



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Sistema de transparencia basado en Blockchain

TESIS

Que para obtener el título de

Ingeniería en Computación

P R E S E N T A

Joya Venegas Jehosua Alan

DIRECTOR(A) DE TESIS

Dra. Rocío Alejandra Aldeco Pérez



Ciudad Universitaria, Cd. Mx., 2022

Agradecimientos

En primera instancia agradezco a mis docentes, ingenieras e ingenieros de gran sabiduría quienes se han esforzado por ayudarme a llegar al punto en el que me encuentro, reconociéndoles los grandes esfuerzos que han realizado aún con las adversidades que se generaron a partir de la pandemia, de manera especial a mi asesora la Dra. Rocío Aldeco quien me ha brindado su apoyo incondicional a lo largo de mis mejores años como universitario, así como en la generación del presente documento. Muchas gracias por su paciencia y compromiso.

Así mismo agradecer infinitamente a mis padres por haber forjado los cimientos de mi desarrollo, sé que el camino no ha sido nada fácil, pero siéntanse con la tranquilidad que han logrado formar dos profesionistas de bien. Todos y cada uno de ustedes (madre, padre, hermano, tía y MRA) han destinado tiempo para enseñarme nuevas cosas, brindándome aportaciones invaluableles que me servirán para toda la vida. Gracias a dios por permitirme compartir en vida este logro tan grande con ustedes, mis seres queridos.

Por último, agradecer al proyecto PAPIIT TA 101021 por el apoyo otorgado para la finalización de este trabajo y por haber impulsado mis deseos por iniciar mis estudios de posgrado.

Declaración de autenticidad

Yo, Jehosua Alan Joya Venegas egresado de la Facultad de Ingeniería, portador del número de cuenta 314221885 declaro que los resultados obtenidos en el presente trabajo son absolutamente originales y auténticos. En tal virtud que el contenido, las conclusiones, la redacción y los efectos académicos que se desprenden del trabajo propuesto de investigación y de este documento son y serán de mi sola y exclusiva responsabilidad legal y académica.

Jehosua Alan Joya Venegas

Ciudad Universitaria, 2022

Resumen

A lo largo de la historia, el ser humano se ha visto en la necesidad de designar autoridades competentes que se encarguen de la administración y planeación de los recursos naturales y económicos que posee una población, esto con el fin de ser destinados de forma eficiente para la búsqueda de un bien común.

Por esta razón es necesario generar sistemas que garanticen esta eficiencia de manera transparente para el usuario y confiables durante su proceso. En ese sentido, *blockchain* garantiza que la información almacenada y procesada se encuentre distribuida en varias computadoras a las que se le conoce como nodos, ejecutando un protocolo de consenso cada vez que se realice una operación de adición o modificación en la base de datos del sistema [1]. Una de las principales características que posee esta tecnología es el proceso de almacenamiento de bloques empleado para evitar la alteración o sabotaje de los datos ya registrados, ya que la validación por funciones hash y su sistema de verificación de transacciones, hacen de esta herramienta una de las más confiables y seguras.

Dada esta propuesta y sabiendo de los alcances que *blockchain* nos permite desarrollar, el usuario final podrá tener a su disposición y en todo momento la información necesaria acerca de las transacciones generadas en un sistema descentralizado orientado a solventar la corrupción en el sistema de verificación vehicular en el Estado de México, así como los detalles más relevantes que permitirán generar trazabilidad, confianza y sobre todo transparencia en un proceso que desafortunadamente ha sido invadido por la corrupción en los últimos años. Dicha propuesta ayudará a generar un impacto positivo en el manejo y administración de las finanzas de la SEDEMA (Secretaría del Medio Ambiente), así como la reducción de emisiones dañinas al medio ambiente.

Síntesis del trabajo

Los conceptos y terminología utilizados en esta sección, son explicados y profundizados a lo largo del escrito, se recomienda leer el capítulo 2 para su mejor entendimiento.

El presente trabajo surge de la necesidad de mejorar el sistema de verificación vehicular en el Estado de México ya que, a partir de la investigación realizada en los últimos meses, se ha detectado que existen áreas de oportunidad en los procesos que se realizan hoy en día en los más de 100 centros de verificación vehicular, entre los que destacan:

- Nulo seguimiento de los protocolos y asignaciones de hologramas a los vehículos.
- No se cuenta con registros históricos que permitan auditar a los verificentros.
- Falta de organización en el almacenamiento de las transacciones.
- El sistema en general funciona de forma centralizada. El personal de cada dependencia registra los valores obtenidos por el vehículo para posteriormente asignar un holograma con base en sus propios resultados. Sin la supervisión en tiempo real de alguna entidad reguladora.
- Si bien existe un rango de valores mínimos necesarios a cumplir para obtener un holograma, estos no se respetan debido al nulo seguimiento en los procesos que cada centro de verificación vehicular realiza. En otras palabras, cada unidad verificadora define sus propios criterios de aceptación.

Estas áreas de oportunidad dan pie a realizar actos fraudulentos o de corrupción en los procesos realizados en los centros de verificación vehicular. Por lo que se establece como objetivo el desarrollar un sistema robusto e incorruptible con ayuda de la tecnología *blockchain*, que permita descentralizar las tareas de validación y asignación de hologramas.

Así mismo el sistema colabora de forma indirecta en reducir los niveles de corrupción en México, también conocido como Índice de Percepción de la Corrupción (IPC). Sin mencionar que ayuda a alcanzar el objetivo propio del programa de verificación vehicular, que es reducir las partículas contaminantes que emiten los vehículos particulares.

El sistema final implementa un *blockchain* privado en el que se integran a los más de 100 centros de verificación vehicular (CVV), los cuales se comunican a través de un canal común, que permite la interacción entre las partes involucradas, esto quiere decir que una transacción es visible ante todas las entidades pertenecientes en la red. Para que una transacción forme parte del libro mayor (*ledger*), esta debe cumplir con los criterios establecidos previamente en el contrato inteligente. Es decir, la transacción realizada en el verificentro 'A', tiene que ser analizada y validada por un verificentro 'B' (diferente al

remitente). Sin embargo, ninguna de las partes involucradas, tiene conocimiento del remitente o destinatario a quien le está validado la transacción. Esto quiere decir que existe una capa de anonimato al momento de realiza la asignación del holograma.

Una vez que las transacciones son validadas de forma descentralizada, forman parte del *blockchain*, que a su vez se sincroniza con cada una de las bases de datos ubicadas en cada CVV. Cabe mencionar que la cadena de bloques, basa su funcionamiento en teoría de números y funciones hash, por lo que resulta imposible el poder manipular las transacciones previamente almacenadas en el *ledger*.

Los resultados obtenidos en el trabajo fueron satisfactorios al lograr comprender la lógica de negocio y las necesidades que el sistema de verificación actual necesitaba, para así desarrollar una herramienta segura, robusta y funcional que permite almacenar las transacciones de todos los centros de verificación vehicular, basando su funcionamiento en *blockchain*. De lo cual se puede afirmar que:

1. Todas las entidades pertenecientes a la red, se ven involucradas en la aceptación o rechazo de transacciones, así como en la asignación de hologramas.
2. Resulta imposible modificar o alterar los registros una vez que forman parte del *ledger*. Por lo que esta información puede ser utilizada para auditorías en algún trabajo futuro.
3. La capa de anonimato entre el receptor y el remitente, ayudan a evitar actos de colusión entre las entidades, ya que resulta más costoso el convencer económicamente a los más de 100 CVV para realizar actos de corrupción, que el realizar las transacciones correctamente.

Índice general

Resumen	4
Síntesis del trabajo	4
Índice general	7
Capítulo 1. Introducción	11
1.2 La corrupción en México	12
1.3 La problemática a resolver	14
1.3 Objetivos	15
1.3.1 Objetivos generales	15
1.3.2. Objetivos particulares	15
1.4 Justificación de la investigación	15
Capítulo 2. Marco Teórico	17
2.1 Introducción a <i>Blockchain</i>	17
2.1.1 Características	18
2.1.2 Beneficios	22
2.1.3 Funciones Hash y Firma digital	22
2.2 Tipos de Blockchain	24
2.2.1 Públicos	24

2.2.2 Privados	25
2.2.3 Federados (Consortium)	26
2.2.4 <i>Blockchain permissionless y permissioned</i>	27
2.3 Protocolos de consenso	28
2.3.1 Protocolos de consenso basados en prueba	29
2.3.2 Protocolos basados en voto	29
2.3.2.1 PBFT (<i>Practical Byzantine Fault Tolerance</i>)	30
2.3.2.2 RAFT	31
2.4 <i>Hyperledger Fabric</i>	34
2.4.1 Contrato inteligente	34
2.4.2 Nodos o <i>Peers</i>	35
2.4.3 Nodo de ordenación	35
2.4.4 Organizaciones	35
2.4.5 Autoridad certificadora	36
2.4.6 Canal	36
2.4.7 Transacción	36
2.4.8 Funcionamiento de <i>Hyperledger Fabric</i>	36
2.5 Sistema de verificación vehicular hoy	39
2.5.1 Sistema de verificación vehicular	40
2.5.2 Funcionamiento actual	41
2.5.2.1 Vehículos exentos	42

2.5.2.2 Holograma doble cero “00”	43
2.5.2.3 Holograma cero “0”	44
2.5.2.4 Holograma uno “1”	45
2.5.2.5 Holograma dos “2”	45
2.5.2.6 Técnica de verificación de no aprobación (Rechazo técnico)	46
2.5.2.7 Criterios de aprobación SDB	46
2.5.2.8 Organización y gestión interna en los CVV	47
2.5.3 Discusión sobre el procedimiento actual	50
Capítulo 3. Estado del Arte	53
3.1 Proyecto XCEED en plantas automotrices de Renault Europa	53
3.2 Garantías Bancarias con Lygon 1B	54
3.3 IBM <i>Food Trust</i>	55
Capítulo 4. Sistema de Verificación Vehicular basado en <i>Blockchain</i>	58
4.1 Arquitectura	59
4.2 Implementación	61
4.2.1 Configuración de Autoridades Certificadoras	62
4.2.2 Creación del Bloque Génesis	63
4.2.3 Creación de Contenedores para Base de Datos (CouchDB)	64
4.2.4 Creación del Canal de comunicación	65
4.2.5 Publicación del <i>Chain Code</i>	65

4.2.6 Publicación del servicio <i>Hyperledger Explorer</i>	68
4.2.7 Creación del servidor para la interfaz gráfica	68
Capítulo 5. Resultados	69
5.1 Análisis del funcionamiento	69
5.2 <i>Hyperledger Explorer</i>	82
Capítulo 6. Conclusiones y trabajo futuro	90
6.1 Trabajo Futuro	92
Bibliografía	94
Anexo A. Código	98
Código de contrato inteligente	98
Código fuente de la implementación	102

Capítulo 1. Introducción

La corrupción es un problema potencial en todo el mundo. En donde se toma ventaja de conflictos de intereses existentes, favoreciendo de manera no transparente, incluso ilegal, a algunas entidades durante la asignación y gasto de fondos públicos.

En el informe de la Encuesta de Calidad Regulatoria e Impacto Gubernamental en Empresas (ENCRIGE) realizado por el INEGI en 2020, se indica que las unidades económicas del país consideran que el porcentaje en los actos de corrupción entre los servidores públicos ha pasado de un 82.2% en 2016 a un 71.5% en 2020. De los cuales, el 72.6%, considera que los actos de corrupción se producen para agilizar trámites y el 37.9% señala que dichos actos se generan para evitar multas o sanciones. [1]

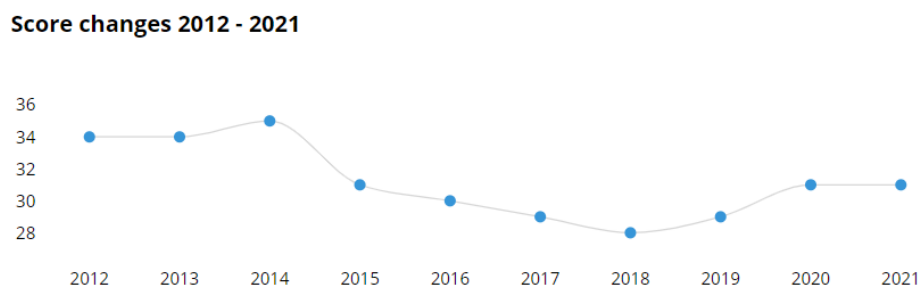
A nivel nacional, se estiman 204.3 mil unidades económicas han sido víctimas de actos de corrupción durante 2020, lo cual representa una tasa de 510 víctimas de corrupción por cada 10,000 unidades económicas, esto sin mencionar los actos de corrupción generados en personas físicas, los cuales se contabilizan en cientos de miles de casos anuales [1]. Estas cifras nos muestran que los gobiernos e instituciones son corruptibles en lo que se refiere al uso y administración de recursos públicos destinados a la mejora de los países, así como en los trámites necesarios para el cumplimiento de las responsabilidades que se tienen como ciudadano. Si bien cada una de las instituciones gubernamentales cuenta con un presupuesto calculado y ajustado a sus necesidades, así como protocolos de seguimiento a las peticiones de los ciudadanos, estos no siempre son empleados de la mejor manera.

Una posible solución adoptada por diversos gobiernos es la transparencia en la solicitud de trámites generales que todo ciudadano debe cumplir, para posteriormente realizar auditorías que puedan mostrar el cumplimiento de los protocolos establecidos por las dependencias gubernamentales. Esto se basa en el argumento de que dicha información es de carácter público, por lo que cualquier ciudadano (sea auditor o no), podrá tener acceso a los medios para consultar y constatar que los procedimientos fueron realizados y validados correctamente.

Sin embargo, los gobiernos a nivel mundial son mayormente centralizados. Esto quiere decir, que las decisiones son tomadas por los representantes de los gobiernos y a su vez, son ellos mismos quienes recaudan, administran, vigilan y castigan los recursos del país y las decisiones de uso de estos.

1.2 La corrupción en México

De acuerdo a la última publicación del Índice de Percepción de la Corrupción (IPC) a nivel mundial realizada en 2021, México obtuvo 31 de 88 puntos (siendo 1 el país con mayor índice que corrupción), ocupando la posición número 124 de 180 en el ranking de corrupción gubernamental, misma que es encabezada por países como Dinamarca, Finlandia y Nueva Zelanda, (todos con 88 puntos, lugares 180,179 y 178 respectivamente). Entre los países con peor posición en el ranking, se encuentran Siria, Sudán y Venezuela siendo los países con



más corrupción a nivel mundial. [35]

Fig. 1.1 IPC en México de 2012 a 2021. [35]

La figura 1.1 indica el IPC por año en México a partir del 2012. Resaltando que el año 2014, el índice de corrupción fue empeorando año tras año hasta llegar a su peor calificación en 2018.

