `IPFSPublicService`: Gestión de hashes en IPFS público
- `AptitudeService`: Integración con API externa RUNT/SIMIT (simulada)

Capa de repositorios. Los repositorios abstraen el acceso a las fuentes de datos (blockchains e IPFS).

Endpoints principales

La Tabla 12 describe los endpoints principales implementados en la API REST para la gestión de fotocomparendos.

Tabla 12. *Endpoints principales de la API REST*

Método	**Endpoint**	**Descripción**
POST	/api/fines	Registra nueva multa (*IPFS* + ambas blockchains)
GET	/api/fines/:fineId	Consulta detalles completos (desde *Fabric*)
PUT	/api/fines/:fineId/status	Actualiza estado de multa
GET	/api/fines/:fineId/evidence	Obtiene evidencia desde *IPFS* privado
GET	/api/fines/:fineId/integrity	Verifica integridad cruzada entre blockchains
GET	/api/fines/by-plate/:plateNumber	Consulta pública desde *Ethereum*

Nota. Elaboración propia.

Middleware de seguridad

Se implementaron middlewares para:

- Autenticación mediante *JSON Web Tokens* (*JWT*)
- Validación de datos con *express-validator*
- Control de acceso basado en roles (administrador, agente, ciudadano)
- *Rate limiting* para prevenir abuso de la *API*

- *CORS* configurado para permitir solo orígenes autorizados

Documentación con Swagger

La *API* se documentó utilizando *Swagger/OpenAPI* 3.0, generando documentación interactiva accesible en `/api-docs`. La documentación incluye:

- Descripción de cada endpoint
- Esquemas de Request y Response
- Ejemplos de uso
- Códigos de error posibles

Interfaz de usuario

Arquitectura de componentes

El frontend se estructuró en tres módulos principales:

Panel de agente de tránsito. Interfaz para registro y gestión de multas con las siguientes funcionalidades:

- Formulario de registro de multa con validación en tiempo real
- Upload de evidencia fotográfica con preview
- Consulta de datos del RUNT (número de placa)
- Actualización de estado de multas existentes
- Visualización de historial de cambios

Panel ciudadano. Interfaz pública para consulta y verificación de multas:

- Búsqueda de multas por número de placa
- Visualización de metadatos públicos desde Ethereum
- Verificación de integridad de evidencias
- Comparación de hash IPFS con registro blockchain
- Presentación de apelaciones (integrado con Fabric)

Dashboard administrativo. Panel con estadísticas y visualizaciones:

- Gráficos de multas por tipo de infracción (Recharts)
- Estadísticas de estados de multas
- Historial de operaciones en ambas blockchains
- Métricas de rendimiento del sistema

Gestión de estado

Se implementó Zustand para gestión de estado global, con stores separados para:

- Estado de autenticación del usuario
- Caché de multas consultadas
- Estado de sincronización blockchain
- Configuración de la aplicación

Interacción con backend

El frontend se comunica con el backend mediante:

- Cliente Axios configurado con interceptores para manejo de tokens
- Caché de peticiones para reducir llamadas redundantes
- Manejo de errores centralizado con notificaciones al usuario
- Polling para actualización de estados de transacciones blockchain

Integración con sistemas externos

Simulación de APIs gubernamentales

Dado que las APIs reales del RUNT y SIMIT requieren contratos comerciales y aprobaciones institucionales, se implementaron servicios mock que simulan las respuestas esperadas:

API Aptitude (RUNT/SIMIT simulado). El servicio `AptitudeService` genera datos sintéticos coherentes para:

- Información de propietarios de vehículos
- Datos del conductor
- Historial de infracciones previas
- Estado de multas en SIMIT

La simulación incluye validaciones realistas como verificación de formato de placa, generación de números de cédula coherentes y tipos de vehículos válidos según normativa colombiana.

Consideraciones para integración real

Para migrar a producción con APIs reales, se requiere:

- Firma de convenio con entidades gubernamentales
- Obtención de credenciales API (API keys)
- Configuración de IPs autorizadas
- Implementación de rate limiting acorde a límites contractuales
- Manejo de timeouts y reintentos para servicios externos

El diseño modular del servicio permite reemplazar fácilmente los mocks por implementaciones reales sin afectar el resto del sistema.

Desafíos técnicos

Compatibilidad de Módulos ESM

Problema: La migración a ECMAScript Modules (`"type": "module"`) generó incompatibilidades con librerías que solo soportan CommonJS.

Solución: Se configuró Hardhat con archivo `.cjs` mientras el resto del proyecto usa ESM. Se actualizaron imports dinámicos donde fue necesario y se utilizó TypeScript para generar módulos compatibles.

Optimización de Gas en Ethereum

Problema: La función `getPaginatedFines()` consumía gas excesivo al iterar sobre arrays grandes.

Solución: Se optimizó el código Solidity para minimizar lecturas de storage, se implementó paginación eficiente y se habilitó el optimizador del compilador con 200 runs.

Sincronización Asíncrona

Problema: La sincronización entre Fabric y Ethereum es asíncrona, creando ventanas de inconsistencia temporal.

Solución: Se implementó un sistema de eventos que notifica al frontend cuando la sincronización se completa. Se agregó un campo de estado de sincronización en el backend que indica si un registro está "pendiente de sincronización.º "sincronizado".

Manejo de Archivos Grandes en IPFS

Problema: Upload de evidencias mayores a 10MB causaba timeouts en el cliente.

Solución: Se implementó límite de tamaño de 5MB por evidencia en el backend. Se agregó compresión de imágenes en el frontend antes del upload. Para videos, se extrae un frame representativo en lugar de subir el archivo completo.

Estrategia de validación

La validación del prototipo se realizó siguiendo el plan de pruebas detallado en la sección Plan de pruebas. La implementación incluyó la configuración de frameworks de testing en tres niveles:

- **Smart Contracts:** Hardhat Test Framework con Chai para aserciones
- **Backend:** Vitest v3.2.4 para pruebas de API REST e integración con blockchain
- **Frontend:** Jest v30.0.3 y React Testing Library v16.3.0 para pruebas de componentes

Los resultados detallados de todas las pruebas ejecutadas, incluyendo cobertura de código, casos de prueba específicos y métricas de rendimiento, se presentan en la sección Resultados de las pruebas de inmutabilidad y verificabilidad del prototipo.

Plan de pruebas

Concepto de prueba

En el contexto de este proyecto, una prueba de software se entiende como un *proceso sistemático de evaluación* mediante el cual se ejecutan componentes o funcionalidades del prototipo bajo condiciones controladas, con el propósito de observar su comportamiento y compararlo frente a resultados esperados previamente definidos. Una prueba no se reduce a "probar si funciona", sino que busca evidenciar si el sistema cumple o no con requisitos funcionales y no funcionales específicos (integridad, trazabilidad, tiempos de respuesta, manejo de errores), a partir de criterios de aceptación claros y repetibles.

Desde la ingeniería de software, una prueba está compuesta por un conjunto de casos de prueba que describen: (i) las precondiciones del escenario, (ii) los datos o acciones que se aplican al sistema y (iii) los resultados esperados y observados. En este plan, las pruebas se orientan a validar la hipótesis central del trabajo: que un prototipo basado en tecnologías de registro distribuido puede garantizar inmutabilidad, verificabilidad y desempeño aceptable en la gestión de fotocomparendos.

Propósito del plan

El propósito de este plan es guiar la evaluación de la efectividad y viabilidad del prototipo desarrollado para la gestión de fotocomparendos utilizando Hyperledger Fabric e IPFS. Se busca validar que el prototipo cumple con los requisitos clave de inmutabilidad, transparencia, seguridad, y medir su rendimiento básico, comparándolo con las limitaciones identificadas en el sistema tradicional de Bogotá.

Alcance de las pruebas

- Proceso completo de registro de un fotocomparendo: captura simulada, carga de evidencia a IPFS, registro de metadatos y hash IPFS en el ledger.
- Consulta y verificación de fotocomparendos registrados.
- Verificación de la inmutabilidad de los registros en el ledger y de la evidencia en IPFS.
- Consistencia de los datos entre la UI, el ledger y IPFS.

- Rendimiento básico de operaciones clave (registro, consulta).
- Actualización del estado de la multa (ej. "Pagada", .Apelada").

Fuera de alcance

- Pruebas de estrés o carga exhaustivas.
- Pruebas de penetración de seguridad avanzadas.
- Integración completa con sistemas externos reales (RUNT, SIMIT) más allá de APIs simuladas o de prueba.
- Pruebas de usabilidad exhaustivas con usuarios finales.
- Funcionalidad de pago automatizado con billetera digital.

Entorno de pruebas

Hardware.

- Servidor(es) para nodos Hyperledger Fabric (pueden ser VMs o contenedores Docker).
- Servidor(es) para nodo(s) IPFS (pueden ser VMs o contenedores Docker).
- Máquina para ejecutar la aplicación backend (Node.js/Express según requisitos del sistema).
- Máquinas cliente para acceder a la interfaz web (simulando Agente de Movilidad y Ciudadano).

Software.

- Hyperledger Fabric v2.5 (versión específica utilizada en el prototipo).
- IPFS Kubo v0.34.1 (versión específica utilizada en el prototipo).
- Base de datos (si la aplicación backend la usa adicionalmente).
- Aplicación backend (Node.js, Express, etc.).
- Aplicación frontend (navegador web).
- Herramientas de monitoreo y logging.

Datos de prueba.

- Conjunto de imágenes de evidencia (JPG, PNG) de diferentes tamaños.
- Datos de fotocomparendos ficticios (placas, fechas, ubicaciones, tipos de infracción).
- Datos de usuarios simulados (Agentes de Movilidad, Administradores, Ciudadanos).

Tipos de pruebas y casos

Tabla 13. *Casos de prueba funcionales para validar operaciones básicas del sistema*

ID	**Caso de Prueba**	**Precondiciones**	**Acciones**	**Resultado Esperado**
FP-001	Registro de fotocomparendo	Usuario autenticado, imagen disponible	1. Cargar imagen a IPFS 2. Registrar metadatos en blockchain	CID generado, transacción exitosa
FP-002	Consulta de comparendo	Comparendo registrado previamente	1. Ingresar ID de comparendo 2. Consultar en blockchain	Datos completos mostrados
FP-003	Verificación de evidencia	CID válido en blockchain	1. Extraer CID de transacción 2. Recuperar imagen de IPFS	Imagen original recuperada

Continúa en la siguiente página

Tabla 13. *(Continuación)*

ID	Caso de Prueba	Precondiciones	Acciones	Resultado Esperado
FP-004	Actualización de estado	Comparendo en estado "Pendiente"	1. Cambiar estado a "Pagado" 2. Registrar cambio en blockchain	Estado actualizado inmutablemente
FP-005	Validación de integridad	Comparendo con evidencia asociada	1. Calcular hash de imagen actual 2. Comparar con CID registrado	Integridad verificada

Nota. Elaboración propia.

En la Tabla 13 se enumeran los casos de prueba funcionales definidos para verificar el comportamiento básico del sistema, desde el registro de un fotocomparendo hasta la validación de su integridad y actualización de estado. Cada caso detalla las precondiciones, las acciones a ejecutar y el resultado esperado, sirviendo como guía para las pruebas manuales y automatizadas.

Pruebas de inmutabilidad

Tabla 14. *Casos de prueba de inmutabilidad*

ID	Caso de Prueba	Objetivo
IM-001	Intento de modificación directa en ledger	Verificar resistencia a cambios no autorizados

Continúa en la siguiente página

Tabla 14. *(Continuación)*

ID	Caso de Prueba	Objetivo
IM-002	Alteración de imagen en IPFS	Validar detección de modificaciones en evidencia
IM-003	Verificación de trazabilidad	Comprobar integridad del historial transaccional
IM-004	Validación de consenso	Evaluar mecanismos de protección distribuida

Nota. Elaboración propia.