De acuerdo a *Transparency International* (organización encargada de realizar el *ranking* a nivel mundial) el IPC es calculado a partir de trece estudios y evaluaciones procedentes de diversas instituciones de prestigio como el Banco Mundial y el Foro Económico Mundial, teniendo entre sus principales indicadores los siguientes puntos:

- Soborno
- Malversación de fondos públicos

- Funcionarios que utilizan su cargo público para obtener lucro personal sin afrontar las consecuencias
- Capacidad de los gobiernos para prevenir la corrupción en el sector público
- Excesiva burocracia en el sector público que puede incrementar las oportunidades de ejercer la corrupción
- Nepotismo en los nombramientos de funcionarios públicos
- Legislación que garantice la transparencia en las declaraciones de finanzas personales y posibles conflictos de interés en los cargos públicos
- Protección legal de denunciantes de casos de soborno y corrupción
- Captura del estado por intereses privados
- Acceso a la información sobre asuntos públicos de interés ciudadano y actividades de gobierno. [35]

Así mismo, el índice de Percepción de la Corrupción **no** cubre los siguientes aspectos:

- Fraude fiscal
- Movimientos ilícitos de fondos
- Facilitadores de la corrupción (abogados, contables, asesores financieros etc.)
- Blanqueo de fondos
- Corrupción en el sector privado
- Economías y mercados informales. [35]

Por lo que se deduce que el IPC en México es directamente afectado por actos de corrupción en el sector público. Según la ENCIG (Encuesta Nacional de Calidad e Impacto Gubernamental) los trámites con mayor índice de corrupción son:

- Permisos para construcción
- Licencia o permiso de suelo
- Inscripción en proceso de licitaciones
- Tramites vehiculares (Expedición de placas, licencias de conducir, verificación vehicular)
- Licencias o permisos para giros especiales
- Juicios laborales o mercantiles

- Permisos para importaciones y exportaciones [36]

1.3 La problemática a resolver

Uno de los trámites gubernamentales más afectados por la corrupción en México, es el de la verificación vehicular. Tan solo en 2015 la Secretaría del Medio Ambiente (SEDEMA), junto con el gobierno del Estado de México, lograron identificar a 28 verificentros que fueron suspendidos por irregularidades en sus procedimientos. El informe indica que fueron aplicados impuestos y sanciones que van desde 500 hasta 40 mil días de salario mínimo, a verificentros que operaban fuera de la normatividad ambiental, con irregularidades en documentación y en el proceso de mediciones de emisiones. [2]

De la misma manera, el gobierno del estado de México a través del programa “Mexiquense no te calles” habilitó una línea de comunicación directa anticorrupción que opera en los horarios de servicio de los centros de verificación. El objetivo es proporcionar transparencia a dichos procesos desde la recopilación de datos, toma de mediciones y resultados de las pruebas realizadas a los vehículos. Sin embargo, este programa no garantiza el cumplimiento de los protocolos establecidos para la verificación vehicular. Aún más, no hay forma de probar que los datos que aquí se presentan han sido verificados y provienen de una fuente confiable. Esto puede permitir que los encargados de reportar los resultados obtenidos en las pruebas de verificación, alteren la evidencia para lograr que los resultados se ajusten a niveles esperados de contaminación. Esto es, hay transparencia, pero no integridad ni trazabilidad.

Para ayudar a contrarrestar este fenómeno, proponemos hacer uso de la tecnología llamada “*blockchain*” o, en español, cadena de bloques. *Blockchain* es una base de datos distribuida gestionada por una red de pares que se adhiere colectivamente a algún protocolo de consenso. Esta tecnología es considerada una de las herramientas más robustas y seguras para la gestión y administración de información en ambientes donde la confianza no existe, teniendo diversas aplicaciones entre las que destacan contratos inteligentes, almacenamiento distribuido en la nube, identidades digitales, cadena de suministros, criptomonedas, registro y administración de datos entre cientos de aplicaciones más. Aunado a estas características, *blockchain* es bien conocido por su capacidad de operar de manera descentralizada, con el

apoyo de cada uno de los elementos integrantes del sistema para poder acreditar o rechazar transacciones que son de carácter público [3].

1.3 Objetivos

1.3.1 Objetivos generales

Desarrollar un sistema basado en *blockchain* que permita almacenar y validar las transacciones que se realizan a diario en los centros de verificación vehicular en el Estado de México.

1.3.2. Objetivos particulares

El sistema propuesto ayudará a generar transparencia en el proceso de verificación vehicular, debido a que las entidades involucradas validarán y asignarán hologramas a los vehículos de forma anónima y aleatoria, esto ayudará a generar confianza entre los ciudadanos ya que evitará la colusión y soborno de los participantes en el proceso.

El proyecto colaborará en la reducción del Índice de Percepción de la Corrupción en México, ya que, el trámite de verificación vehicular entra dentro de los indicadores que afecta directamente al índice antes mencionado.

Así mismo, el sistema, ayudará de forma indirecta a disminuir los niveles de contaminación en el Estado de México y área metropolitana. Esto debido a que se regulará de forma óptima el uso de vehículos que produzcan contaminantes dañinos al medio ambiente beneficiando al IMECA (Índice Metropolitano de la Calidad del Aire) [37].

1.4 Justificación de la investigación

El presente trabajo, nace de la necesidad de radicar la corrupción en México, de acuerdo con el Banco Mundial, el costo monetario de este fenómeno representa el 9% del Producto Interno Bruto (PIB) del país [38]. Así mismo, la necesidad de mejorar y automatizar el proceso de verificación vehicular, es una tarea que el gobierno del Estado de México ha dejado en el olvido.

Hoy en día, no basta con tener cámaras de video vigilancia en todos los centros de verificación, o realizar auditorías de forma parcial. Es necesario contar con soluciones tecnológicas que estén a la vanguardia para evitar colusiones entre empleados y ciudadanos. Es por eso que el presente trabajo se enfoca en el estudio del sistema de verificación vehicular hoy en día, para poder así mejorar los procesos internos, teniendo como resultado un sistema robusto e incorruptible que permita albergar la información de todas las entidades involucradas en el proceso, esto con ayuda de la tecnología *blockchain*.

Capítulo 2. Marco Teórico

Gracias a los avances de la tecnología es posible almacenar amplios volúmenes de información de manera electrónica, ya sea de forma estructurada en una base de datos, en algún sistema de almacenamiento en la nube o inclusive de forma local en dispositivos extraíbles. Esto ha revolucionado los procesos con los que a diario convivimos, por ejemplo, contratación de cuentas bancarias, telefónicas, declaración de impuestos, compra y venta de bienes y servicios. Sin embargo, estos tipos de almacenamientos se han visto limitados en cuanto a los beneficios extra que pudieran aportar al sistema completo, por lo que ha nacido la necesidad de encontrar nuevas y mejores técnicas para el almacenamiento de la información de forma rápida, segura y transparente.

En este capítulo se define el concepto de *blockchain* (cadena de bloques), resaltando los beneficios que aporta dicha tecnología a los sistemas actuales, así como los diferentes tipos de *blockchain* que actualmente se utilizan en la industria. Posteriormente se abordan algunos conceptos técnicos propios de esta tecnología, para finalmente presentar de forma detallada su funcionamiento.

2.1 Introducción a *Blockchain*

Blockchain es una base de datos distribuida que se encuentra en múltiples nodos de una red de manera síncrona, en donde cada transacción del sistema se le denomina “bloque”. Estos bloques contienen una marca de tiempo y un enlace con su bloque previo [4]. Los bloques de transacciones ya registradas en *blockchain* no pueden ser manipuladas ni eliminadas, por lo que esta característica asegura la inmutabilidad de los registros en el sistema. Así mismo todos los bloques contienen su digesto o *hash* único, lo que permite validar la integridad de los registros en todo momento.

En términos simples, *Blockchain* es un libro de registros distribuidos (o *ledger*) punto a punto en donde los registros o bloques, se encuentran enlazados y mediante un hash criptográfico garantizando la integridad e inmutabilidad de las transacciones [5] haciendo que los datos se encuentren disponibles y consistentes en todo momento, siempre y cuando hayan pasado por el proceso de verificación y formen parte del *ledger* (libro mayor).

En tal sentido, se asegura que la implementación de un sistema con *blockchain*, proporciona autonomía, sencillez y elementos de seguridad, lo que se verá reflejado directamente en la reducción de costos y riesgos para las organizaciones que decidan implementar esta tecnología en sus procesos cotidianos.

El concepto de *blockchain* como lo conocemos hoy en día ha ido evolucionando y ajustándose a las necesidades de cientos de aplicaciones con las que a diario interactuamos. Si bien se puede considerar a *Blockchain* como una tecnología “nueva”, la realidad es que la integración de esta herramienta enfocada a resolver un problema real, data del año 1991 en donde Stuart Haber y W. Scott Stornetta desarrollaron el primer trabajo de concatenación de cadenas de bloques protegidas criptográficamente en la que nadie podía manipular las marcas de tiempo de sus registros [6].

Sin embargo, fue hasta 2008 en donde la historia de *Blockchain* empezó a ganar relevancia gracias al trabajo de una persona o un grupo de personas que se denominaban Satoshi Nakamoto, el cual es conocido por ser el desarrollador principal del proyecto Bitcoin. A partir de este momento la palabra *blockchain* comenzaba a ganar relevancia entre los desarrolladores y las compañías enfocadas al ámbito de la seguridad de la información, pero no fue hasta 2009 cuando se lanzó el primer informe sobre los detalles de la tecnología, sus aplicaciones y las características que más adelante se abordarán [7].

A partir de que Nakamoto hace público el código base de este proyecto, llamado *bitcoin*, empieza una revolución intelectual en la que la industria adapta los rasgos de este sistema, para satisfacer las necesidades propias a las cuales no se había podido dar solución, o se tenían soluciones parciales. Aumentando los alcances que se tenían en un inicio con ayuda de aplicaciones de terceros, adaptación de código e inclusive el uso de hardware dedicado.

2.1.1 Características

Las tres características fundamentales que definen a *blockchain* o cadena de bloques son la descentralización, transparencia e inmutabilidad, las cuales profundizaremos a continuación.

- **Descentralización.** Ayuda a proporcionar confianza dentro de la red, ya que no requiere de la existencia de una “autoridad central” que administre el almacenamiento y movimiento de los datos contenidos en el sistema. Debido a que la red es una

implementación punto a punto (peer to peer), no existe necesidad de autenticar a las partes involucradas a través de un tercero, ya que cualquier usuario es capaz de ver las transacciones en la red. Sin embargo, el hecho de que la red se encuentre descentralizada, no se debe entender como falta de control sobre la misma, sino más bien se puede categorizar a la cadena de bloques como una aplicación capaz de establecer sus reglas de funcionamiento de una forma determinada e independiente a cualquier organismo.

- **Transparencia.** Asegura que todos los datos generados a lo largo del ciclo de vida de una aplicación sean registrados en la cadena de bloques de forma pública e inmutable. Todos pueden ver que acciones realizó una aplicación, pero nadie puede modificar esta evidencia.
- **Inmutabilidad.** Garantiza la integridad sobre los bloques de información añadidos, lo que impide la posibilidad de manipular o eliminar el contenido del mismo, esto con ayuda de marcas de tiempo y las funciones hash, que se abordarán más adelante.

Blockchain es capaz de almacenar una gran cantidad de datos de forma consecutiva, por lo que cuenta con un mecanismo que permite realizar consultas a la red de forma eficiente, es decir que se pueden hacer dichas consultas sin tener que descargar toda la información almacenada [8]. Para este propósito, la implementación de *blockchain*, utiliza una estructura llamada árbol de Merkle.

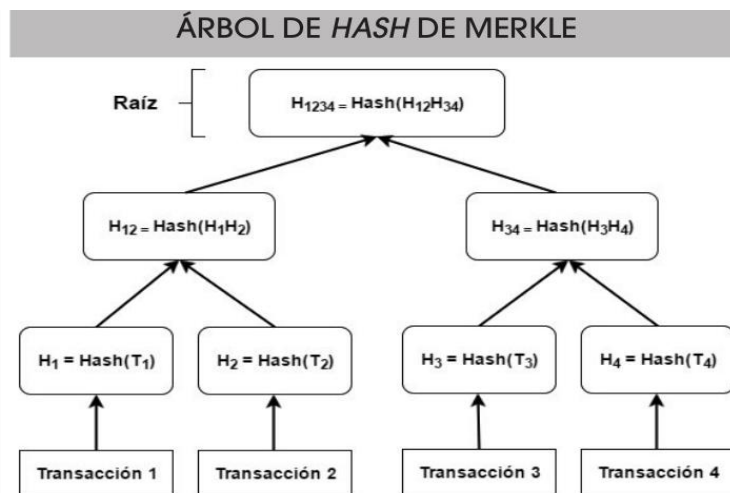


Fig. 2.1 Ejemplo de Árbol de Merkle [8].

Como se puede apreciar en la Figura 2.1, el árbol de Merkle basa su funcionamiento en una estructura de datos tipo árbol, en la cual se genera el hash de la información contenida en cada nodo hoja, posteriormente se concatenan los valores hash del nivel inferior y se le aplica la misma función hash a esta nueva cadena. Repitiendo este proceso se llega a un nivel donde hay sólo un nodo, denominado raíz. [8]. En específico en la figura 2.1 se implementa el uso de un árbol binario, lo que le provee al sistema la capacidad de realizar búsquedas dinámicas y eficientes con una complejidad $O(\log(n))$, donde n es el número total de nodos en la red. Sin embargo, hoy en día existen implementaciones *blockchain* que no necesariamente basan su funcionamiento en arboles binarios.

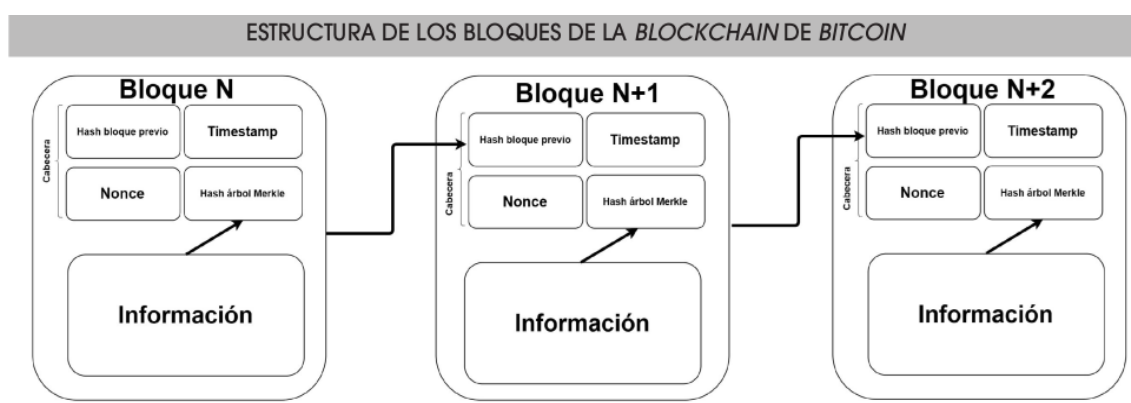


Fig 2.2. Estructura de los bloques de la *blockchain* [8].

En la Figura 2.2, se muestra la información que cada bloque almacenado en el libro mayor (también conocido como “*ledger*”) debe contener para la concatenación de información en forma de bloques de forma consecutiva. Estos elementos se describen a continuación.

- **Valor hash del bloque previo:** Este valor permite que los bloques se conecten de forma lineal con su antecesor y/o su predecesor, formando una cadena inmutable.
- **Marca de tiempo:** También conocido como *timestamp*, permite identificar el instante exacto en el que el bloque fue creado, dicha marca de tiempo es generada de forma automática.
- **Valor utilizado solo una vez ó *nonce* (*Number only used once*):** Es un valor único generado mediante fuerza bruta en implementaciones *blockchain*

públicas, las cuales son parte de la prueba de trabajo en algunas criptomonedas.