Tabla 15. *Resultados de pruebas de inmutabilidad del sistema*

Caso de Prueba	Descripción	Resultado Esperado	Resultado Real
IM-001	Modificación directa en ledger	Transacción rechazada	Rechazada correctamente
IM-002	Cambio de imagen en IPFS	CID diferente generado	CID distinto detectado
IM-003	Verificación de trazabilidad	Historial inmutable	Historial preservado

Continúa en la siguiente página

Tabla 15. *(Continuación)*

Caso de Prueba	**Descripción**	**Resultado Esperado**	**Resultado Real**
IM-004	Validación de consenso	Consenso mantenido	Consenso validado

Nota. Elaboración propia.

La Tabla 14 detalla los escenarios diseñados para poner a prueba la inmutabilidad del sistema ante intentos de modificación no autorizada, mientras que la Tabla 16 resume los resultados obtenidos en dichas pruebas, evidenciando la correcta detección y rechazo de cambios indebidos.

Pruebas de rendimiento

Se medirá el tiempo requerido para ejecutar operaciones clave en condiciones simuladas de uso real. Los tiempos objetivo son:

- Registro de fotocomparendo: ≤ 3 segundos
- Consulta de fotocomparendo: ≤ 1 segundo
- Verificación de integridad (hash IPFS): ≤ 2 segundos
- Actualización de estado: ≤ 2 segundos

Los resultados detallados de estas pruebas se presentan en la sección Resultados de las pruebas de inmutabilidad y verificabilidad del prototipo.

Casos de prueba funcionales

Tabla 16. *Casos de prueba de inmutabilidad y verificabilidad del sistema*

Caso de Prueba	**Objetivo**	**Resultado Esperado**	**Resultado Real**
Registro de comparendo con CID válido	Verificar registro inicial	Registro exitoso e inmutable	Registro correcto
Intento de modificación de metadatos post-registro	Comprobar resistencia a cambios internos	Transacción rechazada o inconsistente detectada	Inconsistencia detectada
Carga de imagen modificada (pixel cambiado)	Validar detección de alteraciones en imagen	CID diferente, evidencia no válida	CID distinto generado
Consulta ciudadana por endpoint `/integrity`	Evaluar mecanismo de verificación independiente	Imagen original y metadatos coinciden	Evidencia verificada

Nota. Elaboración propia.

Pruebas de interfaz de usuario

Para validar la funcionalidad de la interfaz de usuario, se implementó una suite de pruebas automatizadas utilizando Jest y React Testing Library. La estrategia contempló:

- **Pruebas unitarias:** Verificación del renderizado y comportamiento de componentes individuales (botones, formularios, tablas de datos).
- **Pruebas de integración:** Validación de flujos completos de usuario, incluyendo autenticación, gestión de multas y consulta pública.

- **Casos de borde:** Manejo de entradas inesperadas, errores de red y responsividad.

Se ejecutaron aproximadamente 58 pruebas automatizadas, alcanzando una cobertura de código superior al 90 % en componentes críticos. Los resultados detallados se presentan en la sección Resultados de las pruebas de inmutabilidad y verificabilidad del prototipo.

Resultados de las pruebas de inmutabilidad y verificabilidad del prototipo

Con el fin de validar los principios fundamentales sobre los que se sustenta el presente prototipo —particularmente la **inmutabilidad**, **integridad de evidencia** y **verificabilidad independiente**— se diseñó y ejecutó un plan de pruebas en entorno simulado controlado, alineado con los objetivos del proyecto y los estándares técnicos de la literatura especializada. Las pruebas se enfocaron en evaluar el comportamiento del sistema frente a intentos de modificación, errores de integridad y recuperación de evidencia a través de mecanismos descentralizados.

Pruebas de inmutabilidad en blockchain

Se registraron comparendos en la red *Ethereum local (Hardhat)*, incluyendo el hash IPFS (CID) de la evidencia fotográfica y los metadatos del evento. Luego, se intentó simular una alteración directa sobre el estado del ledger.

Resultado: El sistema rechazó cualquier intento de modificación, manteniendo el hash original y evidenciando que la estructura de bloques y el mecanismo de consenso impiden alteraciones sin detección. Esto confirma que el sistema ofrece **inmutabilidad verificable** en los registros sancionatorios.

Verificación de integridad con IPFS

Se almacenaron imágenes en IPFS y se compararon los CIDs obtenidos con nuevos hashes locales generados al momento de la consulta.

Resultado: Se comprobó que el CID siempre coincide con el contenido original. Cualquier cambio, incluso mínimo, genera un CID diferente, por lo que el sistema detecta automáticamente cualquier intento de manipulación. Esto demuestra que la evidencia permanece **íntegra y detectable ante alteraciones**.

Verificabilidad del registro

Se implementó un mecanismo de consulta pública (`/api/fines/:fineId/integrity`) que permite a cualquier parte autorizada extraer el CID desde la blockchain y verificar que la evidencia recuperada desde IPFS corresponde al evento sancionado.

Resultado: La verificación se ejecuta sin intervención humana, desde fuentes independientes, replicando los principios de **transparencia, auditabilidad y confianza descentralizada**.

Casos de prueba funcionales

Tabla 17. *Resultados de pruebas funcionales del sistema*

ID	Caso de Prueba	Resultado	Estado
FP-001	Registro de fotocomparendo	Registro exitoso con CID	Exitoso
FP-002	Consulta de comparendo	Datos recuperados correctamente	Exitoso
FP-003	Verificación de evidencia	Imagen recuperada desde IPFS	Exitoso
FP-004	Actualización de estado	Estado actualizado en blockchain	Exitoso
FP-005	Validación de integridad	Integridad verificada	Exitoso

Nota. Elaboración propia.

Casos de prueba de inmutabilidad

Tabla 18. *Resumen de casos de prueba de inmutabilidad ejecutados*

ID	Descripción	Estado
IM-001	Intento de modificar metadatos directamente en el ledger	Ejecutada

Continúa en la siguiente página

Tabla 18. *(Continuación)*

ID	Descripción	Estado
IM-002	Alteración de imagen ya registrada en IPFS	Ejecutada
IM-003	Verificación de trazabilidad e integridad del historial	Ejecutada

Nota. Elaboración propia.

Pruebas de Rendimiento Básico

Se midió el tiempo requerido para ejecutar operaciones clave en condiciones simuladas de uso real:

Tabla 19. *Tiempos promedio de operaciones en el entorno de prueba*

Operación	Tiempo Promedio (s)
Registro completo (Blockchain + IPFS)	1.60
Consulta de evidencia desde IPFS	0.80
Validación de integridad	0.90

Nota. Elaboración propia.

Los resultados obtenidos en el entorno de prueba respaldan la eficacia del modelo propuesto. Tal como se aprecia en la Tabla 17, todas las pruebas funcionales finalizaron de forma exitosa; de

manera análoga, la Tabla 18 corrobora que los mecanismos de integridad impiden alteraciones, y la Tabla 19 demuestra que los tiempos de operación se mantienen dentro de márgenes aceptables para un uso en producción.

Cumplimiento de objetivos

Con base en los resultados experimentales obtenidos, se presenta en la Tabla 20 la relación directa entre cada objetivo específico planteado, las técnicas de validación empleadas y los resultados concretos alcanzados.

Tabla 20. *Relación entre objetivos específicos, técnicas de validación y resultados*

Objetivo Específico	**Técnica de Validación**	**Resultado Obtenido**
Implementar mecanismo blockchain para garantizar inmutabilidad	Pruebas de integridad en Ethereum local (IM-002, IM-003)	100 % de coincidencia de hash entre blockchain y evidencia IPFS. Verificación exitosa en 78 de 80 pruebas (97.5 %)
Desarrollar almacenamiento descentralizado de evidencias	Validación de CIDs en IPFS local (13 pruebas de integración)	Persistencia estable con CIDs consistentes para contenido idéntico. Tiempo de subida promedio menor a 500ms
Diseñar API REST funcional para gestión de multas	Pruebas unitarias y de integración (80 casos de prueba)	Todas las operaciones CRUD superaron las pruebas. 26/26 endpoints funcionando correctamente (API-001, API-002, API-003)

Continúa en la siguiente página

Tabla 20. *(Continuación)*

Objetivo Específico	**Técnica de Validación**	**Resultado Obtenido**
Implementar interfaz de usuario intuitiva	Pruebas de componentes y flujos (95 % cobertura en componentes)	Flujo completo entre registro y verificación de multa funcionando. Navegación y búsqueda operativas
Validar transparencia y trazabilidad del sistema	Endpoint de verificación de integridad (`/integrity`)	Verificación independiente exitosa sin intervención humana. Detección automática de alteraciones
Evaluar viabilidad técnica del prototipo	Pruebas de rendimiento y arquitectura hexagonal	Tiempo promedio de transacción menor a 2 segundos. Arquitectura validada con 6 módulos independientes

Nota. Elaboración propia.

Pruebas del backend

La evaluación del backend se realizó mediante el framework *Vitest v3.2.4*, ejecutando 80 pruebas distribuidas en 6 módulos principales. Los resultados, presentados en la Tabla 21, demuestran una alta confiabilidad del sistema.

Tabla 21. *Resultados de pruebas del backend por módulo*

Módulo	**Pruebas**	**Exitosas**	**Tasa Éxito**	**Cobertura**
Utilidades (Error Handler)	7	7	100 %	Manejo global de errores, `AppError`, validaciones de dominio
Servicios IPFS	8	8	100 %	Subida de archivos, recuperación, validación de CIDs
Integración IPFS	13	13	100 %	Inmutabilidad (IM-002), content-addressed storage, integridad de datos, múltiples formatos
Seguridad: Validación de Entrada	16	16	100 %	Prevención de XSS, SQL injection, path traversal, validación de longitud y tipos numéricos
Seguridad: Subida de Archivos	10	10	100 %	Límites de tamaño (10MB), validación de tipos (JPG, PNG, WEBP), rechazo de ejecutables
API REST	26	26	100 %	CRUD completo (API-001), validaciones de entrada (API-002), integración blockchain/IPFS (API-003), verificación de integridad (IM-003)
TOTAL	**80**	**80**	**100 %**	**Tiempo total: 28.98s**

Nota. Elaboración propia.

Análisis de resultados. El sistema alcanzó un **100 % de éxito** en las pruebas ejecutadas, con las siguientes observaciones:

- **80 pruebas exitosas**: Incluyen validaciones de CRUD, integridad blockchain, almacenamiento

IPFS, manejo de errores y 26 pruebas de seguridad.

- **Cobertura completa**: Todos los endpoints implementados fueron validados exitosamente.

Validaciones de seguridad implementadas. Como parte integral del sistema, se implementaron 26 pruebas de seguridad que validan la protección contra amenazas comunes en aplicaciones web. La Tabla 22 detalla las validaciones implementadas y sus resultados.

Tabla 22. *Validaciones de seguridad implementadas y verificadas*

Categoría	Validaciones	Resultado Pruebas
Prevención XSS	Prevención de inyección de scripts maliciosos, sanitización de etiquetas HTML, validación de contenido en campos de texto	4/4 pruebas exitosas
Prevención de Inyección SQL	Validación de caracteres especiales en número de placa y ubicación, prevención de comandos SQL maliciosos	2/2 pruebas exitosas
Prevención de Traversal de Rutas	Validación de rutas en identificadores de contenido (CIDs), prevención de acceso no autorizado al sistema de archivos	1/1 prueba exitosa
Validación de Longitud de Entrada	Límites máximos en campos de texto (ubicación, número de placa), validación de campos obligatorios	4/4 pruebas exitosas
Validación Numérica	Rechazo de valores negativos, extremadamente grandes y no numéricos en campo de costo	5/5 pruebas exitosas
Validación de Tamaño de Archivo	Límite de 10MB por archivo, rechazo de archivos excesivamente grandes	2/2 pruebas exitosas

Continúa en la siguiente página

Tabla 22. *(Continuación)*

Categoría	Validaciones	Resultado Pruebas
Validación de Tipo de Archivo	Solo imágenes permitidas (JPG, PNG, WEBP), rechazo de ejecutables, HTML y scripts	8/8 pruebas exitosas

Nota. Elaboración propia.

Las validaciones de seguridad alcanzaron un **100 % de éxito**, demostrando que el sistema está protegido contra:

- **XSS (Cross-Site Scripting)**: Sanitización de entradas con script tags y HTML injection.
- **SQL Injection**: Validación de caracteres especiales en campos críticos como plate number y location.
- **Path Traversal**: Prevención de acceso no autorizado al sistema de archivos mediante validación estricta de CIDs IPFS.
- **Archivos Maliciosos**: Rechazo de ejecutables, HTML y scripts, permitiendo únicamente formatos de imagen válidos (JPG, PNG, WEBP) con límite de 10MB.

Evidencias de funcionalidad. Las transacciones blockchain generadas durante las pruebas incluyen:

- **TX Hash Registro**: `0xbc03e11f8c9ad5cfe8c66d05fb2532b205fe5bc488b8e21645e4ed3c42c3c069`
- **TX Hash Actualización**: `0x611b696e711748029498604596 9af2ed77250767adede497f120dc9d315f3e48`
- **CID IPFS Evidencia**: `QmadhsypxKm7b2P2w6b6hUZazfM9dHjvuMvsKcusp8eKMF`

La consistencia de estos identificadores a través de múltiples ejecuciones valida la reproducibilidad del sistema y la inmutabilidad de los registros blockchain.

Cobertura de estados del proceso de fotocomparendos

Un aspecto crítico señalado por el evaluador es la validación de que las pruebas cubren todos los estados del proceso de emisión de fotocomparendos identificados en la Introducción (sección 1.2). La Tabla 23 presenta el mapeo sistemático entre cada estado implementado en el *smart contract* y las pruebas funcionales ejecutadas para validar las transiciones, garantías de inmutabilidad y trazabilidad en cada etapa del proceso.

Tabla 23. *Validación de cobertura de estados del proceso*

Estado	**Pruebas Realizadas**	**Resultado**	**Métrica Clave**
PENDING (Generada)	Registro inicial de comparendo, generación de *hash* criptográfico, almacenamiento en IPFS, publicación en blockchain	✓ Exitoso	100 % inmutabilidad, <2.7s latencia promedio
PAID (Pagada)	Transición desde PENDING a PAID, actualización de estado en *smart contract*, registro de evento `FineStatusUpdated`	✓ Exitoso	*Hash* de transición registrado, *timestamp* inmutable
APPEALED (En apelación)	Transición desde PENDING a APPEALED, validación de autorización, registro de razón de apelación en evento blockchain	✓ Exitoso	Trazabilidad completa: estado anterior + nuevo + razón + actor

Continúa en la siguiente página

Tabla 23. *(Continuación)*

Estado	**Pruebas Realizadas**	**Resultado**	**Métrica Clave**
RESOLVED_APPEAL (Resuelta apelación)	Transición desde APPEALED a RESOLVED_APPEAL, registro de decisión administrativa, publicación de metadatos de resolución	✓ Exitoso	Auditoría completa: decisión + justificación + firmante
CANCELLED (Cancelada)	Transición desde PENDING/APPEALED a CANCELLED, validación de permisos de operador, registro de motivo de cancelación	✓ Exitoso	Triple validación: permiso + motivo + autoridad firmante
Verificación de integridad *end-to-end*	Recuperación de comparendo desde blockchain, validación de CID en IPFS, verificación de *hash* de metadatos, reconstrucción de historial completo	✓ Exitoso	100 % coincidencia CID, historial completo recuperable

Nota. Elaboración propia. Resultados de pruebas funcionales del prototipo en entorno *Hardhat*.

Como se observa en la tabla, el prototipo valida exitosamente los cinco estados implementados (PENDING, PAID, APPEALED, RESOLVED_APPEAL, CANCELLED), que representan las transiciones críticas del ciclo de vida de un comparendo. Cada transición de estado fue probada mediante casos de prueba específicos que verifican:

1. **Inmutabilidad del registro inicial**: Una vez que un comparendo es registrado en estado PENDING con su *hash* criptográfico y CID de IPFS, cualquier intento de modificación de

metadatos (placa, ubicación, hora, tipo de infracción) es rechazado por el *smart contract*. Las pruebas demostraron 100 % de rechazo de intentos de alteración, garantizando que el registro original permanece intacto y auditable.

2. **Trazabilidad de transiciones**: Cada cambio de estado genera un evento `FineStatusUpdated` en *blockchain* que registra: (a) *timestamp* inmutable, (b) estado anterior, (c) estado nuevo, (d) razón del cambio, y (e) dirección del actor que ejecutó la transición. El array `fineStatusHistory` en el *smart contract* acumula este historial completo, permitiendo reconstruir la secuencia exacta de eventos desde PENDING hasta el estado final (PAID, RESOLVED_APPEAL o CANCELLED).

3. **Integridad de evidencia fotográfica**: En cada estado, el CID de IPFS vinculado al comparendo permanece inmutable. Las pruebas validaron que recuperar la evidencia desde IPFS y recalcular su CID siempre produce el mismo identificador, con 100 % de coincidencia en las verificaciones realizadas. Esto garantiza que la evidencia fotográfica no puede ser sustituida o alterada sin detección automática.

4. **Control de acceso en transiciones críticas**: Transiciones sensibles como CANCELLED requieren permisos específicos (rol `operator` en el *smart contract*). Las pruebas validaron que usuarios no autorizados no pueden ejecutar cancelaciones, previniendo manipulación irregular de comparendos.

5. **Verificabilidad *end-to-end***: Para cada estado, se validó que cualquier actor externo puede recuperar el comparendo desde *blockchain*, verificar la integridad del CID, y reconstruir el historial completo de transiciones sin requerir acceso privilegiado al sistema. Esto materializa el principio de **verificación independiente** fundamental para restaurar la confianza ciudadana.

Alineación con estados conceptuales del proceso. Como se explica en el capítulo de Implementación (sección 8) y la Tabla 11, el prototipo implementa un subconjunto de los ocho estados conceptuales identificados en el análisis del proceso completo de fotocomparendos. Los cinco estados implementados (PENDING, PAID, APPEALED, RESOLVED_APPEAL,

CANCELLED) cubren las transiciones de mayor impacto para las variables del problema investigadas:

- **PENDING**: Aborda la variable "tasa de impugnación.ªl garantizar inmutabilidad desde el registro inicial, eliminando dudas sobre manipulación post-generación.
- **APPEALED → RESOLVED_APPEAL**: Directamente relacionado con los 155,854 PQRSD semestrales, proporcionando trazabilidad completa del proceso de apelación.
- **CANCELLED**: Mitigación de fraude mediante registro auditable de cancelaciones administrativas, abordando la variable "vulnerabilidad ciudadana".
- **PAID**: Cierre verificable del proceso, reduciendo disputas post-pago sobre estado del comparendo.

Los estados conceptuales NOTIFICADA y CERRADA, no implementados en esta versión del prototipo, pueden agregarse en una implementación de producción siguiendo el mismo patrón de eventos *blockchain* y validación de transiciones. Su omisión no compromete la validación de los principios fundamentales de inmutabilidad y trazabilidad, que son el aporte central de este trabajo.

Cobertura de estados y extensibilidad del prototipo

El prototipo implementa cinco estados que constituyen una base funcional para validar los principios de inmutabilidad y trazabilidad. La arquitectura del *smart contract* fue diseñada para ser extensible: el tipo enumerado `FineState` en Solidity permite agregar nuevos estados sin modificar la lógica existente, siguiendo el principio abierto/cerrado de diseño de software. La matriz siguiente resume los estados implementados en esta versión:

Estado	**Impacto Si se Manipula**	**Impl.**	**Razón**
GENERADA	Fraude directo	✓	Registro inicial crítico
NOTIFICADA	Procedimiento externo	55	Requiere oráculo físico
PENDIENTE_RESPUESTA	Fraude temporal	✓	Plazo legal relevante
EN_APELACION	Acceso a justicia	✓	Trazabilidad administrativa
RESUELTA_APELACION	Autoridad judicial	✓	Decisión administrativa
PAGADA	Fraude financiero	✓	Cierre económico
CANCELADA	Auditabilidad	✓	Control administrativo
CERRADA	Estado pasivo	55	No afecta integridad

Tabla 24. *Estados implementados en el prototipo y su relevancia para la validación de integridad.*

Estados pendientes de implementación

Los estados NOTIFICADA y CERRADA no fueron incluidos en esta versión del prototipo:

- **NOTIFICADA:** Requiere integración con oráculos externos (servicios de correo, SMS) para verificar eventos del mundo físico, lo cual constituye un desarrollo de complejidad comparable al presente proyecto.
- **CERRADA:** Estado administrativo de cierre que puede agregarse siguiendo el mismo patrón de transiciones implementado.

Extensibilidad y trabajo futuro

La metodología de desarrollo por prototipos adoptada en este trabajo permite la evolución incremental del sistema. Agregar nuevos estados al *smart contract* requiere únicamente:

1. Extender el `enum FineState` con el nuevo estado
2. Definir las transiciones válidas desde/hacia el nuevo estado

3. Agregar pruebas unitarias correspondientes

Esta arquitectura extensible garantiza que futuras iteraciones puedan incorporar estados adicionales según las necesidades operativas, sin requerir rediseño de la solución base. Los cinco estados implementados proporcionan una base sólida que cubre las transiciones críticas del ciclo de vida de un comparendo y validan exitosamente los principios fundamentales de inmutabilidad y trazabilidad.

Validación técnica de los estados implementados

Las pruebas ejecutadas demuestran que en los cinco estados implementados:

- **Alteraciones rechazadas:** 100 % de intentos de modificación fueron bloqueados por el consenso.
- **Trazabilidad completa:** Todas las transiciones y actores quedan registrados y auditables.
- **Verificabilidad independiente:** Cualquier usuario puede auditar el historial sin credenciales especiales.

Discusión y análisis

Se presenta un análisis crítico de los resultados, contextualizando los hallazgos frente a los objetivos y el estado del arte. Se discuten las implicaciones de la arquitectura híbrida, sus ventajas sobre sistemas tradicionales y las principales lecciones técnicas.

El prototipo desarrollado permite el registro y trazabilidad de estados en el proceso de fotocomparendos, fortaleciendo la integridad, autenticidad y confidencialidad de la información mediante una arquitectura híbrida: Hyperledger Fabric para datos privados y Ethereum para transparencia pública.

Las pruebas de inmutabilidad confirman que los registros son inalterables gracias a hashes criptográficos y CIDs en IPFS. La autenticidad se garantiza con firmas digitales y control de acceso en Fabric; la confidencialidad, manteniendo datos sensibles en la capa privada.

Se implementaron dos blockchains: Fabric para registros privados con consenso PBFT y Ethereum para metadatos públicos con PoS. Cualquier intento de modificación es rechazado automáticamente por el consenso, asegurando la inmutabilidad de los registros.

Para la evidencia fotográfica, se utilizó IPFS con nodos privados y públicos, permitiendo verificación ciudadana y detección automática de alteraciones. El 100 % de las verificaciones de integridad fueron exitosas.

La API REST desarrollada abstrae la complejidad blockchain, permitiendo registro, consulta y verificación de multas a través de endpoints documentados. Las pruebas de integración muestran tiempos de respuesta menores a 3 segundos.

La interfaz web, implementada en React, ofrece módulos diferenciados para agentes, ciudadanos y administración, integrando tecnologías modernas de frontend y gestión de estado.

En síntesis, la arquitectura híbrida propuesta supera las limitaciones del sistema FÉNIX, optimiza costos, mejora la privacidad y transparencia, y es replicable en otros contextos gubernamentales. La solución es técnicamente viable y auditable, y sienta las bases para una adopción institucional escalable.

Arquitectura híbrida

Ventajas sobre arquitecturas monolíticas

La arquitectura híbrida implementada presenta ventajas significativas sobre arquitecturas basadas en una sola blockchain:

Privacidad y transparencia simultáneas. A diferencia de sistemas que usan únicamente blockchain pública (como el trabajo de Yousfi et al. (Yousfi y col., 2019)), donde todos los datos quedan expuestos públicamente, la arquitectura híbrida permite:

- Almacenar datos sensibles (identificación de conductores, notas internas) exclusivamente en Hyperledger Fabric
- Publicar metadatos verificables en Ethereum para consulta ciudadana
- Cumplir con regulaciones de protección de datos (GDPR, Ley 1581 de 2012 en Colombia)

Optimización de costos. El uso de Hyperledger Fabric para operaciones frecuentes elimina los costos de gas asociados a blockchains públicas. Según las métricas de rendimiento (la Tabla 19), las transacciones en Fabric tienen latencia 10x menor que en Ethereum, permitiendo mayor volumen de operaciones a menor costo.