- **Valor de raíz del árbol Merkle** (*root hash*): Con ayuda de este valor es que se referencia toda la información del bloque y permite identificar de forma sencilla la ubicación de este, sin necesidad de recorrer todo el árbol de Merkle.
- **Información**: Contiene los datos relevantes que la lógica de negocio necesita para poder funcionar.

La seguridad es una característica sobresaliente en esta tecnología ya que sustenta su funcionamiento en la criptografía, la cual evita que ningún ser humano o sistema informático sea capaz de quebrantar o corromper el libro de registros. Para poder operar en la red de *blockchain* es necesario disponer de un conjunto de claves asimétricas válidas que le permitan al usuario adicionar bloques de información en el libro mayor. Sin embargo, cabe mencionar que no todos los *blockchain* utilizan el mismo tipo de claves asimétricas. En una cadena de bloques, todas las transacciones deben ser firmadas usando la llave privada del emisor, teniendo en cuenta que su llave pública formará parte de la transacción y podrá ser vista por los miembros que conforman a la red, con el objetivo de identificar al autor del bloque en cuestión.

Una de las particularidades a resaltar en el uso de *blockchain* es la implementación de los contratos inteligentes (*smart contract*) los cuales sirven para establecer acuerdos entre dos o más partes involucradas en un sistema, de lo que se puede hacer, cómo se puede hacer y qué pasa si algo no se hace. Sin embargo, a diferencia de los contratos convencionales escritos en documentos impresos, los contratos inteligentes son capaces de ejecutarse por sí mismos para garantizar el cumplimiento de las normas previamente establecidas en el. En otras palabras, un *smart contract* no es más que un *script*² escrito en cualquier lenguaje de programación, que es ejecutado de forma autónoma y automática sin intermediarios ni mediadores que regulen su comportamiento. Así mismo se caracteriza por ser público, ya que se puede acceder al código fuente desde cualquier entidad perteneciente a la red, en donde podemos encontrar la lógica de negocio para la aplicación particular en cuestión.

² Documento de texto en donde se colocan instrucciones u ordenes que serán ejecutadas por algún interprete o compilador

2.1.2 Beneficios

La trazabilidad es uno de los beneficios más relevantes un sistema basado en blockchain. En sistemas de producción esta propiedad se refiere a la capacidad de rastrear todos los procesos, desde la adquisición de materias primas hasta la producción, consumo y eliminación de este, en otras palabras, un sistema trazable tiene la capacidad de proporcionar a los consumidores cierta garantía acerca de la procedencia legítima de algún bien o servicio conforme a las normas de calidad establecidas [8]. En los sistemas de información, esta propiedad permite hacer transparente la forma en que cierta información fue obtenida, creada, modificada, procesada y almacenada. También la podemos encontrar como *electronic provenance* o linaje electrónico [9].

Según el sentido de flujo en la trazabilidad, podemos aprovechar esta característica para obtener la información hacia atrás, que se refiere a la recepción de productos de los proveedores o generadores de servicios. O la trazabilidad hacia adelante en la que los productos o servicios son entregados a una entidad y a partir de este punto los bienes quedan fuera del control de la empresa.

Visto desde la perspectiva de auditoría, esta propiedad nos permite analizar la forma en que esta información ha sido usada para después hacer responsables a las correspondientes entidades de cualquier uso incorrecto de dicha información [10]. Así, resulta de gran utilidad al permitir recorrer la cadena de bloques y trazar todas las operaciones que se han realizado sobre una determinada dirección para detectar posibles alteraciones o anomalías en los procesos propios de alguna compañía o dependencia. Aprovechando las características de inmutabilidad en los nodos, autenticación mediante llaves simétricas y la transparencia en la red, lo que le permite generar consultas parciales o globales al *ledger*.

2.1.3 Funciones Hash y Firma digital

Como mencionamos anteriormente, hay dos elementos criptográficos usados dentro de blockchain, las funciones *hash* y la firma digital.

Una **función hash** es una función que se utiliza para mapear información digital de cualquier longitud a datos digitales de tamaño fijo. Los valores devueltos por este tipo de función se denominan digesto, valores resumen, o simplemente valores hash [11]. Este valor resumen

tiene la característica de ser único para un valor de entrada dado, así cualquier cambio en los datos de entrada cambia de manera significativa el dato de salida [12].

Existe también la función *hash* criptográfica, la cual es un tipo de función que posee ciertas características matemáticas. Esta función es unidireccional ya que dados ciertos datos de entrada estos se relacionan a un digesto determinado, pero dado un digesto es sumamente difícil obtener el mensaje original de donde se obtuvo éste. Los datos de entrada a menudo se denominan mensaje, y el resultado se denomina resumen o digesto del mensaje. Si se desconocen los datos de entrada, es extremadamente difícil reconstruir esos datos conociendo sólo el digesto, haciendo de esta función una función de sólo ida. Las funciones *hash* son parte fundamental de la criptografía moderna. [11]

Una de las aplicaciones criptográficas más conocidas de los algoritmos de *hash* es la verificación de integridad de un mensaje. Aquí estas funciones se usan para detectar si un mensaje ha sido modificado de manera no autorizada. Su funcionamiento consiste en calcular el digesto del mensaje original para después usar este como prueba para una posible verificación de si el mensaje ha sido modificado. Como se puede apreciar en la figura 2.3, el mensaje original pasa por una función *hash* para obtener un valor digesto, el cual será compartido con un tercero para su validación.

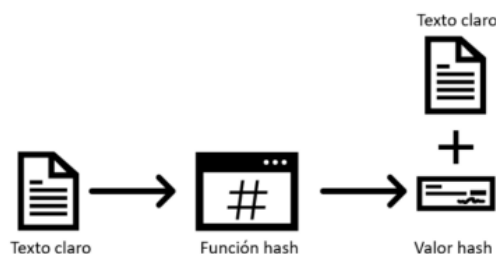


Figura 2.3. Esquema de generación de valor *hash* [11].

En el supuesto en el que el destinatario desea validar la autenticidad e integridad de un archivo que poseen ambas partes (receptor y remitente), por lo que en la figura 2.4 se puede apreciar el procedimiento a seguir una vez recibido el valor *hash* del texto claro original.

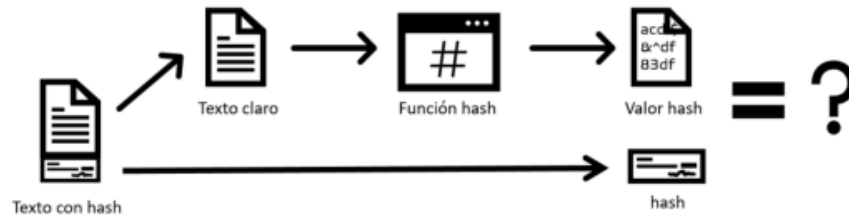


Figura 2.4. Esquema de verificación de integridad [11].

El receptor se encarga de aplicar la misma función *hash* al archivo por validar, obteniendo así un nuevo valor digesto, el cual deberá ser comparado con el valor *hash* que fue enviado por el remitente. De ser así, se afirma que los archivos en cuestión son iguales.

El concepto de firma digital de un documento no es más que el resultado de aplicar cierto algoritmo matemático (función *hash*) a su contenido y, seguidamente, cifrar el resultado con ayuda de la llave privada del remitente, generando así una firma electrónica o digital la cual ayudará a validar la autenticidad e integridad de algún archivo.

En el caso de *blockchain*, las funciones *hash* son usadas para garantizar la integridad de los datos creados por los participantes. Apoyado de la estructura del árbol de Merkle, se garantiza la inmutabilidad de estos datos. Así para cualquiera que quiera modificar el contenido o el orden de los bloques de manera maliciosa, será muy difícil computacionalmente dada la naturaleza sólo de ida de las funciones hash criptográficas.

2.2 Tipos de Blockchain

En la literatura encontramos principalmente dos tipos de cadena de bloques que se diferencian en la forma en que los usuarios tienen acceso a la red y a la información contenida en el *blockchain*. A continuación, se describen estos tipos a detalle.

2.2.1 Públicos

Las *blockchain* públicas, son aquellas a las que cualquiera puede ser parte de la red y los bloques generados pueden ser vistos por cualquiera. En general estas implementaciones son transparentes y los usuarios anónimos, así mismo ningún participante de la red cuenta con más privilegios sobre los demás, por lo que no existe un administrador en la red. Algunas de las redes públicas más conocidas son Ethereum, Bitcoin y Litecoin.

Para poder formar parte de la red pública, es necesario descargar la aplicación correspondiente con los datos más actualizados al momento de querer unirse a la red, lo cual puede llegar a tardar minutos u horas. Una vez que se tiene una copia idéntica del ledger, el usuario tiene la capacidad de proponer y validar transacciones.

La forma en que se validan las transacciones es muy particular de cada implementación, sin embargo, la técnica más utilizada es el realizar una prueba de trabajo (también conocida como “*proof of work*”) resolviendo problemas matemáticos, probando la capacidad de almacenamiento, entre otros, para proponer la adición de un nuevo bloque. Generalmente este tipo de implementaciones, se basan en incentivar a los nodos a proponer bloques de información válidos. La mayoría de las redes públicas dan recompensa en forma de criptomoneda al nodo que proporcione un bloque cuando éste es aceptado, lo cual genera poca eficiencia en la aprobación de transacciones. Este es el proceso conocido como “minado” en Bitcoin, aunque existen otros procesos dependiendo de la técnica usada.

2.2.2 Privados

Las *blockchain* privadas tienen un control del número de nodos involucrados en el sistema, así como la información detallada de cada uno de ellos, la cual es conocida por el resto de los nodos de la red. Esto significa que cada nodo es autenticado por alguna autoridad permitiéndole acceso a los recursos de la red.

A diferencia de implementaciones públicas de *blockchain*, en donde se requiere de pruebas de trabajo o minado para poder agregar nuevos bloques de datos, en un sistema privado se realiza la toma de decisión mediante un sistema de consenso basado en voto, en donde los nodos votan para acreditar o rechazar una transacción. Esto hace que el tiempo de aceptación de una transacción disminuya de forma considerable.

Si bien algunos autores demeritan este tipo de *blockchain* por carecer de anonimato en transacciones y eliminar las recompensas que los usuarios obtienen al agregar un bloque al *ledger*, estas surgen como una opción a empresas privadas que buscan hacer uso de los beneficios de la tecnología *blockchain*, pero manteniendo el control sobre la red, mediante un conjunto de usuarios privilegiados que pueden emitir o denegar permisos. [13]

2.2.3 Federados (Consortium)

Este tipo de *blockchain* son aplicaciones no abiertas a la participación del público en general ya que son controladas por un determinado número de organizaciones o entidades que se encargan de administrar la red en conjunto, así como mantener copias sincronizadas de los registros en cada uno de los nodos. El acceso a los datos es de forma parcial, mediante una interfaz web en la que los administradores proporcionan los datos relevantes a tratar.

Es importante señalar que al ser su acceso vía web y no como “nodos” de la red, es decir, no tienen una copia de la cadena, los usuarios comunes tendrán acceso a tanta información como los administradores decidan mostrarles a través de la misma. Se tendrán entonces opciones que varíen desde un gran nivel de transparencia hasta una transparencia nula [14].

En la Figura 2.5, se puede apreciar un cuadro comparativo de las principales características de los tres tipos de *blockchain*.



	Públicos Bitcoin, Ethereum, Litecoin	Privados Hyperledger, Corda, Quorum	Federados Hyperledger, Corda, Quorum
Cualquiera puede participar	✓	✗	✗
Los participantes actúan, en general, como nodos	✓	✗	✗
Transparencia	✓	≈	≈
Hay un único administrador	✗	✓	✗
Hay más de un administrador	✗	✗	✓
No hay administradores	✓	✗	✗
Ningún participante tiene más derechos que los demás	✓	✗	✗
Se pueden implementar Smart Contracts	✓	✓	✓
Existe recompensa por minado de bloques	≈	✗	✗
Soluciona problema de falta de confianza	✓	✗	≈
Seguridad basada en protocolos de consenso	✓	✗	≈
Seguridad basada en funciones hash	✓	≈	≈
Provee servicios en la nube	NA	NA	NA

✓ Sí ✗ No ≈ A veces NA No Aplica

Figura 2.5. Cuadro comparativo de tipos de *blockchain* [14].

2.2.4 Blockchain *permissionless* y *permissioned*

Existe una subdivisión entre los tipos de *blockchain* que pueden ser implementados en una red pública, privada o federada, estos son: permissionado (*permissionless*) y no permissionado (*permissioned*). El primero de ellos permite agregar nodos a la red sin necesidad de contar

con privilegios adicionales, es decir, cualquier nodo tiene la capacidad de unirse a la red sin necesidad de pasar por un proceso de verificación, a diferencia de las implementaciones *permissioned* en donde el administrador o los administradores del *blockchain* tienen la obligación de autenticar los nodos que conforman la red, así mismo, cada uno de estos nodos cuentan con una serie de privilegios que les permite acceder a cierta información dentro del *ledger*. A continuación, se muestran las combinaciones existentes y algunas de sus aplicaciones.

- **Permisiónadas Públicas:** En este *blockchain*, las personas pueden unirse y/o abandonar en cualquier momento, así mismo, son capaces de leer y escribir transacciones en el *ledger*. La confianza entre participantes es garantizada por el protocolo de consenso usado y total transparencia de las transacciones [15]. Las aplicaciones más conocidas con esta característica, radica en el ámbito de las criptomonedas como *Bitcoin*, *Zerocash* o *Ethereum*.
- **No Permisiónadas Públicas:** En este tipo de *blockchain*, se permite que cualquier nodo pueda leer los datos dentro del *ledger*, pero se requieren de permisos para poder formar parte de los consensos y posteriormente escribir transacciones en el libro mayor [15]. Algunos ejemplos de aplicaciones son *Ripple*, *EOS* y *Libra*.
- **Permisiónadas Privadas:** Este tipo de aplicaciones permiten a las organizaciones el poder colaborar en la red *blockchain* sin la necesidad de compartir su información de forma pública. Es decir, las personas pueden unirse o abandonar en cualquier momento la red, sin embargo, el contrato inteligente involucrado define qué personas tienen los privilegios necesarios para leer o escribir dentro del *ledger*.
- **No Permisiónadas Privadas:** Este tipo de aplicaciones son utilizadas en organizaciones donde los datos recabados son almacenados en el *ledger* por usuarios con privilegios de lectura y escritura. Los nodos de la red son controlados por un administrador, el cual se encarga de otorgar permisos para unirse, abandonar, leer o escribir. Algunas de las aplicaciones más conocidas son: *Hyperledger fabric* [16], *Monax* y *Multichain*.

2.3 Protocolos de consenso

Los protocolos de consenso nacen de la necesidad de coordinar los procesos dentro de un computador o un sistema distribuido de computadores, con el fin de llegar a un acuerdo en

común. En otras palabras, es un tipo de decisión en donde los nodos logran llegar a un acuerdo sobre las transacciones que serán aceptadas o rechazadas para formar parte del libro mayor [17].

Los dos principales protocolos de consenso son “*proof-based*” (basados en pruebas) y “*voting-based*” (basados en votaciones) los cuales serán explicados a continuación.