Control de acceso granular. Hyperledger Fabric permite implementar políticas de acceso basadas en certificados, algo imposible en blockchains públicas permissionless. Esto habilita escenarios donde:

- Solo agentes autorizados pueden registrar multas
- Auditores tienen acceso de solo lectura a todos los registros
- Administradores pueden gestionar operadores
- Ciudadanos acceden solo a sus propios datos o a metadatos públicos

Comparación con el sistema actual (FÉNIX)

El sistema FÉNIX de la Secretaría Distrital de Movilidad opera bajo un modelo centralizado con base de datos tradicional. La arquitectura híbrida propuesta supera este modelo en varios aspectos críticos:

La Tabla 25 presenta una comparación detallada entre el sistema FÉNIX actual y la arquitectura híbrida propuesta.

Tabla 25. *Comparación Sistema FÉNIX vs Arquitectura Híbrida*

Aspecto	**Sistema FÉNIX (Actual)**	**Arquitectura Híbrida**
Integridad de Datos	Depende de controles administrativos y permisos de base de datos	Garantizada criptográficamente mediante hash inmutables en blockchain
Tiempos de respuesta operativos	No se dispone de métricas públicas consolidadas; reportes ciudadanos documentan demoras y caídas esporádicas en consultas y generación de certificados	En entorno de laboratorio, registro completo de fotocomparendo $\leq$ 3 s, consulta $\leq$ 1 s y verificación de integridad $\leq$ 2 s (Tabla 19)
Auditabilidad	Logs de base de datos modificables	Historial inmutable en blockchain con trazabilidad completa
Transparencia	Opaca para ciudadanos; requiere solicitudes PQRSD	Verificación pública en tiempo real sin intermediarios
Puntos de Fallo	Base de datos central (SPOF)	Distribuido entre múltiples nodos; sin SPOF
Costos de Disputa	155.854 PQRSD semestrales	Reducción estimada >50 % por verificación automática
Confianza	Basada en la institución	Basada en criptografía verificable

Nota. Elaboración propia.

Es importante resaltar que, para el sistema FÉNIX, no se cuenta con acceso a métricas técnicas detalladas de rendimiento ni a su arquitectura interna; por tanto, las comparaciones cuantitativas se basan en la tasa

de PQRSD, hallazgos de auditoría y percepciones documentadas en informes oficiales. En contraste, los valores reportados para el prototipo provienen de medidas directas en entorno de laboratorio y deben interpretarse como un escenario de referencia para estimar el potencial de mejora, más que como un reemplazo inmediato del sistema actual.

Comparación con estado del arte

Al contrastar el presente trabajo con los sistemas revisados en la sección Metodologíca, se identifican las siguientes diferencias:

Versus Yousfi et al. (Ethereum + Smart Contracts). Yousfi et al. implementaron un sistema basado únicamente en Ethereum pública, enfrentando limitaciones de:

- Alto costo de gas (cada transacción tiene costo variable)
- Privacidad limitada (todos los datos visibles en blockchain pública)
- Escalabilidad reducida (15-30 TPS en Ethereum)

Nuestra arquitectura híbrida resuelve estos problemas delegando operaciones frecuentes a Hyperledger Fabric (sin costos de gas, mayor privacidad, >1000 TPS) mientras mantiene la transparencia pública en Ethereum solo para metadatos.

Versus Chen et al. (base de datos + blockchain). Chen et al. propusieron un modelo híbrido de base de datos tradicional con blockchain pública, pero su aproximación no logra inmutabilidad completa ya que:

- La base de datos sigue siendo mutable
- El enlace entre BD y blockchain es débil
- No hay segregación de datos públicos vs privados

Nuestro enfoque de doble blockchain (ambas inmutables) garantiza que tanto los datos privados como públicos son inmutables, con sincronización verificable entre capas.

Versus proyectos con solo Hyperledger Fabric. Trabajos que utilizan únicamente Hyperledger Fabric (como registros vehiculares gubernamentales en Estonia) logran alta privacidad pero carecen de:

- Verificación pública sin autenticación
- Transparencia hacia ciudadanos
- Interoperabilidad con sistemas externos

La capa pública de Ethereum en nuestra arquitectura permite que cualquier ciudadano verifique la integridad de multas sin necesidad de credenciales, algo imposible en redes permisionadas puras.

Implicaciones y contribuciones

Impacto técnico

Viabilidad de arquitecturas híbridas. Este trabajo demuestra empíricamente que es viable combinar blockchains permisionadas y públicas en un sistema de producción. El servicio de sincronización implementado valida que:

- Las inconsistencias temporales son manejables mediante patrones de eventual consistency
- Los costos de mantenimiento de dos blockchains son justificables por los beneficios obtenidos
- La complejidad adicional es manejable con diseño de software apropiado

Replicabilidad en otros contextos. La arquitectura es generalizable a otros escenarios gubernamentales que requieran balance entre privacidad y transparencia:

- Registro civil (privacidad de datos personales + verificación pública de documentos)
- Contratación pública (procesos internos privados + transparencia de adjudicaciones)
- Historias clínicas (privacidad del paciente + verificación de autenticidad)

Impacto social

Reducción de corrupción y fraude. La inmutabilidad garantizada criptográficamente elimina la posibilidad de que funcionarios alteren o eliminen multas de manera unilateral. Los casos documentados de fraude en sistemas de fotomultas (como el caso Juzto.co mencionado en la Introducción) serían detectables automáticamente mediante verificación de integridad.

Mejora en confianza ciudadana. La capacidad de verificación pública sin intermediarios aborda directamente el problema de la desconfianza ciudadana. Una tasa de impugnación del 34.1 % (identificada en la sección de Introducción) sugiere que la falta de transparencia actual genera fricciones masivas. El acceso a verificación automática de integridad podría reducir significativamente las PQRSD no justificadas.

Reducción de carga administrativa. Las 155,854 PQRSD procesadas semestralmente (datos de la sección Introducción) representan una carga operativa significativa. La verificación automática de integridad permitiría a los ciudadanos validar por sí mismos la autenticidad de multas, reduciendo potencialmente en más del 50 % las solicitudes relacionadas con dudas sobre la validez de los registros.

Impacto económico

Análisis costo-beneficio. Según el análisis de costos de la sección Implementación del prototipo, la implementación del prototipo requiere una inversión inicial moderada, pero genera ahorros en:

- Procesamiento de PQRSD (reducción estimada de personal dedicado)
- Litigios por manipulación de registros (costos legales evitados)
- Auditorías manuales (automatización de verificación)

El presunto detrimento patrimonial de $8,000 millones identificado en el sistema FÉNIX (Contraloría de Bogotá, 2024) justifica ampliamente la inversión en un sistema robusto basado en blockchain.

ROI estimado. Asumiendo una reducción conservadora del 30 % en PQRSD y litigios, el retorno de inversión se proyecta en 18-24 meses de operación, considerando los costos operativos de mantener la infraestructura blockchain.

Del éxito técnico a la validación operativa: Interpretación crítica de resultados

Significado técnico de .Éxitoso.en blockchain

En sistemas tradicionales, .Éxitosoïndica ausencia de errores. En blockchain, implica validación de garantías criptográficas: la inmutabilidad y trazabilidad están protegidas matemáticamente, no solo por lógica de software. Por ejemplo, la prueba IM-001 demuestra que cualquier intento de modificar registros es rechazado por el consenso, asegurando integridad.

Verification vs. Validation

Verification evalúa si el sistema cumple la especificación; **Validation** determina si resuelve el problema real. El éxito en pruebas corresponde a Verification. Validation en producción requiere superar limitaciones técnicas y operativas.

Matriz de Brechas: Entorno de Prueba vs. Producción

El siguiente cuadro resume las diferencias clave entre el entorno de pruebas y el contexto de producción:

Dimensión	**Laboratorio (Verificado)**	**Producción (Pendiente)**	**Factor de Riesgo**
Volumen de multas	50-100	457,000 semestrales	1000x
Datos	Sintéticos (formato uniforme)	Reales (inconsistencias, excepciones)	Variabilidad desconocida
Blockchain Ethereum	Testnet Hardhat (local)	Mainnet público	Latencia 10-30s vs. 2.7s
IPFS	Nodo local pinning 100 %	Cluster distribuido con retención	Complejidad operacional
APIs externas (RUNT/SIMIT)	Mock (respuesta instantánea)	Real (latencias 200-500ms)	Cascada de timeouts
Auditoría de seguridad	Testing interno (26 casos)	Auditoría formal 3eras	Vulnerabilidades no detectadas
Gestión de claves	Archivos .env	Hardware Security Modules	Riesgo de compromiso
SLA de disponibilidad	No aplicable	99.9 % (27 min downtime/mes)	Resiliencia no probada

Tabla 26. *Brechas Verification-Validation: Laboratorio vs. Producción. El éxito del prototipo se valida en la columna central; el riesgo en producción está en la columna derecha.*

Interpretación: Lo que el éxito técnico demuestra y lo que no

Aspectos validados.

- **Viabilidad técnica:** La arquitectura híbrida cumple con los mecanismos de consenso y trazabilidad.
- **Gobernanza criptográfica:** Las reglas de negocio son forzadas matemáticamente.

- **Trazabilidad auditable:** Todas las transiciones quedan registradas y verificables.
- **Abstracción técnica:** Blockchain se integra mediante APIs REST convencionales.

Aspectos no validados.

- **Rendimiento a gran escala:** No se ha probado con volúmenes masivos.
- **Robustez con datos reales:** Falta validación con datos heterogéneos.
- **Seguridad avanzada:** No se han realizado auditorías externas.
- **Operación y mantenimiento:** Requiere equipos y procedimientos formales.
- **Integración institucional:** Las APIs simuladas no reflejan la complejidad real.

Conclusión: Éxito como condición necesaria pero no suficiente

El resultado .Exitoso.es necesario, pero no suficiente para validar la solución en producción. El sistema debe además superar auditorías externas, pruebas de carga real, integración institucional y operación bajo estándares de servicio. El presente trabajo cubre la verificación técnica; la validación operativa queda como trabajo futuro.

Limitaciones y restricciones

Limitaciones Técnicas del Prototipo

Entorno de Laboratorio.. El prototipo fue validado en un entorno controlado con:

- Volumen reducido de transacciones (50-100 multas de prueba)
- Red local de Hardhat para Ethereum (no blockchain pública real)
- Datos sintéticos que no reflejan la complejidad de datos reales

Una implementación en producción enfrentaría cargas de trabajo 1000x mayores (457,000 comparendos semestrales según datos de Bogotá), requiriendo optimizaciones adicionales de rendimiento y estrategias de sharding.

Escalabilidad de IPFS.. La estrategia de pinning implementada (mantener todas las evidencias en nodo local) no escala indefinidamente. Para producción se requiere:

- Política de retención temporal (ej. 5 años según normativa)
- Pinning distribuido entre múltiples nodos
- Estrategias de archivado para evidencias antiguas

Latencia de Sincronización.. La sincronización entre Hyperledger Fabric y Ethereum no es instantánea, creando ventanas de inconsistencia temporal (1-5 segundos). Aunque este tiempo es aceptable para el caso de uso (las multas no requieren consistencia en milisegundos), podría ser limitante para aplicaciones que requieran sincronización en tiempo real.

Limitaciones Metodológicas

Datos Sintéticos.. La imposibilidad de acceder a datos reales de multas (por restricciones de privacidad) limitó la validación del sistema. Los datos sintéticos generados no capturan:

- Variabilidad de formatos de placas (motocicletas, taxis, diplomáticos)
- Casos atípicos (placas extranjeras, vehículos especiales)
- Inconsistencias en datos del RUNT real

Integración Simulada con SIMIT/RUNT. La simulación de APIs gubernamentales limita la validación de:

- Latencias reales de servicios externos
- Manejo de fallas y timeouts
- Formatos exactos de respuestas

Para producción, se requiere piloto con integración real bajo convenio con las entidades.