2.3.1 Protocolos de consenso basados en prueba

Los protocolos de consenso basados en pruebas se caracterizan por elegir al nodo que verificará una transacción solicitándole una prueba. Esta prueba puede ser la solución de una función computacionalmente difícil. Una vez que un nodo muestra dicha prueba, verifica una transacción y con ayuda del resto de los nodos de la red se decide si este bloque se agregará al libro mayor. Usualmente son utilizados en implementaciones públicas de *blockchain* en donde los usuarios obtienen una recompensa por añadir nuevos bloques a la red mediante el uso de su propio *hardware*. Algunos ejemplos de su uso son las criptomonedas como *bitcoin* o *ethereum*, en donde los usuarios ponen a disposición su poder computacional para “minar” los bloques que formarán parte de la red usando el protocolo llamado “Prueba de Trabajo”.

Actualmente existen varios protocolos basados en pruebas, los cuales han sido perfeccionados y modificados para sus aplicaciones. Entre ellos tenemos los siguientes:

1. *Proof of Work* (Prueba de trabajo): Resolviendo un acertijo matemático computacionalmente complicado.
2. *Proof of Stake* (Prueba de participación): Invirtiendo el dinero que se tiene.
3. *Proof of Elapsed Time* (Prueba de tiempo transcurrido): Se establecen lapsos de tiempo planificados
4. *Proof of Luck* (Prueba de suerte): Selecciona de forma aleatoria al nodo.
5. *Proof of Space* (Prueba de espacio): Selecciona dependiendo el espacio de disco disponible. [7]

2.3.2 Protocolos basados en voto

Los protocolos de consenso basados en voto nacen de la necesidad de reducir el tiempo de verificación de una transacción. Así mismo ayudan a solventar la sobrecarga de procesamiento requerido en la red para poder agregar bloques de datos al libro mayor. Su

funcionamiento esta basado en la idea de elegir por medio de un proceso de votación al nodo que podrá generar y agregar bloques al libro mayor sin necesidad de mostrar alguna prueba en particular como lo hacen los protocolos basados en prueba.

Este tipo de protocolos son usualmente usados en redes *blockchain* de tipo privado, en las cuales todos los nodos de la red son capaces de conocer la identidad de los demás participantes del sistema, lo que permite que se genere un consenso grupal o parcial (dependiendo las reglas de negocio) sobre las transacciones que serán añadidas en el *ledger*. Existen diversos protocolos de este tipo, aunque hay dos que son los más usados. Estos los explicaremos a continuación.

2.3.2.1 PBFT (*Practical Byzantine Fault Tolerance*)

Practical Byzantine Fault Tolerance es un protocolo que selecciona un grupo de nodos de la red mediante una autoridad central, de los cuales se denomina a un nodo como líder, dejando a lo demás como nodos de respaldo. Es de resaltar que todos los nodos seleccionados, se encuentran en constante comunicación con el objetivo de llegar a acuerdos internos dentro de la propia red, asumiendo que cada uno de los nodos es una entidad honesta que cuenta con una copia del *ledger*. [18]

El proceso de agregar un nuevo bloque a la red mediante el protocolo PBFT es mostrado en la Fig. 2.6 y funciona de la siguiente manera:

1. El cliente envía una petición al nodo líder que se estableció con autoridad, indicando que se desea insertar información a un bloque.
2. El nodo líder se encarga de recolectar los datos de la solicitud y agrupar la información en bloques.
3. Los datos son computados mediante una función *hash* para posteriormente ser compartido con el resto de los nodos de respaldo mediante un mensaje tipo *broadcast*⁴.
4. Se espera la aprobación de por lo menos dos terceras partes del número total de nodos en la red. El proceso de validación se realiza comparando el *hash* obtenido por cada nodo de respaldo y el *hash* obtenido por el nodo líder.

⁴ En una red de comunicaciones, el mensaje tipo *broadcast* se transmite a todos los miembros de una red para lograr una comunicación masiva de paquetes o datos.

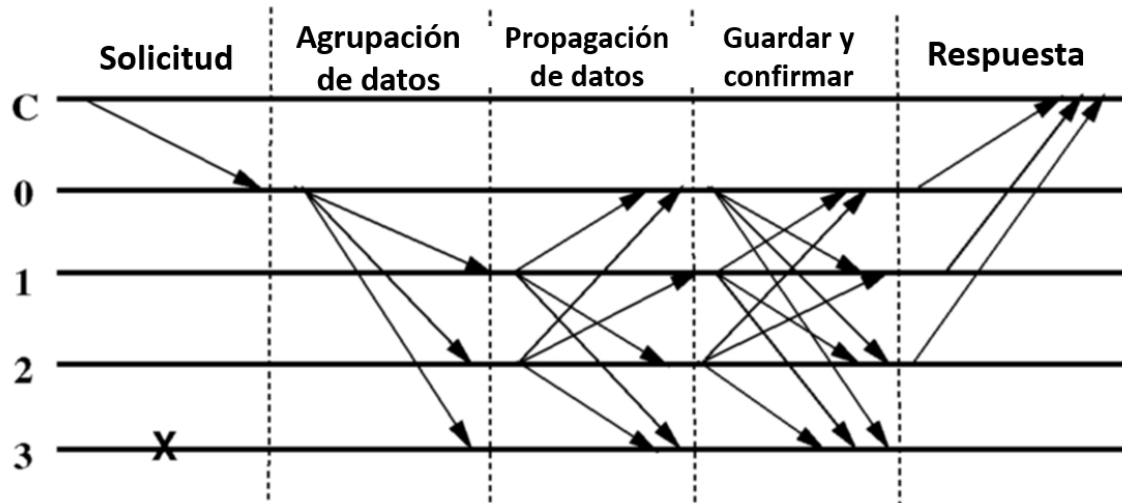


Figura 2.6 Proceso para agregar bloque en protocolo PBFT. [18]

Esto requiere designar al nodo líder y a los nodos de respaldo mediante un sistema de votación en la que cualquier entidad tiene la capacidad de postularse en un sistema democrático. Así mismo en una red privada en la que se han establecido con anterioridad los roles que cada uno de los nodos debe tomar, PBFT disminuye por completo la carga de trabajo que se debe realizar en cada nodo para su autenticación. A diferencia de una red pública, en donde la escalabilidad de la red resulta afectada debido a un sobre flujo en los canales de comunicación entre los nodos de respaldo ya que, al aumentar dichos nodos, por consecuencia la carga de información enviada y recibida por cada uno de ellos aumenta de manera considerable, lo que también representa un aumento en el consumo de energía en toda la infraestructura. Adicional a esto, PBFT es propenso a recibir ataques “Sybil”, ya que una entidad dentro de la red es capaz de crear múltiples identidades falsas, con lo cual se pudiera tener el control de forma parcial sobre la toma de decisiones en la red pública [18].

2.3.2.2 RAFT

El algoritmo de consenso RAFT fue creado en 2014 para ayudar a reducir el sobre flujo de mensajes entre nodos que nos proporciona la implementación PBFT. En este protocolo la comunicación se genera únicamente entre el nodo de respaldo y el nodo líder, eliminando mensajes entre nodos respaldo lo cual disminuye considerablemente la sobrecarga de información que viaja en los canales de la red. Así mismo, el uso de máquinas de estado en cada nodo es una de las principales características de este protocolo.

Cada uno de los nodos de la red puede tomar el estatus de líder, seguidor o candidato. El primero de ellos es el responsable de generar y actualizar la bitácora de transacciones en la red, debido a las peticiones del cliente, cabe mencionar que solamente puede existir un nodo líder en la red.

Los nodos seguidores (*followers*) se encargan de responder las peticiones provenientes del nodo líder y los nodos candidatos, mediante peticiones remotas (*Remote Procedure Calls*) también conocidas como RPC, mientras que los nodos candidatos, se encuentran a la espera de ser promovidos como los nuevos nodos líderes. El algoritmo de consenso RAFT se divide en las siguientes fases:

1. **Elección de nodo líder:** RAFT se encarga de dividir el tiempo total de ejecución en secciones previamente indexadas, en las cuales los nodos de respaldo se encuentran en constante comunicación con el nodo líder mediante RPC. En algún momento se dejarán de recibir peticiones del nodo líder, lo cual indica que dicho nodo ha dejado su rol actual para convertirse en un nodo de respaldo más, dando pie a la elección de un nuevo líder.

El primer paso en el proceso de elección consiste en actualizar el índice global de las secciones de tiempo en cada uno de los nodos, posteriormente se realiza un cambio de estado de “seguidor” a “candidato” en los nodos involucrados para finalmente participar en el proceso de elección, el cual llega a su fin cuando se cumple con alguno de los siguientes supuestos:

- a. Un nodo candidato gana la elección cuando recibe la mayoría de los votos en la red, lo cual permite que dicho nodo cambie su estado a “líder”, enviando un mensaje tipo *broadcast* hacia los demás nodos haciéndoles saber su nuevo estatus.
- b. Un nodo candidato se auto proclama ganador antes de que la votación termine, sin embargo, los nodos de respaldo se encargan de verificar el índice en la marca de tiempo del mensaje y compararlo con el índice actual por lo que, si el índice de dicho mensaje es menor al índice actual, se ignora la proclamación. No obstante, si el índice es superior al actual, entonces se reconoce a este nuevo nodo como el líder de la red, y se actualiza el estatus de todos los nodos de “candidato” a “seguidor”.

- c. El proceso de elección puede llegar a tener un empate en el número de votos para cada nodo, por lo que el proceso se reiniciará, aumentando el índice en las marcas de tiempo y asignando de forma aleatoria el orden en el que los nodos son representados, para evitar lidiar nuevamente con un empate.
2. **Replicación de registros:** El líder electo procesa las transacciones enviadas por los clientes para posteriormente asignarles un índice, logrando así mantener un orden en las peticiones. El líder crea un bloque de estas transacciones y las emite en paralelo a todos los seguidores, que a su vez replican esta transacción y envían un acuse de recibo al líder. [18]
 3. **Ejecución de transacciones:** Una vez que el nodo líder recibe la mayoría de las confirmaciones por parte de los nodos seguidores, la transacción es ejecutada y el cliente es notificado sobre el estatus de su petición.

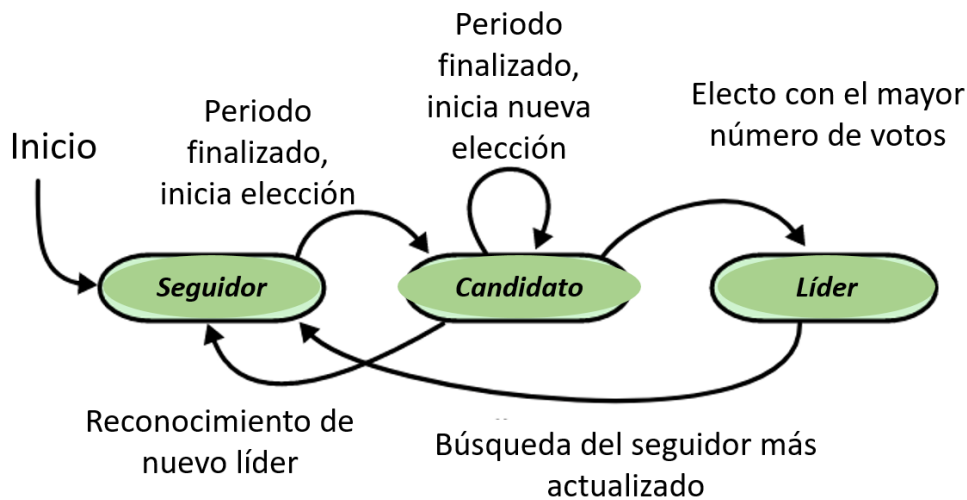


Figura 2.7. Algoritmo de consenso RAFT. [18]

RAFT es considerado como un algoritmo tolerante a fallos, por sus siglas en inglés CFT (*crash fault tolerant*) debido a que soluciona los problemas simples de concurrencia en nodos al utilizar los roles presentados en esta sección, por lo que resulta una alternativa muy atractiva para evitar las colisiones en la red debido al sobre flujo de información que pueden llegar a procesar los nodos involucrados. Sin embargo, en caso de que existiera algún nodo malicioso el algoritmo no es capaz de verificar ni validar la integridad de los datos por sí mismo. De ser así, la red presentaría un retraso considerable en el tiempo de validación de las transacciones, al verse incapaz de tomar una decisión en consenso entre los nodos maliciosos y los nodos confiables. Por lo que este tipo de implementación es muy

recomendable en un *quorum* previamente definido y establecido con dependencias conocidas por todos. Como pudiera ser un *blockchain* privado orientado a la toma de decisiones de una empresa, gobierno o entidad federativa.

Algunas de las plataformas más conocidas que adoptan RAFT como protocolo de consenso son: *Hyperledger Fabric*, *Hyperledger Iroha*, *Oracle*, *Hydrachain* y *BigchainDB*.

2.4 *Hyperledger Fabric*

Hyperledger Fabric es una plataforma permissionada de código abierto creada por *Linux Foundation* y diseñada como una base para desarrollar aplicaciones de nivel empresarial y soluciones industriales. Cuenta con controles de privacidad avanzados, por lo que solo que se comparten los datos que se asignen, y estos son distribuidos entre los participantes de la red una vez que hayan sido verificados y autenticados.

Fabric tiene una arquitectura altamente modular y configurable que permite la innovación, la versatilidad y la optimización para una amplia gama de casos de uso de la industria, incluidos banca, finanzas, seguros, atención médica, recursos humanos, cadena de suministro e incluso entrega de música digital [19].

Hyperledger Fabric admite la creación de contratos inteligentes codificados en lenguajes de programación de alto nivel, como Java, Go o Node.js. Así mismo hace uso de protocolos de consenso basados en voto por lo que no es necesario tener una criptomoneda nativa para soportar la creación de transacciones o impulsar la ejecución de contratos inteligentes. De esta manera, además de tener costo cero, incrementa el rendimiento de la red en términos de procesamiento de transacciones y latencia. Por estas razones es que en el presente proyecto se hará uso de esta plataforma. Así, a continuación, explicaremos los conceptos base de *Hyperledger Fabric*.

2.4.1 Contrato inteligente

Un contrato inteligente, también conocido como *smart contract* o *chaincode* en *Fabric*, es un archivo ejecutable que contiene la codificación de la lógica de negocio y que define los activos y las instrucciones necesarias para modificarlos. En otras palabras, es el archivo de configuración que se encarga de definir las reglas del sistema.

A diferencia de los contratos normales como los conocemos, estos se ejecutan de forma automática y dan como resultado, un conjunto de datos “clave-valor” que pueden ser enviados a la implementación y aplicado al libro mayor en formato JSON.

2.4.2 Nodos o *Peers*

Una red *blockchain* se compone principalmente de un conjunto de nodos o pares, también conocidos como *peers*. Estos son parte esencial en la infraestructura de una red distribuida debido a que se encargan de albergar copias de la información que se encuentra en el libro mayor, así como la definición de los contratos inteligentes.

2.4.3 Nodo de ordenación

El nodo de ordenación es fundamental en *blockchain* privadas, ya que al no contar con algún protocolo de consenso como *Proof of Work* o *Proof of Space*, se debe recabar la información de las transacciones que cada uno de los nodos envía al *ledger*, para posteriormente ser ordenada y propagada a toda la red.

La red de *Hyperledger Fabric* reduce el tiempo de propagación de transacciones a través de un nodo de anclaje (también conocido como nodo representante) el cual será el único encargado en cada organización de mantener contacto directo con el nodo de ordenamiento para la propagación y envío de nuevas transacciones dentro de su propia organización.

2.4.4 Organizaciones

Las organizaciones son grupos de nodos (*peers*), que son invitados a unirse a una red *blockchain* mediante un miembro proveedor de servicios, por sus siglas en inglés MSP, el cual se encarga de validar que las firmas y transacciones involucradas en los procesos de escritura y lectura en el libro mayor sean válidas.

Una organización puede ser tan grande como un corporativo multinacional o tan pequeña como un individuo.

2.4.5 Autoridad certificadora

En *Hyperledger Fabric*, la autoridad certificadora predeterminada se encarga de emitir garantías basadas en protocolos de llave pública a los que se les conoce como certificados raíz o certificados de inscripción. Estos certificados son otorgados a los nodos pertenecientes a la red una vez que forman parte de esta.

2.4.6 Canal

Los canales son una capa privada de comunicación entre organizaciones específicas que tienen la característica de llegar a ser asignadas a ciertos usuarios de la red. Cada canal consta de un *blockchain* separado en el cual solo los miembros de este son capaces de leer y escribir. A diferencia de los *peers*, nodos de ordenación y las autoridades certificadoras que conforman la infraestructura física de la red, los canales son procesos mediante el cual las organizaciones se conectan entre sí e interactúan.

2.4.7 Transacción

Una transacción es creada cuando se invoca la ejecución de algún contrato inteligente mediante una aplicación cliente con el objetivo de leer o escribir información en el *ledger*. Una vez que la transacción es endosada y aceptada por los participantes de la red, la entidad ordenadora se encarga de agregar la transacción dentro del libro mayor y enviarla a todos los nodos participantes mediante un mensaje tipo *broadcast*.

2.4.8 Funcionamiento de *Hyperledger Fabric*

Para llevar a cabo una transacción en la red se sigue el siguiente flujo, que consta de 3 etapas:

1. **Propuesta:** Una aplicación cliente envía una propuesta de transacción a un subconjunto de *peers*, los cuales ejecutarán el *smart contract* para producir una propuesta de modificación al *ledger*, la cual será endosada. Dicha propuesta regresará al cliente que inició la petición.

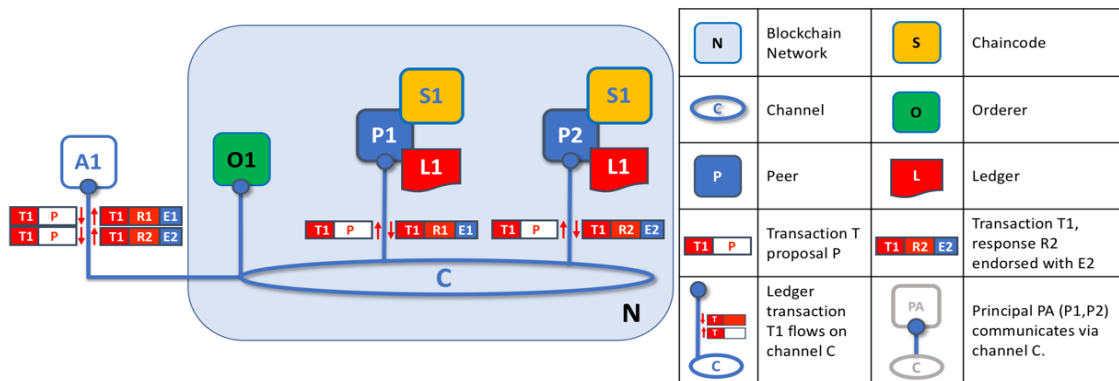


Figura 2.8 Diagrama de envío de una transacción (Propuesta). [27]

Como se puede apreciar en la Figura 2.8, la aplicación A1 genera la propuesta de transacción T1 y se las manda a los *peers* P1 y P2 en el canal C. P1 ejecuta el *smart contract* S1 con la propuesta T1, generando la transacción T1 con la respuesta R1 y el endoso E1. De manera independiente, P2 hace lo propio generando la transacción T2 con la respuesta R2 y el endoso E2. A1 recibe las dos transacciones de respuesta a T1 endosadas, E1 y E2.

2. **Ordenamiento y empaquetado de transacciones en bloques:** La aplicación cliente envía las transacciones endosadas a un nodo de ordenamiento (*ordering service*). Dicho nodo, crea los bloques de transacciones para ser distribuidos a todos los *peers* del canal.

Los nodos de ordenamiento reciben varias transacciones de forma concurrente de las aplicaciones clientes. Estos trabajan en conjunto para organizar las transacciones y empaquetarlas en bloques.

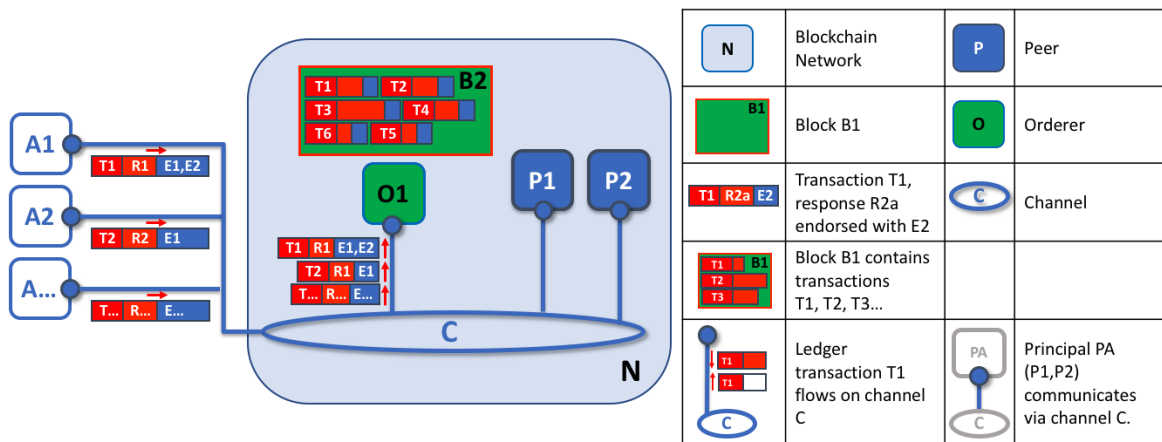


Figura 2.9 Diagrama de ordenamiento y empaquetado de transacciones. [27]

Como se puede ver en la Figura 2.9, el nodo de ordenamiento O1 recibe las transacciones de las aplicaciones clientes A1, A2, A... O1 empaqueta las transacciones recibidas en el bloque B2.

Del ejemplo anterior se extraen las siguientes ideas:

- El orden de las transacciones en el bloque propuestas por el *ordering service* se van a mantener en el libro mayor.
- El bloque propuesto por el *ordering service* es inmutable y, por tanto, no hay bifurcaciones en el libro mayor.
- Los nodos de ordenamiento no ejecutan contratos inteligentes.

3. **Validación y compromiso:** el nodo de ordenamiento distribuye los bloques a todos los *peer* conectados a él. Cada *peer* en el canal verifica que las transacciones del bloque sean válidas (es decir, que se haya endosado por los *peer* de la organización). Las transacciones que no son válidas se mantienen en el bloque inmutable, pero se actualizan en el estado del libro mayor.

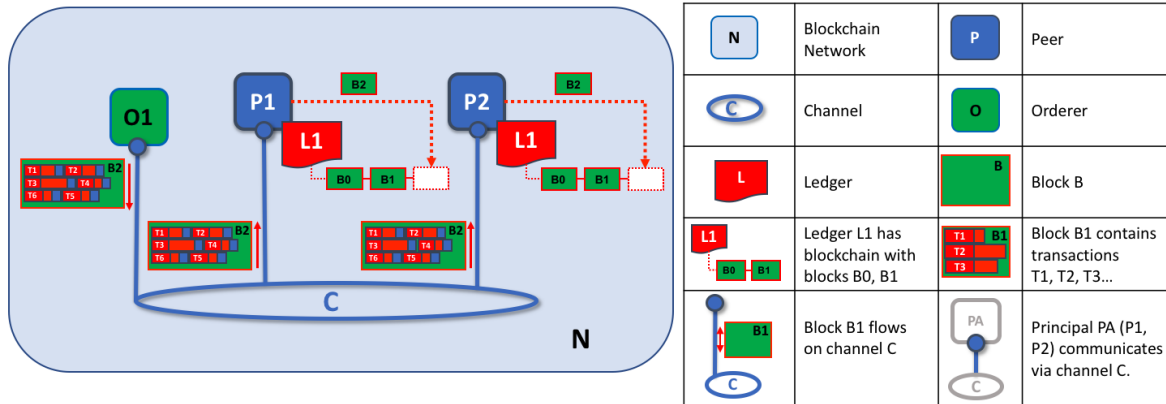


Figura 2.10 Diagrama de validación y compromiso en las transacciones. [27]

En la Figura 2.10 se puede apreciar como el nodo de ordenamiento distribuye los bloques a los *peers*. O1 distribuye el bloque B2 a los *peers* P1 y P2. El *peer* P1 procesa el bloque B2 y lo agrega al *ledger* L1, el *peer* P1 hace lo propio en su *ledger* L1. Al final, los *ledger* han sido actualizados y cada *peer* le debe informar a la aplicación cliente que la transacción ha sido procesada.

A todo este flujo de trabajo de la transacción en *Fabric* se le llama consenso. Las aplicaciones solo son notificadas cuando el ledger ha sido actualizado, es decir, hasta que el proceso ha finalizado.

El servicio de ordenamiento implementa Raft que es tolerante a fallos (CFT) y está basado en el protocolo Raft explicado en la sección 2.3.2.2. Se basa en el modelo de "líder y seguidores", se elige un líder por canal y sus decisiones son replicadas por sus seguidores.

2.5 Sistema de verificación vehicular actual

En el capítulo anterior se profundizó sobre el origen y las principales características que la tecnología *blockchain* le proporciona a cualquier sistema que base su funcionamiento en esta implementación. Así mismo se presentaron los diferentes tipos de *blockchain* (permisionado y no permisionado) que existen actualmente y se ejemplificaron los casos en donde es recomendado el uso de uno con respecto al otro. Posteriormente se citaron algunos de los protocolos de consenso más conocidos en la industria, así como el uso de herramientas que nos permiten la fácil integración de esta tecnología. Por último, se citaron algunas de las

aplicaciones más recientes e innovadoras que actualmente se encuentran en la industria resolviendo problemáticas de la vida diaria, las cuales basan su funcionamiento en el uso de la cadena de bloques.

En este capítulo se presenta un análisis detallado del funcionamiento del sistema de verificación vehicular al día de hoy, mencionando algunos acontecimientos que fueron relevantes a lo largo de su historia, ayudando a entender de mejor manera el porque es que surgieron los protocolos y procedimientos con los que actualmente se verifican las unidades automotrices. Posteriormente se presenta una discusión sobre las ventajas y desventajas del sistema actual, resaltando las vulnerabilidades que pueden ser explotadas en la actualidad para realizar actos de corrupción.

2.5.1 Sistema de verificación vehicular

El sistema de verificación vehicular como lo conocemos hoy en día tiene como principal objetivo el reducir al máximo las emisiones contaminantes de los vehículos en circulación, ya que en 1989 el entonces departamento del Distrito Federal comenzó a realizar estudios sobre la calidad del aire, señalando como principal causa de contaminación a los vehículos automotores.

Sin embargo, fue hasta 1993 cuando se comenzaron a operar los “Macrocentros de Verificación” bajo el estándar de revisión BAR 90, que se caracterizaba por el uso de dinamómetros⁵ y la medición de niveles de Dióxido de carbono (CO₂) y oxígeno (O₂).

De 1999 a 2000 se creó el Programa Integral de Reducción de Emisiones Contaminantes (PIREC), que hacía obligatorio el cambio de convertidores catalíticos en autos de modelos 1991 a 1996, para su mejor funcionamiento. Este modelo resultó defectuoso debido a irregularidades producto de la corrupción. Los talleres concesionados comenzaron a vender las facturas y el microchip sin cambiar el convertidor catalítico. Para evitar dichas irregularidades se instaló un nuevo sistema de cámaras de vídeo que incluye la transmisión de imágenes segundo a segundo. [23]

⁵ Dispositivo empleado para absorber o disipar la potencia generada por una máquina basándose en la medición de ciertos parámetros como el par torsional y la velocidad angular, es un equipo auxiliar en las pruebas de aceleración simulada (PAS), empleado para la medición objetiva de las emisiones vehiculares [30]

A partir de 2013 se crea el Centro de Inspección y Vigilancia Ambiental Remota (CIVAR) el cual es un sistema de monitoreo remoto de alta seguridad, a través de video vigilancia que tiene como principal objetivo el supervisar el correcto desarrollo de los procesos de verificación vehicular, a la par que se desarrolló el software de verificación, incorporando candados para el cifrado de datos. Fue hasta 2016, cuando se incorporó la evaluación del sistema de diagnóstico a bordo (SDB) y se realizó un ajuste en los límites de emisiones permitido.

2.5.2 Funcionamiento actual

A continuación, se describen los pasos a seguir para realizar el proceso de verificación vehicular en el Estado de México, centrándose únicamente en los vehículos particulares que utilizan gasolina como combustible, posteriormente se detectarán los puntos vulnerables a ser corruptibles.

Los vehículos registrados en el Estado de México deberán realizar y aprobar la verificación de emisiones vehiculares y revisión de componentes de control cada semestre, salvo los casos en que se obtenga un holograma doble cero “00”, en cuyo caso la unidad estará exenta de la obligación hasta por tres semestres [24].

Las unidades matriculadas en el Estado de México deberán continuar verificando conforme al color del engomado de circulación o al último dígito numérico de las placas de circulación del vehículo, como se puede apreciar en la Figura 2.11



Figura 2.11 Calendario de verificación vehicular en el Estado de México para el año 2022. [28]

Se deberá agendar una cita en la página [Cita verificación EDOMEX](#), ingresando los datos del auto, así como el día, hora y centro de verificación vehicular (CVV) de su preferencia. Posteriormente, deberá presentarse a la sucursal elegida con una identificación oficial, comprobante de la última verificación, tarjeta de circulación y asegurándose de no contar con infracciones de tránsito pendientes de pago.

2.5.2.1 Vehículos exentos

Existe una categoría de autos que son acreedores para quedar exentos del proceso de verificación vehicular, siempre y cuando cumplan con los criterios establecidos en la tabla 2.1.

Vehículos	Descripción	Vigencia
Eléctrico	Vehículo con una fuente de energía (eléctrica) donde la energía eléctrica es la fuente de propulsión principal sin combustión.	Permanente
Híbrido categoría I	Vehículo con dos fuentes de energía (eléctrica y gasolina) donde la energía eléctrica es la fuente de propulsión principal sin combustión y se obtiene desde una toma de corriente. La fuente de combustión interna se usa únicamente para alimentar el banco de baterías.	De 8 años, con posibilidad de renovación.
Híbrido categoría II	Vehículo con dos fuentes de energía (eléctrica y gasolina), en el cual la energía eléctrica permite la propulsión sin combustión, en periodos de operación en los que no se requiere máxima potencia.	De 8 años, por única ocasión. Aplica para unidades que ya tienen el holograma.

Tabla 2.1. Criterios necesarios para quedar exento del proceso de verificación

Si el vehículo no cumple con alguna de las consideraciones de la tabla anterior, es necesario someter la unidad a una serie de pruebas que serán mencionadas a lo largo este capítulo, con el objetivo de otorgar un holograma correspondiente a los resultados de dichas pruebas.