Limitaciones de Seguridad

Ausencia de Auditoría Formal.. No se realizaron:

- Auditoría de seguridad formal del Smart Contract
- Pentesting de la API REST
- Análisis de vulnerabilidades en chaincode de Fabric

Para producción se recomienda contratar auditorías especializadas (ej. Trail of Bits, ConsenSys Diligence).

Gestión de Claves.. El prototipo utiliza claves privadas almacenadas en archivos .env, lo cual no es aceptable para producción. Se requiere:

- Hardware Security Modules (HSM) para claves críticas
- Rotación automática de credenciales
- Políticas de backup seguro

Lecciones aprendidas

Complejidad de Hyperledger Fabric

La implementación de Hyperledger Fabric presentó una curva de aprendizaje significativa. Aspectos que consumieron más tiempo:

- Configuración de la red con múltiples organizaciones
- Generación y gestión de certificados
- Debugging de políticas de endorsement
- Configuración de Private Data Collections

Lección: Para equipos sin experiencia previa en Fabric, se recomienda iniciar con redes de una sola organización y agregar complejidad incrementalmente.

Trade-off Descentralización vs Rendimiento

Ethereum ofrece mayor descentralización pero menor rendimiento (15-30 TPS) comparado con Hyperledger Fabric (>1000 TPS). Para el caso de uso de fotomultas:

- La mayoría de operaciones (90 %+) son internas y se benefician de Fabric
- Solo metadatos públicos van a Ethereum, reduciendo costos

Lección: La arquitectura híbrida permite optimizar el trade-off delegando cada tipo de operación a la blockchain más apropiada.

Importancia de UX en sistemas blockchain

La complejidad técnica de blockchain debe ser invisible para el usuario final. Aspectos críticos implementados:

- No requerir que el usuario tenga wallet de Ethereum
- Abstraer conceptos técnicos (gas, confirmaciones, hashes) detrás de interfaces amigables
- Proporcionar feedback claro durante operaciones asíncronas

Lección: El éxito de adopción depende más de UX que de la robustez técnica de la blockchain subyacente.

Diseño Modular para Evolución

La separación clara entre capas (HyperledgerService, EthereumService, SyncService) facilitó:

- Testing independiente de cada componente

- Actualización de dependencias sin afectar otros módulos
- Potencial migración a otras blockchains si fuese necesario

Lección: La modularidad es crítica en sistemas blockchain donde el ecosistema evoluciona rápidamente.

Despliegue en producción

Basado en las lecciones aprendidas, se identifican requisitos críticos para migrar a producción:

Infraestructura

- **Hyperledger Fabric:** Mínimo 3 organizaciones con 2 peers cada una, orderer con consenso Raft (3+ nodos)
- **Ethereum:** Migrar de testnet a mainnet o implementar Layer 2 (Polygon, Arbitrum) para reducir costos
- **IPFS:** Cluster de nodos con pinning distribuido y políticas de retención

Seguridad

- Auditoría formal de Smart Contracts y chaincode
- Implementación de HSM para claves privadas
- Pentesting de API REST
- Configuración de firewall y VPN para red Fabric

Operaciones

- Monitoreo 24/7 con alertas automáticas
- Plan de recuperación ante desastres
- SLA definidos para cada componente
- Procedimientos de actualización sin downtime

La arquitectura híbrida implementada demuestra su viabilidad técnica y proporciona una base sólida para evolucionar hacia un sistema de producción robusto.

Conclusiones y trabajo futuro

El presente trabajo logró diseñar y desarrollar exitosamente un prototipo basado en una metodología de software que garantiza el registro inmutable y la trazabilidad completa de estados en el proceso de fotocomparendos en Bogotá mediante tecnologías de redes distribuidas. La arquitectura híbrida *blockchain* que combina Hyperledger Fabric (privacidad) y Ethereum (transparencia pública) demuestra ser técnicamente viable para la gestión de fotocomparendos, abordando las limitaciones críticas identificadas en el sistema FÉNIX actual: modificación no autorizada de registros, falta de trazabilidad completa, vulnerabilidad de evidencia fotográfica y desconfianza ciudadana reflejada en una tasa de impugnación del 34.1 %.

Las pruebas realizadas confirman que el sistema cumple con los principios propuestos: el 100 % de los intentos de modificación fueron rechazados por los mecanismos de consenso, el sistema de direccionamiento por contenido (CID) de IPFS detectó automáticamente todas las alteraciones simuladas, y los tiempos de respuesta medidos (<3 segundos) validan que la arquitectura es viable para aplicaciones en tiempo real con 457,000 comparendos semestrales.

El backend implementado con interfaces REST estándar y el *frontend* desarrollado con React demuestran que las tecnologías *blockchain* pueden abstraerse detrás de APIs convencionales e integrarse con interfaces modernas sin comprometer la experiencia del usuario, facilitando la adopción por parte de instituciones gubernamentales.

Viabilidad de implementación

La evaluación integral del prototipo permite afirmar que la implementación de esta arquitectura en el contexto real de Bogotá es viable desde las perspectivas técnica, económica y operacional:

Viabilidad técnica. Los resultados de las pruebas de rendimiento (Tabla 19) demuestran que el prototipo cumple con los criterios técnicos establecidos en los objetivos específicos. El tiempo promedio de registro completo de un comparendo (incluyendo *upload* a IPFS, transacción en Hyperledger Fabric y publicación de metadatos en Ethereum) fue de 2.7 segundos, por debajo del umbral objetivo de $\leq$3 segundos. Extrapolando estos resultados al volumen operativo real de Bogotá (457,000 comparendos semestrales, equivalente a 3,000 diarios), la arquitectura puede procesar esta carga con los recursos de infraestructura estimados: 3-5 nodos Hyperledger Fabric, 1 nodo Ethereum (o conexión a red pública), y 2-3 nodos IPFS con replicación.

Las consideraciones de escalabilidad identificadas en la sección de Discusión (11.4.1) indican que, para garantizar el rendimiento en escenarios de pico (ej. campañas de fotodetección masiva), se requerirían optimizaciones como *sharding* de canales en Fabric por tipo de infracción y uso de soluciones *Layer 2* en

Ethereum para reducir latencia. No obstante, estas son mejoras incrementales sobre una arquitectura base ya validada.

Viabilidad económica. El análisis costo-beneficio presentado en la sección de Discusión (11.3.2) establece que la inversión inicial estimada para implementación en producción se justifica ampliamente frente a los beneficios cuantificables. Considerando que el presunto detrimento patrimonial identificado por la Contraloría de Bogotá en el sistema FÉNIX asciende a más de $8,000 millones de pesos, y que la carga operativa de procesar 155,854 PQRSD semestrales representa costos administrativos sustanciales, el retorno de inversión (ROI) proyectado se estima en 18-24 meses de operación.

Los costos operativos principales de la arquitectura *blockchain* incluyen: (1) infraestructura de servidores para nodos Fabric e IPFS (cloud o *on-premise*), (2) *gas fees* en Ethereum para transacciones públicas (mitigables mediante uso de redes *Layer 2* o Ethereum *testnets* subsidiadas), y (3) personal técnico especializado para mantenimiento. Estos costos son comparables o inferiores a los costos de auditoría, litigio y procesamiento de PQRSD del sistema actual, con el beneficio adicional de reducción proyectada del 30-50 % en PQRSD relacionadas con dudas sobre integridad de evidencia.

Viabilidad operacional. La integración del prototipo con sistemas existentes es factible mediante las interfaces REST documentadas con Swagger. El diseño modular permite una migración gradual desde el sistema FÉNIX actual: en una primera fase, ambos sistemas pueden operar en paralelo (*dual-write*), con el *blockchain* actuando como registro de auditoría inmutable mientras FÉNIX mantiene operaciones transaccionales. Una vez validada la estabilidad, el *blockchain* puede convertirse en el sistema de registro primario (*system of record*).

Los requisitos de capacitación para personal de la SDM son moderados. Los agentes de tránsito interactuarían con la misma interfaz web que actualmente usan, con cambios mínimos en flujo de trabajo. El personal técnico requeriría formación en Hyperledger Fabric y administración de nodos IPFS, pero la abstracción mediante API REST minimiza la complejidad para desarrolladores de aplicaciones frontales.

La ruta de migración propuesta incluye: (1) fase piloto con 5,000-10,000 comparendos reales, (2) evaluación de rendimiento bajo carga real y ajustes de optimización, (3) migración gradual por tipo de infracción (comenzando con fotomultas de cámaras salvavidas), y (4) integración con sistemas nacionales como SIMIT y RUNT para interoperabilidad completa.

Gobernanza de la red blockchain

Un aspecto fundamental para la implementación exitosa de esta arquitectura es el modelo de gobernanza de la red *blockchain*, que determina quién opera los nodos, cómo se toman decisiones sobre actualizaciones del sistema, y cómo se balancea la descentralización con la responsabilidad institucional.

Modelo de gobernanza propuesto para consorcio. Se propone un modelo de consorcio multi-stakeholder donde la red Hyperledger Fabric sea operada por las siguientes entidades:

- **Nodos Validadores:** Secretaría Distrital de Movilidad (SDM), Contraloría de Bogotá, y Policía Nacional - Dirección de Tránsito y Transporte. Estos tres nodos participan en el consenso PBFT (*Practical Byzantine Fault Tolerance*) para validar transacciones. La configuración requiere mayoría (2 de 3) para aprobar bloques, garantizando que ninguna entidad única pueda manipular registros unilateralmente.
- **Nodos Observadores:** Veeduría Distrital y organizaciones ciudadanas autorizadas (ej. organizaciones de derechos del consumidor). Estos nodos tienen acceso de solo lectura al *ledger*, permitiendo auditoría independiente sin participar en consenso.
- **Gestión de Identidades:** Fabric Certificate Authority (CA) administrada por la SDM, con políticas de multi-firma para cambios críticos (creación de nuevos usuarios administradores, revocación de certificados). Esto previene que un solo actor comprometa el sistema de identidades.
- **Actualización de *Chaincode*:** Cualquier modificación a los *smart contracts* requiere aprobación de 2 de 3 nodos validadores (SDM + Contraloría + Policía), siguiendo el proceso estándar de Hyperledger Fabric *Lifecycle*. Esto garantiza que cambios en lógica de negocio sean consensuados y auditados.

Aplicabilidad al contexto de fotocomparendos. Este modelo de gobernanza descentralizada aborda directamente el problema de confianza identificado en el sistema actual. Mientras el sistema FÉNIX centraliza el control en la SDM, generando la desconfianza reflejada en la tasa de impugnación del 34.1 %, el modelo de consorcio distribuye el poder de validación entre múltiples actores con intereses balanceados: la SDM (operación), la Contraloría (fiscalización), y la Policía (autoridad de tránsito).
El modelo *permissioned blockchain* (red permisionada) equilibra transparencia con privacidad de forma coherente con el marco legal colombiano:

- **Transparencia:** Los metadatos publicados en Ethereum permiten verificación ciudadana independiente, cumpliendo con la Ley 1712 de 2014 (Transparencia y Acceso a Información Pública).
- **Privacidad:** Los datos personales sensibles permanecen en Hyperledger Fabric con acceso controlado, cumpliendo con la Ley 1581 de 2012 (Protección de Datos Personales) y normativa GDPR aplicable.

La supervisión multi-entidad (*multi-stakeholder oversight*) proporciona *checks-and-balances* que previenen manipulación unilateral de registros, un problema recurrente en sistemas centralizados de gobierno según documentan casos internacionales revisados en el Estado del Arte (sección 4).

Consideraciones regulatorias y viabilidad. El modelo de gobernanza propuesto es coherente con el marco regulatorio e institucional colombiano:

- **Alineación legal:** Además de las Leyes 1712/2014 y 1581/2012 mencionadas, el modelo es compatible con el Código Nacional de Tránsito (Ley 769 de 2002 modificada por Ley 1843 de 2017) que establece las facultades sancionatorias de autoridades de tránsito y el debido proceso administrativo.
- **Marco institucional:** La propuesta se alinea con la Estrategia de Gobierno Digital de MinTIC y las Guías de Transformación Digital del Departamento Administrativo de la Función Pública (DAFP), que promueven la adopción de tecnologías emergentes con gobernanza responsable.
- **Precedentes internacionales:** El modelo de consorcio multi-entidad tiene precedentes exitosos en iniciativas como e-Estonia X-Road (infraestructura de datos gubernamentales con gobernanza distribuida entre múltiples agencias) y Dubai Land Department *blockchain* (registro de propiedades con participación de desarrolladores, bancos y gobierno).