2.5.2.2 Holograma doble cero “00”

Los vehículos ligeros de uso particular tendrán 30 días naturales a partir de la fecha de expedición de la factura para realizar el proceso de verificación. Teniendo que someterse a la prueba SDB (Sistema de Diagnóstico a Bordo) la cual se encarga de detectar fallos eléctricos, químicos y mecánicos que pueden aumentar los niveles de emisión de partículas contaminantes. El sistema SDB permite conectar la computadora del vehículo con un

software dedicado que muestra de forma textual los componentes del auto que se encuentran deteriorados, así como los sensores que han dejado de funcionar en la unidad.

Así mismo las unidades deberán presentar y acreditar la prueba de verificación vehicular por emisiones. Si por alguna razón las lecturas del sistema SDB resultan negativas y no se acredita la prueba de verificación, el usuario tendrá otras dos oportunidades más para pasar este proceso, como se puede apreciar en la Figura 2.12.



Figura 2.12 Criterios de aceptación para holograma 00. [28]

2.5.2.3 Holograma cero “0”

Para obtener un holograma cero, los vehículos a gasolina, gas natural o gas licuado deberán cumplir los criterios de aprobación SDB. En caso de que el vehículo no acredite la prueba SDB, se llevará a cabo la prueba de verificación vehicular por emisiones y no deberá rebasar los límites máximos permisibles establecidos en la Tabla 2.2.

Hidrocarburos (HC) mol/mol (ppmh)	Monóxido de carbono (CO) cmol/mol (%)	Óxidos de Nitrógeno (NO _x) ⁽¹⁾ mol/mol (ppm)	Oxígeno (O ₂) cmol/mol (%)	Dilución (C+CO ₂) cmol/mol (%)		Factor Lambda
				Min.	Máx.	
80	0.4	250	0.4	13 7*	16.5 14.3*	1.03

Nota de equivalencias: ppmh, partes por millón referido al hexano.

Los óxidos de nitrógeno que se señalan en la presente Tabla no aplicarán en la prueba estática.

*Valores aplicados para vehículos automotores a gas.

Los vehículos cuyas características técnicas no hayan sido registradas por parte de las empresas que los fabrican o comercializan en el país, se verificarán con procedimiento estático.

Tabla 2.2. Límites máximos permisibles de emisiones de contaminantes para obtener holograma 0

Se tendrá dos oportunidades más para realizar la prueba al vehículo en dado caso que la primera prueba no sea exitosa, como se puede apreciar en la Figura 2.13.



Figura 2.13 Criterios de aceptación para holograma 0. [28]

2.5.2.4 Holograma uno “1”

A los vehículos a gasolina modelos 1994 a 2005 cuyos niveles de emisión no sobrepasen los límites establecidos en la Tabla 2.3 se les asignará la calcomanía uno.

Prueba	HC (ppm)	CO (%vol)	NOx (ppm)	CO+CO2 (%vol)		O2 (%vol)	Factor Lambda
				Min	Max		
Dinámica	100	0.7	700	13.0	16.5	2.0	1.03
Estática	100	0.5	No aplica	13.0	16.5	2.0	NA/1.03 Ralenti/crucero

Los vehículos cuyas características técnicas no hayan sido registradas por parte de las empresas que los fabrican o comercializan en el país, se verificarán con procedimiento estático.

Tabla 2.3. Límites máximos permisibles de emisiones de contaminantes para obtener holograma 1

2.5.2.5 Holograma dos “2”

Los vehículos a gasolina modelo 1993 y anteriores que cuenten con carburador y obtengan niveles inferiores a los de la Tabla 2.4 podrán portar el holograma número dos.

Prueba	HC (ppm)	CO (%vol)	NOx (ppm)	CO+CO2 (%vol)		O2 (%vol)	Factor Lambda
				Min	Max		
Dinámica	350	2.5	2000	13.0	16.5	2.0	1.05
Estática	400	3.0	NA	13.0	16.5	2.0	NA/1.05 Ralentí/crucero

Los vehículos cuyas características técnicas no hayan sido registradas por parte de las empresas que los fabrican o comercializan en el país, se verificarán con procedimiento estático.

Tabla 2.4. Límites máximos permisibles de emisiones de contaminantes para obtener holograma 2

El ciudadano tendrá la oportunidad de volver a realizar la prueba dos ocasiones más, tanto para holograma uno y dos, como se puede apreciar en la Figura 2.14

	PRIMER INTENTO		SEGUNDO INTENTO		TERCER INTENTO	
	OBDDII	EMISIONES	OBDDII	EMISIONES	OBDDII	EMISIONES
HOLOGRAMA 1 	N/A	 (700 NOx)	N/A	 (700 NOx)	N/A	N/A
HOLOGRAMA 2 	N/A	 (2000 NOx)	N/A	 (2000 NOx)	N/A	N/A

Figura 2.14 Criterios de aceptación para holograma 1 y 2. [28]

2.5.2.6 Técnica de verificación de no aprobación (Rechazo técnico)

La obtendrán aquellos vehículos que en el proceso de verificación por SDB presenten algún código de falla del tren motriz asociado a alguno de los monitoreos indicados como obligatorios, al mismo tiempo que no cumplan con los niveles establecidos en las Tablas 2.2,2.3, y 2.4 de límites permisibles.

2.5.2.7 Criterios de aprobación SDB

El criterio de aprobación SDB se realizará para las unidades que aspiren a obtener holograma tipo “0” o “00”. Se considera que un vehículo automotor aprueba el método de prueba a través del Sistema de Diagnóstico a Bordo (SDB) si cumple con todos los criterios de aprobación señalados a continuación:

- Lograr comunicación con la ECU ⁶ del vehículo automotor a través del dispositivo de adquisición de datos para SDB.
- Contar con los monitores obligatorios por tipo de SDB conforme a la NOM-167-SEMARNAT-2017.
- Que todos los monitores obligatorios no indiquen “no completado”, “not ready” o “no listo”.
- No presentar códigos de falla del tren motriz asociados a alguno de los monitores indicados como obligatorios. [24]

2.5.2.8 Organización y gestión interna en los CVV

De acuerdo a la información establecida en el manual de procedimientos para la verificación automotriz en el Estado de México, cada una de las unidades verificadoras, deberán considerar al personal suficiente para cubrir las necesidades de su servicio, sin embargo, la Tabla 2.5 muestra una plantilla del personal mínimo necesario para el óptimo funcionamiento del establecimiento.

PERSONAL	CANTIDAD
Director General	1
Gerente de Operaciones	1
Gerente Técnico	1
Auxiliar Administrativo	1
Administrador de Hologramas	1
Administrador de Centro	1
Técnico Verificador	3
Total	9

Tabla 2.5 Lista de personal mínimo necesario para el correcto funcionamiento de un CVV

⁶ Por sus siglas en ingles la Unidad de Control del Motor es un sistema embebido que se encarga de monitorear y controlar diversas funciones del motor para obtener el mejor desempeño del mismo.

El presente trabajo se enfocará en profundizar únicamente en los procedimientos y obligaciones de los técnicos, que a su vez se dividen en: técnico capturista, técnico verificador y técnico de impresión.

1. Técnico capturista:

Es la persona que recibe al cliente a su llegada, le solicita los documentos necesarios para la realización del trámite (tarjeta de circulación, holograma y certificado del vehículo), revisa que la unidad no cuente con multas pendientes para posteriormente invitar a pasar al cliente a la sala de espera.

Luego en el sistema:

- Captura y verifica todos los datos del vehículo
- Digitaliza la documentación solicitada al cliente
- Asigna el número de línea en donde se realizará la verificación

2. Técnico verificador:

Encargado de realizar la verificación mediante escaneo SDB, prueba dinámica, estática u opacidad. Al recibir el vehículo deberá hacer una inspección visual sobre los siguientes puntos:

- Prueba de humo
- Tapón de gasolina
- La bayoneta del aceite
- Filtro de aire
- El tubo de escape
- El tapón del radiador
- Mangueras de vacío
- El filtro del canister
- Sistema de ventilación desgaste y abombamiento de ruedas
- Tracción del vehículo y odómetro.
- Validar que la unidad no cuente con multas pendientes de pago.
- Validar los datos, placas y hologramas anteriores; y verificar la asignación de la línea

Posteriormente deberá ingresar el vehículo a la línea correspondiente en donde se realizará:

- La prueba de humo: Se acelerará el vehículo a 24 km/h por un tiempo de 30 segundos, para posteriormente identificar si el vehículo arroja humo negro o azul.
- La prueba de carga: Se colocará la pipeta en el escape, posteriormente se acelerará el vehículo a 24 km/h para aplicarle la prueba PAS 5024 ⁷, durante 60 segundos.
- La prueba de velocidad: Se acelerará el vehículo a 40 km/h para aplicarle la PAS 2540 ⁸, durante 60 segundos.

Una vez realizado el proceso, el técnico revisará en la computadora que los datos de emisión hayan sido enviados.

Cabe resaltar que entre sus obligaciones se encuentra el calibrar la línea de verificación cada 24 horas (prueba de fugas, dinamómetro y analizador de gases), así como la celda de carga con la prueba de dinamómetro, verificar que tanto la sonda como las pipetas no tengan fugas y abrir la llave de seguridad del gas para aplicar la calibración del equipo.

3. Técnico de impresión:

Es el encargado de verificar la información final de los niveles de contaminación de cada unidad. Así mismo se encarga de imprimir los resultados para su valoración y posteriormente la asignación de holograma a los vehículos según los criterios mencionados en las tablas 2.2, 2.3 y 2.4.

⁷ Ciclo de prueba de un vehículo en dinamómetro utilizando la prueba de aceleración simulada (PAS), en donde a una velocidad constante de 24 Km/h, se aplica una carga externa al motor equivalente al 50 por ciento de la potencia requerida para acelerar al vehículo a una tasa de aceleración de 5.6 kilómetros por hora por segundo. [24]

⁸ Ciclo de prueba de un vehículo en dinamómetro utilizando la prueba de aceleración simulada (PAS), en donde a una velocidad constante de 40 km/h, se aplica una carga externa al motor equivalente al 25% de la potencia requerida para mantener esta velocidad bajo condiciones reales de manejo. [24]

2.5.3 Discusión sobre el procedimiento actual

Si bien el procedimiento de verificación vehicular ha logrado mejorar a lo largo de su corta historia con ayuda de los avances tecnológicos, en esta sección hablaremos de los puntos más vulnerables que existen hoy en día en dicho sistema actual. A partir de estos puntos se genera una gran brecha de oportunidades para realizar acciones fraudulentas o de corrupción.

En un principio, el gobierno del entonces Distrito Federal, comenzó a detectar este tipo de acciones en la mayoría de los centros de verificación vehicular, ya que como mencionó en secciones anteriores, a partir del año 2000 con el inicio del Programa Integral de Reducción de Emisiones Contaminantes (PIREC), el cual hacía obligatorio el cambio de convertidores catalíticos en autos, los talleres concesionados comenzaron a vender las facturas y el microchip sin cambiar el convertidor catalítico. Esto obligó al sistema a implementar medidas de vigilancia mediante cámaras de vídeo que incluían la transmisión de imágenes segundo a segundo. Sin embargo, esta implementación que actualmente se encuentra en funcionamiento, carece de efectividad al verse rebasado por el número total de unidades verificadoras en el Estado de México (138) multiplicado por el número de carriles promedio (4) que existen en cada uno de estos CVV, lo cual nos da como resultado un total de 552 cámaras de video vigilancia aproximadamente que deberán ser monitoreadas durante las 11 horas diarias de su operación, lo que resulta un sistema muy costoso que no garantiza el cumplimiento de los procedimientos establecidos por el gobierno del Estado de México.

Desafortunadamente la corrupción que existe en los verificentros es un “mal sabido por todos”, y se lleva a cabo de manera colectiva entre los integrantes y personas externas al CVV. Lo que nos hace plantearnos las siguientes preguntas: ¿Existe alguna dependencia que se encargue de auditar los procesos realizados en cada una de las unidades verificadoras? De ser así, el sistema actual, ¿permite el correcto análisis de la información ingresada al sistema?, ¿existe alguna garantía de que la información presentada no haya sido manipulada por algún trabajador con anterioridad?, ¿se cuenta con algún control en el inventario y asignación de estampados? A lo largo de esta sección se indagará sobre estos cuestionamientos.

Como se mencionó en secciones anteriores, el programa de verificación vehicular nació de la necesidad de controlar y reducir los niveles de contaminación en el aire, por lo que la SEDEMA planeó y desarrolló el sistema que actualmente se encuentra en vigor en el

Estado de México. Así mismo la Dirección General de Prevención y Control de la Contaminación Atmosférica (DGPCCA) es la encargada de llevar a cabo auditorías en los más de 100 CVV.

Según el informe de auditoría número 023-0013-2019 [25] realizado en el estado Mexiquense entre el periodo de enero y junio de 2019, se llevó a cabo una inspección en 118 CVV que tenían como objetivo los siguientes puntos:

- Verificar la existencia y cumplimiento de un Programa Anual de Trabajo.
- Verificar la existencia de procedimientos formalmente establecidos.
- Verificar la existencia y efectividad de los controles internos.
- Verificar aleatoriamente la integración de expedientes.
- Verificar la realización de visitas técnicas de supervisión. [25]

El documento completo relata de forma superficial las acciones realizadas en los verificentros, así como resultados, observaciones y acciones por mejorar. Dicha auditoría es la más reciente que existe actualmente y se enfoca en su mayoría en la parte técnica del sistema, verificando cuestiones de calibración en los equipos, uso de formatos específicos para almacenar información, así como el seguimiento a los procedimientos ya establecidos en la gaceta oficial, dejando completamente de lado las operaciones ya realizadas en dichos CVV.

La conclusión del informe se cita textualmente a continuación: “Como resultado de la aplicación de los procedimientos establecidos para la auditoría, se determinó que durante el periodo del 01 de enero al 30 de junio de 2019, la Dirección General de Prevención y Control de la Contaminación Atmosférica, programó y realizó visitas técnicas de supervisión a Centros de Verificación Vehicular autorizados en el Estado de México, en las cuales se observaron deficiencias administrativas para su atención y en los instrumentos de control utilizados para evidenciar los resultados obtenidos, ya que las solicitudes de visitas técnicas para la apertura de líneas, no se atienden en el orden de su recepción y el personal que realiza las visitas, utiliza un formato de acta diferente a la que se establece en el oficio de comisión; por lo que se generaron 2 observaciones de Control Interno y se convinieron con la Directora General de Prevención y Control de la Contaminación Atmosférica las acciones de mejora orientadas a fortalecer control interno y los aspectos legales de las visitas técnicas de

supervisión a Centros de Verificación Vehicular autorizados en el Estado de México, mismas que se detallan en el apartado de resultado del presente informe.”[25]

De las conclusiones anteriores, se puede deducir lo siguiente:

1. No existen auditorías internas que verifiquen la correcta asignación de hologramas en semestres anteriores, este tipo de visitas son exclusivamente de carácter técnico y tienen como objetivo el validar que la maquinaria utilizada en cada una de las líneas de verificación se encuentre en buenas condiciones para su correcto funcionamiento.
2. En el sistema actual, el técnico capturista es el encargado de asignar una línea de verificación a cada vehículo que llega al establecimiento, sin embargo, este número de línea no es almacenado de forma correcta debido a la falta organización en el registro de autos, por lo que le resulta imposible al auditor detectar errores en los CVV.