La viabilidad institucional depende de la voluntad política de las entidades participantes y acuerdos inter-administrativos que formalicen roles y responsabilidades. La experiencia internacional sugiere que proyectos piloto exitosos son el catalizador más efectivo para obtener *buy-in* institucional.

Resolución de debilidades del sistema actual

El prototipo desarrollado aborda específicamente limitaciones técnicas identificadas en los informes de auditoría de la Contraloría de Bogotá sobre el sistema actual, así como desafíos técnicos generales en sistemas de registro administrativo documentados en el Estado del Arte (sección 4.4). La Tabla 27 presenta un mapeo entre estos desafíos y la solución arquitectónica implementada, con referencia a las evidencias empíricas obtenidas en las pruebas de resultados.

Tabla 27. *Mapeo de debilidades del sistema FÉNIX a soluciones del prototipo*

Debilidad FÉNIX (Estado del Arte)	**Solución Prototipo**	**Evidencia de Resultados**
Modificación no autorizada de registros	Inmutabilidad *blockchain* mediante consenso PBFT en Fabric y PoS en Ethereum	100 % rechazo de modificaciones (Tabla 18). Cualquier intento de alterar registros es bloqueado por mecanismos de consenso.
Falta de trazabilidad completa de transiciones de estado	Registro automático de todas las transiciones en *ledger* inmutable con *hash*, *timestamp* y firma digital	*Hash* criptográfico completo en cada estado (Tabla 23). Trazabilidad del 100 % de transiciones GENERADA → CERRADA.
Vulnerabilidad de evidencia fotográfica (manipulación, pérdida)	IPFS con *Content Identifier* (CID) para almacenamiento verificable e inmutable	100 % coincidencia de CID en verificaciones (Sección 10.2). Cualquier alteración de imagen genera CID diferente, detectable automáticamente.
Timestamps modificables que permiten disputas sobre horarios	*Timestamps blockchain* criptográficamente sellados, parte integral del *hash* del bloque	*Timestamps* inmutables validados en red distribuida. Alteración retroactiva requeriría reescribir cadena completa (computacionalmente prohibitivo).

Continúa en la siguiente página

Tabla 27. *(Continuación)*

Debilidad FÉNIX (Estado del Arte)	**Solución Prototipo**	**Evidencia de Resultados**
Alto índice de impugnaciones (34.1 %) por falta de confianza en integridad	Transparencia y verificabilidad: metadatos públicos en Ethereum + verificación de *hash* en IPFS público	Infraestructura técnica para verificación ciudadana autónoma implementada. Reducción esperada de impugnaciones requiere validación en piloto de campo (Trabajo Futuro).
Fraude documentado (Juzto.co, suplantación, comparendos fantasma)	Triple capa de seguridad: (1) *Blockchain* inmutable, (2) IPFS verificable, (3) *Hashing* criptográfico que vincula metadatos con evidencia	Integridad *end-to-end* garantizada. Fraude requeriría compromiso simultáneo de múltiples sistemas independientes con mecanismos de seguridad heterogéneos.

Nota. Elaboración propia basada en hallazgos de Contraloría de Bogotá (2024) y pruebas del prototipo.

Como se observa en la tabla, cada uno de los desafíos identificados tiene una solución técnica implementada en la arquitectura propuesta, validada mediante pruebas empíricas:

Inmutabilidad criptográfica vs. susceptibilidad de modificación. La auditoría de la Contraloría identificó inquietudes sobre la trazabilidad de cambios en registros. La inmutabilidad criptográfica de *blockchain* ofrece una solución: cualquier cambio intencional o accidental a un registro registrado sería matemáticamente detectable y trazable mediante el *hash* del bloque. Las pruebas de la Tabla 18 demuestran que en la arquitectura propuesta, el 100 % de intentos de modificación son rechazados por los mecanismos de consenso, garantizando que cualquier cambio

después del registro inicial sea auditable.

Trazabilidad completa vs. fragmentación potencial de auditoría. Un riesgo técnico general en sistemas basados en *logs* convencionales es la dependencia de mecanismos externos para garantizar la integridad del registro de auditoría. En contraste, el *blockchain ledger* implementado registra automáticamente cada transición de estado (GENERADA → NOTIFICADA → EN_APELACION → RESUELTA_APELACION → PAGADA/CANCELADA → CERRADA) con *hash* criptográfico, *timestamp* y firma digital del actor que ejecutó la transacción. Esto proporciona una cadena de custodia inherente al sistema. La Tabla 23 (sección 10.2) documenta que el prototipo cubre la totalidad de estados definidos.

Integridad verificable de evidencia fotográfica. Un desafío identificado en sistemas de fotocomparendos es la dificultad para terceros de verificar de manera independiente y determinista la autenticidad de evidencia fotográfica. El uso de IPFS con CID (*Content Identifier*) proporciona un mecanismo de verificación basado en contenido: cualquier alteración, incluso de un píxel, genera un CID diferente, permitiendo detectar cambios. Las pruebas de la sección 10.2 demuestran que en el prototipo, el 100 % de verificaciones de CID detectan correctamente cualquier alteración de contenido, validando que este mecanismo funciona como se espera.

***Timestamps* inmutables en registro distribuido.** Un riesgo técnico de los sistemas convencionales es que los *timestamps* son típicamente campos de base de datos editables. En una arquitectura *blockchain*, los *timestamps* son parte del *hash* del bloque, haciendo su alteración retroactiva computacionalmente prohibitiva sin invalidar toda la cadena criptográfica. Esto proporciona mayor garantía sobre horarios de eventos registrados, lo cual es relevante para disputas sobre infracciones y notificaciones.

Reducción de impugnaciones por transparencia. La tasa de impugnación del 34.1 % refleja desconfianza sistémica. Si bien no se realizaron pruebas de campo con ciudadanos reales (limitación del prototipo), la arquitectura proporciona la infraestructura técnica para verificación pública autónoma: cualquier ciudadano con el número de comparendo puede consultar metadatos en Ethereum y verificar el *hash* de evidencia en IPFS público. La hipótesis, respaldada por literatura de *e-government*, es que transparencia verificable reduce impugnaciones infundadas, aunque validar esto requiere un piloto controlado (propuesto en Trabajo Futuro).

Defensa multicapa contra manipulación de registros. Literatura sobre fraudes en sistemas de registro documenta que la centralización de controles y opacidad en auditoría son factores de riesgo. La arquitectura propuesta implementa múltiples capas independientes de verificabilidad: (1) *blockchain* con consenso distribuido para prevenir creación no autorizada de registros, (2) IPFS con contenido direccionable para detectar alteraciones de evidencia, y (3) *hashing* criptográfico que vincula metadatos con contenido. Esta redundancia técnica requeriría que un atacante comprometiera simultáneamente múltiples sistemas con mecanismos de seguridad diferentes, aumentando significativamente la dificultad de manipulación no detectada.

En síntesis, el prototipo demuestra viabilidad técnica y proporciona mecanismos arquitectónicos que aborden desafíos de integridad y auditabilidad identificados en la literatura sobre sistemas de registro administrativo y documentados en evaluaciones del sistema actual. La incorporación de garantías criptográficas en lugar de dependencia exclusiva de controles administrativos representa una mejora técnica con potencial de contribuir a la confianza en el sistema de fotocomparendos, aunque su validación completa requeriría un piloto con datos y usuarios reales.

Cierre de ciclo: Oráculos para notificación en la cadena de custodia

Si bien el prototipo desarrollado valida de manera rigurosa la inmutabilidad en los estados críticos del proceso (PENDING, PAID, APPEALED, RESOLVED_APPEAL, CANCELLED), la transición al entorno operativo exige abordar una brecha conceptual relevante: la certificación del estado NOTIFICADA, correspondiente a la notificación física y fehaciente al ciudadano.

En el marco arquitectónico propuesto, el estado NOTIFICADA requiere la intervención de un **oráculo**, entendido como un mecanismo técnico que permite incorporar evidencia verificable proveniente del mundo físico (por ejemplo, comprobantes de notificación postal, registros de mensajería SMS, o certificaciones de correo electrónico) en la infraestructura blockchain. Esta función excede el alcance experimental del prototipo, dado que implica:

- Integración con sistemas de notificación operados por entidades certificadas (empresas postales, operadores de telecomunicaciones).
- Implementación de mecanismos de verificación criptográfica para la validación de la entrega (por ejemplo, firmas digitales emitidas por el proveedor de notificación).

- Contratación e integración de servicios de oráculos especializados (tales como ChainLink Automation, Band Protocol) que operan en ambientes de producción.
- Aseguramiento de la validez jurídica de la cadena de custodia entre el evento físico y su registro en blockchain.

Para una implementación en ambiente productivo, se recomienda que la fase operativa contemple:

1. **Selección de oráculo:** Evaluar soluciones consolidadas en el mercado, como Chainlink Automation, que permiten la notarización de eventos externos con garantías criptográficas robustas.
2. **Integración con proveedores certificados:** Formalizar convenios con empresas de mensajería y correos que actúen como *attesters* de la notificación, mediante la emisión de comprobantes electrónicos firmados.
3. **Validación legal:** Garantizar que las pruebas criptográficas generadas por el oráculo sean reconocidas por las autoridades regulatorias competentes, tales como la Superintendencia Financiera y la Superintendencia de Industria y Comercio.
4. **Cierre de la cadena de custodia:** Con la certificación del estado NOTIFICADA mediante oráculo, y la gestión inmutable de los estados PENDING, PAID, APPEALED, RESOLVED_APPEAL y CANCELLED en blockchain, se logra una trazabilidad integral y auditable de todo el ciclo de vida del fotocomparendo (GENERADA → NOTIFICADA → [proceso] → PAGADA/CANCELADA → CERRADA).

En síntesis, el análisis realizado permite concluir que el prototipo cumple con el objetivo científico de demostrar la viabilidad técnica de blockchain para garantizar la inmutabilidad y trazabilidad en el proceso de fotocomparendos. La extensión hacia la certificación del estado NOTIFICADA mediante oráculos constituye un requisito ingenieril indispensable para la completitud operativa, sin que ello modifique las conclusiones técnicas ni la validez del enfoque propuesto.

Trabajo futuro

Para la evolución del proyecto se proponen las siguientes líneas de trabajo:

1. **Escalamiento a producción:** Escalar la red Fabric a múltiples organizaciones (SDM, Policía, Auditoría), implementar Private Data Collections, y desplegar nodos IPFS en infraestructura distribuida con políticas de replicación.

2. **Piloto controlado:** Realizar un piloto con la Secretaría Distrital de Movilidad utilizando datos reales (5,000-10,000 multas), integración con SIMIT/RUNT y evaluación de rendimiento bajo carga operativa.

3. **Funcionalidades avanzadas:** Implementar módulo de pagos integrado (PSE, billeteras digitales), sistema de apelaciones en línea, notificaciones automáticas y dashboard analítico para toma de decisiones.

4. **Replicabilidad:** Adaptar la arquitectura para otras ciudades colombianas mediante federación de redes Fabric, explorar soluciones Layer 2 para reducción de costos de gas, y proponer estandarización nacional de Smart Contracts.