El segundo punto es vital para entender que existe una brecha de seguridad en el sistema, ya que fácilmente puede ser saboteado por los mismos integrantes del verificentro. En el supuesto caso en el que una de las líneas de verificación se encuentre en mal estado debido a que el técnico verificador no calibró los instrumentos necesarios para la toma de mediciones, esta línea es propensa a capturar niveles erróneos de contaminación en los automóviles, lo cual puede ser aprovechado por los trabajadores del CVV para generar reportes exitosos en autos que no necesariamente cumplan con todos los requisitos establecidos en las tablas 2.2, 2.3 y 2.4 para la obtención de algún holograma. Así mismo, el técnico de impresión tiene la facultad de validar y/o modificar los datos obtenidos para así imprimir el reporte generado y posteriormente otorgarle un estampado al vehículo, según sus niveles de contaminación registrados.

El carecer de auditorías que verifiquen y validen los procedimientos realizados en cada uno de los verificentros, obliga al sistema completo en sustentar su funcionamiento en la honestidad y honradez de tres figuras principales (técnico capturista, técnico verificador y técnico de impresión) para el correcto funcionamiento de cada verificentro, por lo que el sistema actual entra en la categoría de sistema centralizado propenso a fallas.

Capítulo 3. Estado del Arte

La relevancia que tiene *blockchain* hoy en día, ha ido aumentando conforme a la necesidad de las personas y organizaciones por mantenerse conectados, interactuar con sus clientes, automatizar sus procesos y generar confianza en los procedimientos que habitualmente solían hacerse en oficinas de gobierno, bancos o tiendas departamentales.

Generalmente cuando se habla de implementaciones *blockchain* en la vida cotidiana, rápidamente se relaciona esta idea con el área de las criptomonedas como lo son *bitcoin*, *ethereum*, *cardano*, entre otras. Sin embargo, el concepto de criptomoneda es solo uno de los cientos de aplicaciones que se han desarrollado con base a las características que la cadena de bloques nos proporciona.

Es de resaltar que el uso de *blockchain* se ha enfocado en mejorar los procedimientos con los que a diario convivimos, aportando una serie de beneficios como trazabilidad, seguridad, privacidad, transparencia, confianza entre otras características mencionadas en secciones anteriores. Por lo que algunas empresas transnacionales líderes en tecnología como Amazon, IBM y Oracle, se han dedicado al desarrollo de módulos especializados en aplicaciones *blockchain*.

En las siguientes secciones, se citarán algunos ejemplos de aplicaciones que se han sido considerablemente beneficiadas por *blockchain* y que actualmente ayudan a solucionar problemas de la vida cotidiana en sus respectivas áreas de estudio.

3.1 Proyecto XCEED en plantas automotrices de Renault Europa

El proyecto XCEED, es una iniciativa de la compañía Renault en colaboración con IBM para poder certificar la calidad de los componentes de cada vehículo, desde el diseño, producción y distribución.

Esto con ayuda de la creación de una red confiable que les provee la capacidad de compartir información del cumplimiento de las normativas entre los fabricantes de piezas a lo largo de la cadena de suministro, utilizando la tecnología *Blockchain*, *Big Data* e Inteligencia Artificial es posible analizar y aprobar las más de 6000 características

reglamentarias que son necesarias para la creación de un vehículo, ya que de esto depende la aprobación legal del cumplimiento de las normativas y características como:

- Características de seguridad y reglamentarias.
- Características geométricas.
- Características de los materiales.

El principal objetivo de XCEED es proporcionar una plataforma de trazabilidad en el cumplimiento de normativas en todo el ecosistema de la industria automotriz europea, aumentando la productividad del 15% al 30% para los servicios en cuestión, una reducción del 10% en los costos relacionados con la gestión del incumplimiento en las normativas de calidad locales. [33]

Debido a las políticas de privacidad de las empresas relacionadas, no se detalla de forma técnica el procedimiento y las características específicas del sistema, sin embargo logramos dimensionar que la necesidad de intercambiar información en tiempo real, proporcionando transparencia y trazabilidad son requerimientos que aumentarán su demanda en los próximos años. *Blockchain* tiene futuro y Renault lo ha entendido desde 2015.

3.2 Garantías Bancarias con Lygon 1B

El proyecto Lygon 1B nace de la necesidad de digitalizar el proceso mediante el cual inquilinos, bancos y propietarios realizan la negociación de garantías bancarias. Es decir, cuando un minorista quiere arrendar una propiedad para abrir una tienda, el propietario requiere una garantía bancaria que diga: "Eres bueno para el alquiler". Hasta la fecha, este proceso ha sido realizado a través de docenas de contratos en papel que aumentan el tiempo de espera para los propietarios e inquilinos.

Los documentos en papel son propensos a la falsificación, así mismo los propietarios deben almacenar estos documentos en cajas fuertes y consolidarlos manualmente con los arrendamientos de propiedades, sin menciona que los bancos pierden de vista el documento en el momento en que se emite. Y para los clientes finales, el proceso es engorroso.

Es por esto que ANZ, Westpac, Commonwealth Bank, IBM Australia y la empresa de administración de propiedades Scentre Group, desarrollaron una plataforma de cadena de

bloques en la nube que administra el proceso antes mencionando, garantizando en todo momento la seguridad y transparencia de los datos.

La plataforma realiza los trámites de forma digital teniendo procedimientos estandarizados, datos inmutables y trazabilidad en las transacciones, lo que facilita el seguimiento a los trámites relacionados, aportando una velocidad excepcional. En el proyecto piloto, lo que solía tomar hasta un mes, tomó menos de un día en llevar a cabo el trámite completo, esto significa una mejora del 90-95%. [34]

Debido a que los procesos de garantía son similares en casi todos los países, si se lograra estandarizar y digitalizar documentos para posteriormente formar parte de una cadena de bloques, se podría desarrollar una herramienta unificada y funcional en cualquier parte del mundo.

3.3 IBM *Food Trust*

Gracias a los avances de transporte, comunicación, acuerdos de comercio y la globalización en general, es posible abastecer de productos primarios a la población en casi cualquier parte del mundo, sin importar las condiciones climáticas, tipo de suelo o la fauna que radica en los países destino.

Es por esto que se puede tener acceso a frutas, verduras, legumbres, cereales, etc en cualquier temporada del año gracias a la importación y exportación de productos primarios, lo cual genera una gran cantidad de información que IBM en colaboración con Walmart ha logrado integrar en su plataforma llamada *IBM Food Trust*, la cual tiene como principal objetivo el generar confianza entre los consumidores, sobre el origen y procesamiento de los productos que a diario encuentran en tiendas y supermercados.

La herramienta permite a todos los involucrados el poder añadir información detallada sobre el origen, técnica de riego, tipo de transporte y tiempo en bodega de la materia prima, con el objetivo de crear y mantener un registro de los productos involucrados desde su cultivo en el campo hasta su venta en supermercados, logrando involucrar todos los procesos relacionados y generar un histórico que ayudará a fortalecer la confianza entre los

consumidores al tener acceso a este tipo de información al momento de adquirir dichos alimentos.

IBM Food Trust basa su funcionamiento en un blockchain tipo *permissioned*, permitiendo a todas las personas involucradas, el poder acceder al *ledger* y escribir la información detallada de cada uno de los procesos por los que ha pasado el producto. Sin embargo, por el tipo de blockchain utilizado en esta herramienta, existe una capa de seguridad entre los involucrados y la cadena de bloque completa para cada uno de los productos en cuestión, esto quiere decir que los agricultores, transportistas, dueños de bodegas y tiendas de autoservicio únicamente podrán escribir su información correspondiente, sin saber el origen previo o destino del producto. Otorgándole al cliente final el acceso a consultar el histórico completo del alimento que está a punto de adquirir.

Algunas de las ventajas que proporciona *Food Trust* son:

- Generar confianza en la industria agropecuaria.
- Aumentar la eficiencia de procesos, reduciendo las pérdidas de productos perecederos.
- Garantizar la calidad de los bienes vendidos.
- Empoderar a los productores, proveedores y consumidores minoristas.

Para el correcto funcionamiento de la red *blockchain* en la implementación *Food Trust* a nivel mundial, IBM propuso albergar la infraestructura en su sistema de nube, con las siguientes reglas de negocio a seguir:

- Dentro del *blockchain* existen *peers* de confianza que tienen la responsabilidad de monitorear la integridad del *ledger* en todo momento, otorgar permisos a usuarios y solucionar posibles incidentes de seguridad [22].
- Cada uno de los integrantes de la red (*peer*) tiene la capacidad de consumir los *smart contract* propuestos en la implementación, con miembros previamente registrados en la red.

- Los contratos inteligentes se ejecutan en canales privados donde solo las partes interesadas tienen acceso a los datos [22].

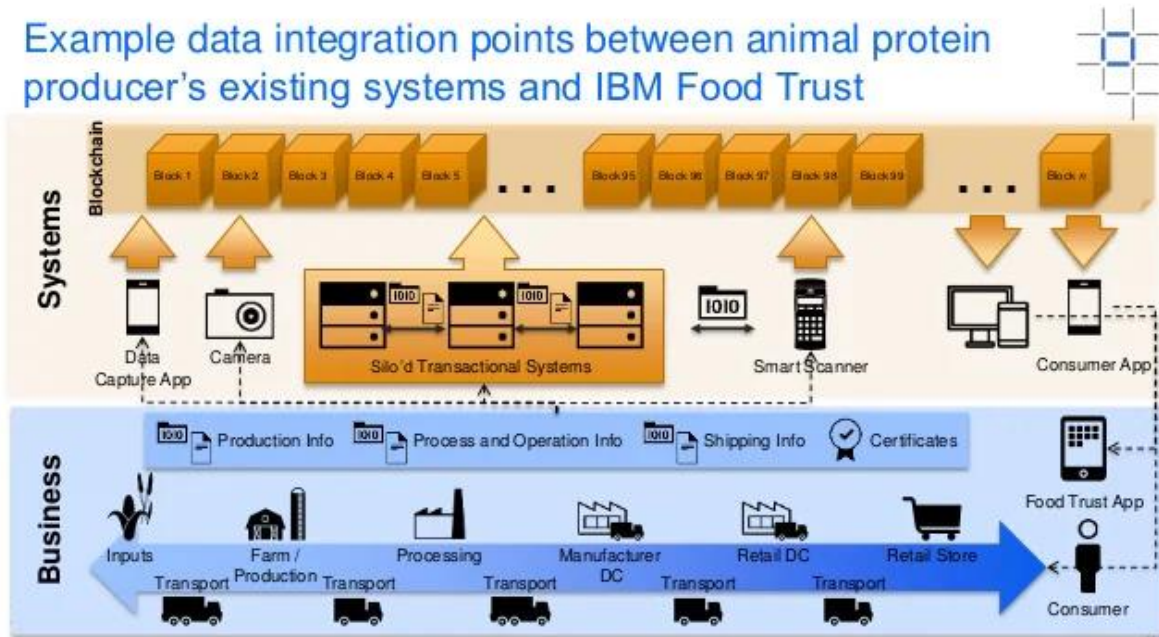


Figura 3.2 Proceso de negocio relacionado con *IBM Food Trust* [22].

Como se puede apreciar en la Figura 3.2, el proceso de negocio que sigue un producto agropecuario para poder llegar a los supermercados y tiendas de conveniencia depende de varios proveedores que prestan sus servicios para poder transportar, procesar, manufacturar y distribuir estos alimentos. Al mismo tiempo, cada uno de los proveedores se encuentra en constante interacción con la cadena de bloques la cual se va actualizando conforme el flujo completo del producto. Permitiendo al final que el consumidor pueda acceder a los datos generados en cualquier parte, o etapa del sistema.

Capítulo 4. Sistema de Verificación Vehicular basado en *Blockchain*

Con base en el análisis del procedimiento actual en los CVV presentado en la sección anterior, en este capítulo se propone mejorar el flujo completo de verificación mediante el uso de *blockchain*. Esto ayudará a generar confianza entre los usuarios a partir de las características ya mencionadas que aporta esta tecnología, beneficiando directamente a los ciudadanos que bimestralmente cumplen con sus obligaciones vehiculares.

Iniciando por la descentralización de los procesos internos en cada uno de los verificentros, a partir de una *API rest*⁹ que servirá como interfaz gráfica para la interacción con el *ledger* de *blockchain*, ayudando a recabar y almacenar la información necesaria para poder fundamentar la toma de decisión sobre los resultados obtenidos en cada uno de los vehículos. Partiendo del supuesto en el que los dinamómetros encargados de la toma de mediciones de partículas contaminantes siempre se encuentren calibrados y generando mediciones correctas (proceso que le compete a la Entidad Mexicana de Acreditación ‘EMA’), se podrá disminuir la colusión entre los trabajadores de las unidades verificadoras, evitando actos de corrupción.

Para poder llevar a cabo esta propuesta, será necesario integrar los 138 CVV del Estado de México en una red de comunicación exclusiva para esta implementación, los cuales serán los encargados de resguardar la seguridad e integridad de las transacciones realizadas, así como de los datos almacenados en esta implementación.

En las siguientes secciones se profundiza sobre las características y consideraciones que son necesarias para la implementación de esta aplicación.

⁹ Interfaz de programación de aplicaciones por sus siglas en inglés API, es un conjunto de reglas que determinan cómo las aplicaciones o los dispositivos pueden conectarse y comunicarse entre sí. [31]

4.1 Arquitectura

El objetivo principal de la aplicación presentada es lograr descentralizar los procedimientos que son propensos a ser corruptibles en cada uno de los CVV, además de que la misma tecnología *blockchain* aportará una serie de características extra que pueden ser explotadas de forma conveniente para el mejoramiento del proceso. Algunas de estas características serán implementadas en la solución propuesta y otras se mencionan como trabajos futuros.

Estableciendo una analogía entre los conceptos propios de *Hyperledger Fabric* y los centros de verificación vehicular, se definen los componentes presentados en la Tabla 4.1. En ella se puede apreciar la abstracción del sistema de verificación a conceptos propios de *blockchain*, resaltando el uso de las organizaciones como cada uno de los establecimientos físicos en donde el ciudadano puede acudir a realizar la inspección de su unidad, los pares son representados como trabajadores en cada una de estas organizaciones, el canal compartido es una red de comunicación en el que todas las organizaciones podrán enviar y actualizar la información de libro mayor.

<i>Hyperledger Fabric</i>	Centros de verificación vehicular
Organizaciones	Cada uno de los 138 CVV
Pares (<i>peer</i>)	Cada uno de los trabajadores
Canal	La red de comunicación interna entre todos los CVV
<i>Orderer (Chaincode)</i>	Lógica de negocio en los CVV

Tabla 4.1: Analogía entre la terminología *Hyperledger Fabric* y los CVV

Con base a los componentes descritos anteriormente, se propone la topología presentada en la Figura 4.2 como propuesta para la implementación de nuestro sistema. En ella podemos ver que se reduce el número de verificentros de 138 a 10 CVV, esto con el objetivo de ejemplificar de forma sencilla el funcionamiento de la propuesta. Así mismo se puede apreciar que todas las organizaciones (CVV) ya se encuentran en un canal de comunicación con el *chaincode*, el cual permite la conexión con el resto de los verificentros. Es de resaltar que todas las organizaciones tienen previamente almacenado el contenido del *chaincode* de forma local, por lo que si existiera algún cambio en la lógica de negocio o

alguna organización quisiera alterar el chaincode, éste deberá ser propuesto para su validación por todas las organizaciones existentes. Posteriormente deberá ser almacenado y ejecutado de forma local en cada CVV, lo cual le proporciona una capa de seguridad al sistema, evitando la alteración de la lógica que se presentará en secciones posteriores.