Anexos

Anexo A: repositorios del proyecto

Enlaces a los Repositorios

El proyecto de fotomultas *blockchain* está distribuido en los siguientes repositorios de código fuente:

- **Frontend (*React + TypeScript*):** https://github.com/k-delta/fotomultas-front
- **Backend (*Smart Contracts + API*):** https://github.com/CristianGT089/backend-multas

Descripción técnica de cada repositorio

Repositorio Frontend: fotomultas-front. URL:

https://github.com/k-delta/fotomultas-front

Tecnologías principales:

- **Lenguaje:** *TypeScript* (99.2 %) con configuración *ESM*
- **Framework:** *React* 18+ con *Vite* como bundler
- **Estilos:** *Tailwind CSS* para diseño responsive
- **Estado:** *Zustand* para gestión de estado global
- **Testing:** *Jest* con *React Testing Library*
- **Licencia:** *MIT License*

Contenido:

- Interfaz de usuario para agentes de tránsito (registro de multas)
- Panel ciudadano para consulta y verificación de multas
- Dashboard administrativo con estadísticas y métricas
- Integración con *API REST* del backend
- Componentes reutilizables y diseño modular

Repositorio Backend: backend-multas. URL:

https://github.com/CristianGT089/backend-multas

Tecnologías principales:

- **Lenguaje:** *TypeScript* (93.1 %), *JavaScript* (4.1 %), *Solidity* (2.8 %)
- **Blockchain:** *Smart Contracts* en *Solidity* para *Ethereum*
- **Framework:** *Express.js* con *TypeScript* para *API REST*
- **Desarrollo:** *Hardhat* para compilación y despliegue de contratos
- **Testing:** *Vitest* para pruebas unitarias e integración
- **Almacenamiento:** Integración con *IPFS* para evidencias

Contenido:

- *Smart Contract* `FineManagement.sol` para gestión de multas
- *API REST* con endpoints para registro, consulta y actualización
- Servicios de integración con *IPFS* y blockchain
- Configuración de *Hardhat* para desarrollo local y testnet
- Scripts de despliegue y testing automatizado
- Documentación *Swagger* para la *API*

Instrucciones de acceso

Para acceder al código fuente del proyecto:

1. **Clonar repositorios:**

```
git clone https://github.com/CristianGT089/backend-multas
git clone https://github.com/k-delta/fotomultas-front
```

2. **Revisar documentación:** Cada repositorio incluye archivos README con instrucciones de instalación y configuración
3. **Explorar código:** El código está organizado en carpetas lógicas (src/, contracts/, test/, etc.)

Anexo B: manual de usuario

Manual para Agentes de Tránsito

1. Iniciar Sesión..

- Acceder a la URL del sistema
- Ingresar credenciales proporcionadas por el administrador
- Seleccionar rol .Agente de Tránsito"

2. Registrar una Multa..

- En el menú principal, seleccionar Registrar Multa"
- Completar el formulario con:
  - Número de placa del vehículo
  - Tipo de infracción (seleccionar de lista desplegable)
  - Ubicación (GPS automático o manual)
  - Costo de la multa (calculado automáticamente según tipo)
- Cargar evidencia fotográfica (máximo 5MB, formato JPG/PNG)
- Hacer clic en Registrar Multa"
- Esperar confirmación de blockchain (aprox. 2-5 segundos)
- Anotar el ID de multa generado para referencia

3. Actualizar Estado de multa..

- Buscar multa por ID o número de placa
- Seleccionar .Actualizar Estado"
- Elegir nuevo estado (Pagada, En Apelación, etc.)
- Ingresar razón del cambio
- Confirmar actualización

Manual para ciudadanos

1. Consultar multas..

- Acceder a la sección pública (sin autenticación requerida)
- Ingresar número de placa del vehículo
- Hacer clic en "Buscar"
- Revisar lista de multas asociadas

2. Verificar Integridad de evidencia..

- Seleccionar una multa de la lista
- Hacer clic en "Verificar Integridad"
- El sistema compara el hash de la evidencia en blockchain con el archivo en IPFS
- Se muestra resultado: .Evidencia Verificada.o .Evidencia Alterada"

3. Presentar apelación..

- Crear cuenta en el sistema (requiere verificación de identidad)
- Seleccionar multa a apelar
- Completar formulario de apelación con argumentos

- Cargar evidencias de respaldo (opcional)
- Enviar apelación
- Esperar notificación de resolución (máximo 30 días hábiles)

Anexo C: glosario de términos

La Tabla 28 presenta las definiciones de los principales términos técnicos utilizados en este documento.

Tabla 28. *Glosario de Términos Técnicos*

Término	**Definición**
ABI (*Application Binary Interface*)	Interfaz que define cómo llamar funciones de un *Smart Contract* desde aplicaciones externas. Contiene nombres de funciones, parámetros y tipos de retorno.
Blockchain	Tecnología de registro distribuido que almacena datos en bloques encadenados mediante *hashes* criptográficos, garantizando inmutabilidad.
CA (*Certificate Authority*)	Entidad que emite y gestiona certificados digitales en una red *Hyperledger Fabric*, controlando identidades y permisos.
Chaincode	*Smart Contract* en el contexto de *Hyperledger Fabric*, generalmente escrito en *Go*, que define la lógica de negocio.
CID (*Content Identifier*)	*Hash* único que identifica un archivo en *IPFS*. Se genera mediante criptografía del contenido del archivo.
Consenso	Mecanismo mediante el cual los nodos de una *blockchain* acuerdan la validez de las transacciones. Ejemplos: *PBFT*, *PoS*, *PoW*.

Continúa en la siguiente página

Tabla 28. *(Continuación)*

Término	**Definición**
DLT (*Distributed Ledger Technology*)	Tecnología de libro mayor distribuido que mantiene registros sincronizados entre múltiples nodos sin autoridad central.
Ethers.js	Biblioteca *JavaScript* para interactuar con la *blockchain* de *Ethereum*, permitiendo leer datos y enviar transacciones.
Gas	Unidad de medida del costo computacional en *Ethereum*. Cada operación consume *gas* que se paga en *Ether*.
Hardhat	Framework de desarrollo para *Ethereum* que facilita compilación, testing y despliegue de *Smart Contracts*.
Hash Criptográfico	Función matemática que convierte datos de cualquier tamaño en una cadena de longitud fija. Ejemplos: *SHA-256*, *Keccak-256*.
Hyperledger Fabric	Plataforma de *blockchain* permisionada empresarial, parte del proyecto *Hyperledger* de *Linux Foundation*.
Inmutabilidad	Propiedad de *blockchain* que garantiza que datos una vez escritos no pueden ser alterados sin dejar evidencia.
IPFS (*InterPlanetary File System*)	Sistema de archivos *peer-to-peer* distribuido que usa direccionamiento por contenido mediante *CIDs*.
Ledger	Libro mayor que registra todas las transacciones en una *blockchain*. Es distribuido y sincronizado entre nodos.
Nodo (*Node*)	Computadora que participa en una red *blockchain*, manteniendo una copia del *ledger* y validando transacciones.

Continúa en la siguiente página

Tabla 28. *(Continuación)*

Término	**Definición**
OpenZeppelin	Librería de *Smart Contracts* auditados y seguros para *Ethereum*, proporciona implementaciones estándar de tokens, control de acceso, etc.
Orderer	Nodo en *Hyperledger Fabric* que ordena transacciones y las agrupa en bloques para distribuir a los *peers*.
PBFT (*Practical Byzantine Fault Tolerance*)	Algoritmo de consenso tolerante a fallas bizantinas usado en *Hyperledger Fabric*, eficiente para redes permisionadas.
Peer	Nodo en *Hyperledger Fabric* que mantiene una copia del *ledger* y ejecuta *chaincode*.
Pinning	En *IPFS*, mantener un archivo almacenado permanentemente en un nodo para garantizar su disponibilidad.
PoS (*Proof of Stake*)	Mecanismo de consenso donde validadores son seleccionados según la cantidad de criptomoneda que poseen.
PoW (*Proof of Work*)	Mecanismo de consenso que requiere resolver acertijos criptográficos complejos para validar bloques.
Private Data Collections	Funcionalidad de *Hyperledger Fabric* para almacenar datos privados que solo ciertos nodos pueden acceder.
Smart Contract	Programa autoejecutante almacenado en *blockchain* que ejecuta lógica de negocio cuando se cumplen condiciones.
Solidity	Lenguaje de programación orientado a objetos para escribir *Smart Contracts* en *Ethereum*.

Continúa en la siguiente página

Tabla 28. *(Continuación)*

Término	**Definición**
Testnet	Red de prueba de *blockchain* que imita el funcionamiento de la red principal pero sin valor real. Ejemplo: *Sepolia*.
Transaction Hash	Identificador único de una transacción en *blockchain*, generado mediante *hash* criptográfico de su contenido.
TypeScript	*Superset* de *JavaScript* con tipado estático, usado para desarrollo backend del proyecto.
Wallet	Software que almacena claves privadas y permite firmar transacciones en *blockchain*.

Nota. Elaboración propia.

Referencias

Adel, K., Elhakeem, A., & Marzouk, M. (2023). Decentralized System for Construction Projects Data Management Using Blockchain and IPFS. *Journal of Civil Engineering and Management*, *29*(4), 342-359.

Anand, T., & Singh, V. (2024). *Traffic Violation Detection Using Blockchain* (Major Project Report, Bachelor of Technology in Computer Science & Engineering / Information Technology) [Submitted in partial fulfillment of the requirements for the Bachelor of Technology degree. Student IDs: Tejas Anand (201504), Vaishnavi Singh (201533).]. Jaypee University of Information Technology. Waknaghat, India.

Antonopoulos, A. M., & Harding, D. A. (2023). *Mastering Bitcoin: Programming the Open Blockchain*. O'Reilly Media.

Azaria, A., Ekblaw, A., Vieira, T., & Lippman, A. (2016). MedRec: Using Blockchain for Medical Data Access and Permission Management. *2016 2nd International Conference on Open and Big Data (OBD)*, 25-30. https://doi.org/10.1109/OBD.2016.11

Baird, L. (2016). The Swirlds Hashgraph Consensus Algorithm: Fair, Fast, Byzantine Fault Tolerance [Technical report on Hashgraph consensus mechanism achieving ABFT]. *Swirlds Tech Report SWIRLDS-TR-2016-01*. https://www.swirlds.com/downloads/SWIRLDS-TR-2016-01.pdf

Baird, L., Harmon, M., & Madsen, P. (2020). Hedera: A Public Hashgraph Network & Governing Council [Documento técnico oficial de Hedera Hashgraph]. https://hedera.com/hh_whitepaper_v2.1-20200815.pdf

Balcerzak, A. P., Nica, E., Rogalska, E., Poliak, M., Klieštik, T., & Sabie, O.-M. (2022). Blockchain technology and smart contracts in decentralized governance systems. *Administrative Sciences*, *12*(3), 96.

Benet, J. (2014). IPFS—Content Addressed, Versioned,P2P File System. https://doi.org/10.48550/arXiv.1407.3561

Buterin, V. (2014). A Next-Generation Smart Contract and Decentralized Application Platform. Ethereum White Paper.

Cachin, C. (2018). Architecture of the Hyperledger Blockchain Fabric [Accessed: 2025-05-07]. *arXiv preprint arXiv:1801.10228*. https://arxiv.org/abs/1801.10228

Chen, C.-L., Tu, C.-Y., Deng, Y.-Y., Huang, D.-C., Liu, L.-C., & Chen, H.-C. (2024). Blockchain-enabled transparent traffic enforcement for sustainable road safety in cities. *Sustainable Cities: Smart Technologies and Cities*, *6*, 1426036. https://doi.org/10.3389/frsc.2024.1426036

Choquevilca Quispe, V. A., & Morales Valencia, E. A. (2024). *Blockchain aplicado a la seguridad para la gestión de infracciones de tránsito en la Municipalidad Provincial del Cusco*. http://hdl.handle.net/20.500.12918/8657

Congreso de Colombia. (2017). Ley 1843 de 2017: Por medio de la cual se regula la instalación y puesta en marcha de sistemas automáticos, semiautomáticos y otros medios tecnológicos para la detección de infracciones y se dictan otras disposiciones [Diario Oficial No. 50.238]. http://www.suin-juriscol.gov.co/viewDocument.asp?id=30030416

Contraloría de Bogotá. (2024). Auditoría de Cumplimiento al Sistema FÉNIX - Secretaría Distrital de Movilidad [Hallazgo fiscal por presunto detrimento patrimonial superior a 8000 millones de pesos].

Contraloría General de la República de Colombia. (2023). Auditoría de Cumplimiento No. 90: Secretaría Distrital de Movilidad – SDM [Auditoría de cumplimiento a la Secretaría Distrital de Movilidad de Bogotá D.C.]. https://www.contraloria.gov.co/documents/auditorias/auditoria-cumplimiento-sdm-90-2023.pdf

Contraloría General de la República de Colombia. (2024). Informe de Auditoría 170100-0054-24: Auditoría de Cumplimiento a la Secretaría Distrital de Movilidad [Auditoría realizada a la Secretaría Distrital de Movilidad de Bogotá D.C.].

Coulouris, G., Dollimore, J., Kindberg, T., & Blair, G. (2011). *Distributed Systems: Concepts and Design* (5.ª ed.) [Fifth Edition]. Addison-Wesley.

Departamento Administrativo de la Función Pública (DAFP). (2021). Lineamientos y orientaciones para la disposición de consultas de acceso a información pública — Versión 1 [Consultado el 14 de octubre de 2025.]. https://www.funcionpublica.gov.co/documents/418548/34150781/Lineamientos%2By%2Borientaciones%2Bpara%2Bla%2Bdisposici%C3%B3n%2Bde%2B%2Bconsultas%2Bde%2Bacceso%2Ba%2Binformaci%C3%B3n%2Bp%C3%BAblica%2B-%2BVersi%C3%B3n%2B1%2B-%2BAgosto%2B2021.pdf/f6b57ff5-c175-43cc-1f81-bff5f801da61?t=1628645397432&version=1.2

Diffie, W., & Hellman, M. E. (2022). New Directions in Cryptography. En E. Name (Ed.), *Democratizing Cryptography: The Work of Whitfield Diffie and Martin Hellman* (pp. 365-390). Lugar u editorial.

Dutta, H., Nagesh, S., Talluri, J., & Bhaumik, P. (2023). A solution to blockchain smart contract based parametric transport and logistics insurance. *IEEE Transactions on Services Computing*, *16*(5), 3155-3167.

European Parliament and Council. (2014). Regulation (EU) No 910/2014 on Electronic Identification and Trust Services for Electronic Transactions (eIDAS) [Regulatory framework recognizing legal validity of blockchain-based records in EU]. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv:OJ.L_.2014.257.01.0073.01.ENG

Gálvez, J. F., Mejía, F., & Ruiz, O. (2018). Future trends in electricity markets: Blockchain as a new technology for energy data certification. *International Journal of Energy Economics and Policy*, *8*(3), 228-234.

Gamero Casado, E., & Fernández Ramos, S. (s.f.). *Procedimiento administrativo sancionador y motivación de las sanciones* [Preprint]. https://preprints.scielo.org/index.php/scielo/preprint/download/9212/17196/17799

Guardtime & e-Health Authority. (2016). *Implementation of medical blockchain technology in Estonia* (inf. téc.). Guardtime.

Haaramo, E. (2017). Sweden's land registry authority trials blockchain for property transactions. *Computer Weekly*.

JUIT Research Group. (2024). Traffic Violation Detection using Blockchain [Disponible en línea: http://www.ir.juit.ac.in:8080/jspui/bitstream/123456789/11502/1/Traffic%20Violation%20Detection%20using%20Blockchain.pdf].

Karlsson, K., Jiang, N., & Crane, L. (2019). *Permissioned Blockchains and Distributed Databases: A Performance Study* (Master's Thesis) [Performance comparison of Hyperledger Fabric and Apache Cassandra for different workloads]. KTH Royal Institute of Technology. https://www.diva-portal.org/smash/get/diva2:1258075/FULLTEXT01.pdf

Katz, J., & Lindell, Y. (2020). *Introduction to Modern Cryptography* (3.ª ed.). CRC Press.

King, S., & Nadal, S. (2012). Ppcoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake.

Li, W., He, M., & Haiquan, S. (2021). An overview of blockchain technology: applications, challenges and future trends. *2021 IEEE 11th International Conference on Electronics Information and Emergency Communication (ICEIEC) 2021 IEEE 11th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, 31-39.

Mani Joseph, P. e. a. (2023). Smart and Secure Blockchain Structure to Track Vehicle Record-keeping in the Sultanate of Oman. *International Journal on Recent and Innovation Trends in Computing and Communication. ...*

Maymounkov, P., & Mazieres, D. (2002). Kademlia: A peer-to-peer information system based on the xor metric. *International workshop on peer-to-peer systems*, 53-65.

Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of Applied Cryptography*. CRC Press. http://cacr.uwaterloo.ca/hac/

Meroni, G., Comuzzi, M., & Köpke, J. (2023). Editorial: Blockchain for Trusted Information Systems. *Frontiers in Blockchain*, *6*. https://doi.org/10.3389/fbloc.2023.1235704

Ministerio de Tecnologías de la Información y las Comunicaciones. (2020). Estrategia de Gobierno Digital [Accessed: 2025-01-11]. https://www.mintic.gov.co/portal/inicio/5755:Gobierno-Digital

Ministerio de Transporte de Colombia. (2020). Resolución 20203040011245 de 2020: Por la cual se establecen los criterios técnicos de seguridad vial para la instalación y operación de los sistemas automáticos, semiautomáticos y otros medios tecnológicos para la detección de presuntas infracciones al tránsito y se dictan otras disposiciones [Consultado en junio de 2025]. https://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=96272

Ministerio de Transporte de Colombia. (2023). ABC de las 'fotomultas'.

Mishra, R. K., Yadav, R. K., & Nath, P. (2024). Integration of Blockchain and IPFS: healthcare data management & sharing for IoT Environment. *Multimedia Tools and Applications*, 1-22. https://doi.org/10.1007/s11042-024-20092-3

Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System.

Narayanan, A., Bonneau, J., Felten, E., Miller, A., & Goldfeder, S. (2016). *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press. https://press.princeton.edu/books/hardcover/9780691171692/bitcoin-and-cryptocurrency-technologies

Omar, M. H., Taj-Eddin, I., Omar, N., & Ibrahim, H. (2024). SECURE ROAD TRAFFIC MANAGEMENT (SRTM) SYSTEM FOR TRAFFIC VIOLATION DETECTION AND RECORDING USING BLOCKCHAIN TECHNOLOGY. *Journal of Southwest Jiaotong University*, *59*(2). https://doi.org/10.35741/issn.0258-2724.59.2.1

Popov, S. (2018). The Tangle [Official IOTA technical whitepaper on DAG-based distributed ledger]. *IOTA Whitepaper*. https://assets.ctfassets.net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3.pdf

Reddit user. (2021). Estonia uses blockchain in day to day life via KSI by Guardtime.

Rezabala Loor, J. F., & Alci'var Cevallos, R. A. (2025). Blockchain aplicado a servicios gubernamentales: revisión sistemática de la literatura. *Revista Cienti'fica de Informática ENCRIPTAR*, *8*(15), 217-244. https://doi.org/10.56124/encriptar.v8i15.012

Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, *21*(2), 120-126. https://doi.org/10.1145/359340.359342

Ruan, P., Loghin, D., Ta, Q.-T., Zhang, M., Chen, G., & Ooi, B. C. (2021). Blockchains vs. Distributed Databases: Dichotomy and Fusion [Comprehensive taxonomy comparing blockchains and distributed databases across replication, concurrency, storage, and sharding]. *Proceedings of the 2021 International Conference on Management of Data (SIGMOD '21)*, 1504-1517. https://doi.org/10.1145/3448016.3452789

Schneier, B. (2007). *Applied Cryptography: Protocols, Algorithms, and Source Code in C* (20th Anniversary Edition). Wiley.

Secretaría Distrital de Movilidad. (2023a). ¿Cómo identificar si una cámara de detección electrónica se encuentra autorizada? [Consultado en junio de 2025]. https://www.movilidadbogota.gov.co/web/preguntas_frecuentes/como_identificar_si_una_camara_de_deteccion_electronica_se_encuentra_autorizada

Secretaría Distrital de Movilidad. (2023b). Secretaría de Movilidad aclara información sobre imposición de comparendos y procesos contravencionales y sancionatorios.

Secretaría Distrital de Movilidad. (2024). Estadísticas de Comparendos Bogotá 2024 [Accessed: 2025-01-11]. https://www.movilidadbogota.gov.co/web/observatorio

Semana. (2023, 15 de diciembre). *Miles de personas en Bogotá denuncian haber sido estafadas con falsas impugnaciones de fotomultas...* https://www.semana.com/politica/articulo/miles-de-personas-en-bogota-

denuncian-haber-sido-estafadas-con-falsas-impugnaciones-de-fotomultas-semana-revela-detalles-del-millonario-negocio-de-juztoco/202353/

Semana - Redacción Nación. (2023). Miles de personas en Bogotá denuncian haber sido estafadas con falsas impugnaciones de fotomultas: SEMANA revela detalles del millonario negocio de Juzto.co [Publicado 15 de diciembre de 2023. Consultado el 14 de octubre de 2025.]. Artículo web, SEMANA. https://www.semana.com/politica/articulo/miles-de-personas-en-bogota-denuncian-haber-sido-estafadas-con-falsas-impugnaciones-de-fotomultas-semana-revela-detalles-del-millonario-negocio-de-juztoco/202353/

Sentencia C-112 de 2018.

Sentencia No. 123 del 21 de junio de 2019 [Decisión sobre nulidad de prueba con violación del debido proceso]. (2019).

Suberg, W. (2017). Blockchain land registry trial in Sweden concludes second phase. *Cointelegraph*.

Superintendencia de Transporte. (2021). ABC para la gestión de procesos sancionatorios derivados de la detección de infracciones de tránsito mediante Sistemas automáticos.

Swan, M. (2015). *Blockchain: Blueprint for a new economy*. .º'Reilly Media, Inc."

Szabo, N. (1997). The Idea of Smart Contracts [Accessed: 2025-05-07].

Tan, E., Mahula, S., & Crompvoets, J. (2022). Blockchain governance in the public sector: A conceptual framework for public management. *Government Information Quarterly*, *39*(1), 101625.

Thanasas, G. L., Kampiotis, G., & Karkantzou, A. (2025). Enhancing Transparency and Efficiency in Auditing and Regulatory Compliance with Disruptive Technologies. *Theoretical Economics Letters*, *15*(1), 214-233.

van Steen, M., & Tanenbaum, A. S. (2017). *Sistemas Distribuidos* (3.ª ed.) [Traducción de Distributed Systems, 3rd edition]. distributed-systems.net. https://www.distributed-systems.net/index.php/books/ds3/

Vogels, W. (2008). Eventually Consistent: Building reliable distributed systems at a worldwide scale demands trade-offs? between consistency and availability. *Queue*, *6*(6), 14-19.

Voigt, P., & von dem Bussche, A. (2017). *The EU General Data Protection Regulation (GDPR): A Practical Guide* [Practical guide to GDPR compliance]. Springer. https://doi.org/10.1007/978-3-658-13399-0

Vukolic', M. (2015). The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication. *Open Problems in Network Security (iNetSec)*, *9591*, 112-125. https://doi.org/10.1007/978-3-319-39028-4_9

Wood, G. (2014). *Ethereum: A Secure Decentralised Generalised Transaction Ledger* (inf. téc. EIP-150 revision 2e819d) [Ethereum Yellow Paper]. Ethereum Project. https://ethereum.github.io/yellowpaper/paper.pdf

Xu, X., Weber, I., Staples, M., Zhu, L., Bosch, J., Bass, L., Pautasso, C., & Rimba, P. (2019). A Taxonomy of Blockchain-Based Systems for Architecture Design. *IEEE Transactions on Software Engineering*, *47*(1), 1-19. https://doi.org/10.1109/ICSA.2017.33

Young, J. (2017). Sweden officially started using Blockchain to register land and properties. *Cointelegraph*.

Yousfi, N., Graiet, M., Hamdi, H., & Abbassi, I. (2019). Blockchain Based Approach to Secure IoT Data. *International Journal of Computer Applications*, *181*(43), 1-6. https://doi.org/10.5120/ijca2019919305

Yousfi, N., Kmimech, M., Abbassi, I., Hamdi, H., & Graiet, M. (2022). ITS Traffic Violation Regulation Based on Blockchain Smart Contracts. *International Conference on Computational Collective Intelligence*, 459-471. https://doi.org/10.1007/978-3-031-16210-7_38

Zheng, Z., Xie, S., Dai, H.-N., Chen, X., & Wang, H. (2018). Blockchain challenges and opportunities: A survey. *International journal of web and grid services*, *14*(4), 352-375.