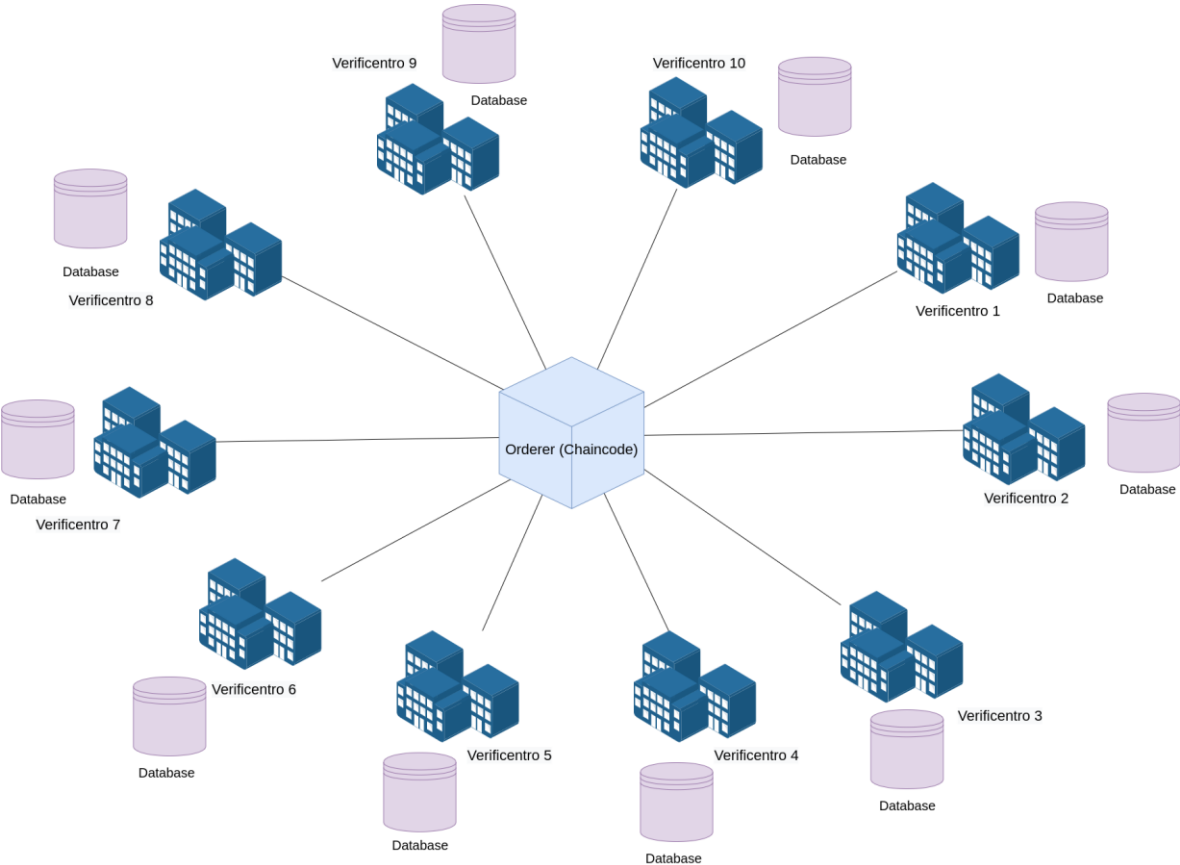


Figura 4.2: Topología de red *blockchain*, propuesta para 10 CVV

Otro punto por resaltar en la Figura 4.2 es el uso de una base de datos independiente en cada organización, lo cual le permitirá acceder a la información almacenada en el *ledger* de forma eficiente. Recordando que la propia implementación *Blockchain* se asemeja a una base de datos distribuida, en la que cual la información que se genera es alojada en un libro mayor que a su vez, se replica en cada uno de los nodos y en sus respectivas bases de datos. Es por esto que, únicamente se pueden realizar consultas al *ledger* sin tener la posibilidad de alterar datos mediante sentencias SQL.

4.2 Implementación

Una vez establecidos los componentes que formarán parte de la red *blockchain*, se detallará de forma técnica los pasos a seguir, así como las herramientas utilizadas para la implementación del sistema. Iniciando por mencionar que la implementación se ejecuta en un entorno Linux, en específico en la distribución Oracle Linux 8.2. El *software* complementario se describe a continuación:

- Docker:

La herramienta docker en su versión 20.10.7 es un *software* encargado de generar contenedores. De manera similar a cómo una máquina virtual se encarga de virtualizar el *hardware*, los contenedores virtualizan aplicaciones o servicios para su ejecución en cualquier ordenador [26].

Dentro de la familia Docker, existe docker-compose el cual se encarga de definir y ejecutar aplicaciones Docker de contenedores múltiples, utilizando un archivo con extensión yaml para iniciar y configurar todos los servicios de la aplicación.

- Go

Es un lenguaje de programación que se caracteriza por ser concurrente y compilado. Inspirado en la sintaxis de C pero muy semejante al dinamismo que Python provee. Fue desarrollado por Google y hoy en día, es ampliamente recomendado para aplicaciones *blockchain*, gracias al excelente rendimiento que provee al ejecutar contratos inteligentes de forma rápida y eficiente.

Go es ideal para la codificación de la lógica de negocio en sistemas que requieran robustez por lo que, en las siguientes secciones, se presentará parte de la codificación del *chaincode* que se implementa en el presente trabajo.

- CouchDB

Es un gestor de base de datos de código abierto que se caracteriza por su fácil implementación de un sistema NoSQL al emplear JSON para almacenar los datos y Javascript como lenguaje de consulta por medio del protocolo HTTP tipo un API.

A diferencia de las bases de datos convencionales, CouchDB no almacena datos y relaciones en tablas. Cada base de datos es una colección de documentos independientes los cuales cuentan con sus propios esquemas.

4.2.1 Configuración de autoridades certificadoras

El primer paso para montar la red *blockchain*, es generar las entidades certificadoras para cada una de las organizaciones, por lo que se diseñó un archivo tipo YAML que alberga las características necesarias para la creación de los contenedores en los 10 CVV.

A continuación, en la Figura 4.3 se muestra el fragmento de código para la creación del contenedor que alberga la autoridad certificadora de la organización uno.

```
ca_org1:
  image: hyperledger/fabric-ca
  environment:
    - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
    - FABRIC_CA_SERVER_CA_NAME=ca.org1.example.com
    - FABRIC_CA_SERVER_TLS_ENABLED=true
    - FABRIC_CA_SERVER_PORT=7054
  ports:
    - "7054:7054"
  command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
  volumes:
    - ./fabric-ca/org1:/etc/hyperledger/fabric-ca-server
  container_name: ca.org1.example.com
  hostname: ca.org1.example.com
  networks:
    - test
```

Figura 4.3: Creación de contenedor y Autoridad Certificadora

Del fragmento de código, se puede resaltar el uso de las variables de entorno necesarias para distinguir una autoridad certificadora del resto, así como el uso y asignación de un puerto específico en donde el servicio se encontrará a la espera de peticiones. Así mismo se asigna el *hostname* y el nombre del contenedor.

De forma paralela, se crea la estructura de archivos de la Figura 4.4 que son necesarios para el almacenamiento de los certificados, llaves públicas y privadas de cada una de estas entidades, las cuales se utilizarán para poder firmar y acreditar transacciones dentro del *ledger*.

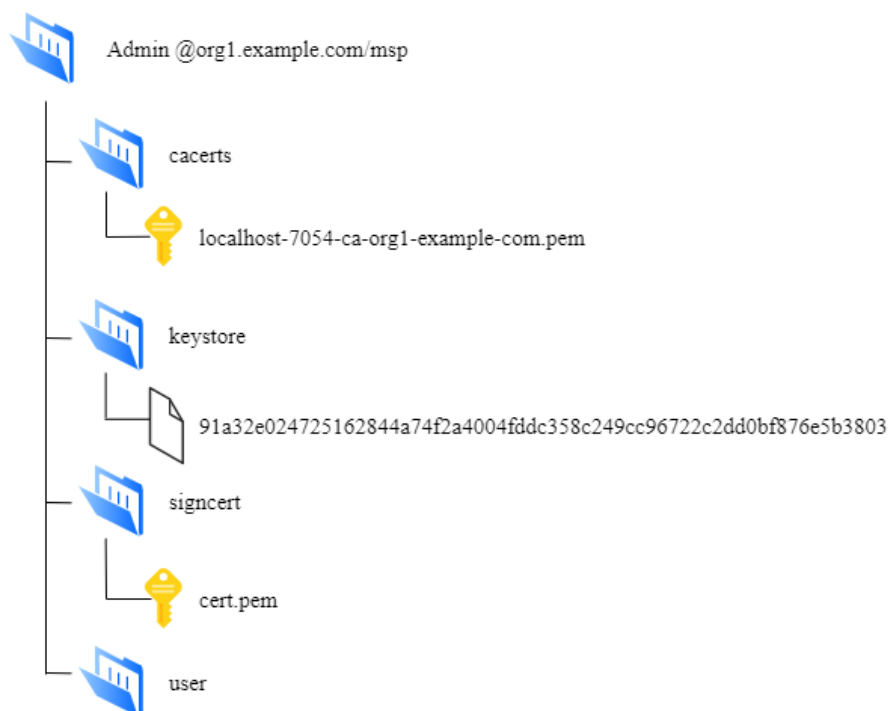


Figura 4.4: Sistema de archivos donde se albergan los certificados y llaves publicas/privadas

Como se puede apreciar en la Figura 4.4, se crean dos tipos de certificados, el primero de ellos se encuentra alojado en la carpeta “cacerts” y es propio de la autoridad certificadora para la organización uno, el cual se encarga de validar que dicha organización sea parte de del consorcio establecido en la red *blockchain*. El segundo certificado se encuentra en la carpeta “signcerts” y es mediante el cual se firman las transacciones realizadas por la presente organización, para posteriormente formar parte del *ledger*.

4.2.2 Creación del bloque génesis

La creación del primer bloque, también conocido como “bloque génesis”, es una de las tareas más relevantes que deben ejecutarse antes de comenzar a utilizar implementaciones *blockchain*, ya que se encarga de inicializar una transacción que debe ser reconocida por

todas las organizaciones perteneciente a la red para indicar el punto de partida de la cadena, así como el *hash* que permita validar y agregar nuevas transacciones al *blockchain*.

Otra de las tareas que se realizan en este paso, es la creación de contraseñas y variables de entorno necesarias para que los *peers* inscritos, formen parte del consorcio en la red privada de *blockchain*.

4.2.3 Creación de contenedores para base de datos (CouchDB)

La implementación de *Hyperledger Fabric* proporciona a la red una capa de accesibilidad a los datos de forma eficiente lo que permite realizar consultas al *ledger* sin necesidad de recorrer toda la cadena de bloques en búsqueda de un elemento o una serie de elementos en específico, esto con ayuda de CouchDB y la creación de contenedores mediante Docker, que simularán la implementación de “n” número de nodos equivalente al número de organizaciones en la red.

Como se puede apreciar en la Figura 4.5, se establecen los parámetros necesarios para la creación de un contenedor Docker que alojará el servicio CouchDB para cada una de las organizaciones, en específico para la organización uno. Definiendo el puerto 5984 que se mantendrá a la espera de peticiones, el nombre del contenedor y la red a la cual estará asociada esta implementación.

```
Couchdb0:  
  container_name: couchdb0  
  image: couchdb:3.1.1  
  environment:  
    - COUCHDB_USER=admin  
    - COUCHDB_PASSWORD=adminpw  
  ports:  
    - 5984:5984  
  networks:  
    - test
```

Figura 4.5: Creación del contenedor Docker para el servicio CouchDB asociado a la organización 1

Así mismo es importante resaltar que la base de datos implementada en *Hyperledger Fabric* se considera de solo lectura, esto quiere decir que únicamente se podrán realizar consultas a los datos sin tener la posibilidad de alterar la integridad de estos. En otras

palabras, es una réplica unidireccional¹⁰ del *ledger* que se sincroniza en tiempo real y tiene la capacidad de albergar los datos del libro mayor.

4.2.4 Creación del canal de comunicación

Una vez generados los *peers* y las autoridades certificadoras es necesario establecer el canal de comunicación entre ellos para poder replicar las transacciones realizadas en todos los nodos de la red por lo que, en la creación de un canal, es necesario indicar el número de *peers* participantes en el consorcio, así como establecer las variables de entorno necesarias para su correcta comunicación.

Cabe mencionar que una de las características que proporciona *Fabric* al sistema es su escalabilidad, lo que permite integrar de forma sencilla a más organizaciones a la red aún cuando esta se encuentra ya funcionando, así mismo cada una de las organizaciones tienen la facultad de abandonar la red en cualquier momento.

Como se puede apreciar en la Figura 4.2, la aplicación *blockchain* propuesta únicamente funciona con un canal de comunicación común a todas las organizaciones y es mediante el cual se generarán transacciones y se replicarán los registros agregados al *ledger*. Sin embargo, si en algún momento fuera necesario el agregar un nuevo canal para la comunicación parcial o total de la red, este pudiera ser implementado en cualquier momento.

4.2.5 Publicación del *chain code*

El *chain code*, también conocido como *smart contract* o contrato inteligente, se encarga de albergar la lógica de negocio de nuestro sistema. Este debe ser acreditado y albergado por cada una de las organizaciones perteneciente a la red *blockchain*, es por esto que a continuación, se detalla el procedimiento a seguir para la propagación del contrato.

Para realizar la codificación del *chain code*, es necesario analizar la lógica de negocio, estableciendo los criterios necesarios que deberán seguir las organizaciones para la creación y acreditación de transacciones, así como el uso y accesibilidad a los datos almacenados en

¹⁰ La replica unidireccional hace referencia a la inserción de datos únicamente desde el *ledger* hacia la base de datos, no de forma contraria.

el ledger. A continuación, se presentan extractos de código que resultan relevantes para la correcta comprensión del flujo completo de la aplicación presentada.

Como se puede apreciar en la Figura 4.5, es necesario definir la estructura que tendrá cada una de las transacciones realizadas en el *ledger*, se puede asimilar este proceso al del modelado de una tabla en una base de datos, ya que se le indican los parámetros y el tipo de datos que deberán ser llenados para poder ingresar un registro al libro mayor.

```
type Car struct {
    ID                string `json:"id"`
    HologramaObtenido string `json:"hologramaObtenido"`
    Niv               string `json:"niv"`
    Comentarios      string `json:"comentarios"`
    Marca            string `json:"marca"`
    Modelo           string `json:"modelo"`
    Placas           string `json:"placas"`
    VerificentroId   string `json:"verificentroId"`
    TecnicoId        string `json:"tecnicoid"`
    OdometroId       string `json:"odometroid"`
    CCVValId         string `json:"ccvvalid"`
    ValidadorId      string `json:"validadorid"`
    LineaVerifica    string `json:"lineaverifica"`
    Status           string `json:"status"`
    TapaGasolina     string `json:"tapagasolina"`
    BayonetaAceite   string `json:"bayonetaaceite"`
    FiltroAire       string `json:"filtroaire"`
    TurboEscape      string `json:"tuboescape"`
    TaponRadiador    string `json:"taponradiador"`
    MangueraVacio    string `json:"mangueravacio"`
    Ruedas           string `json:"ruedas"`
    LucesTyD         string `json:"lucestyd"`
    CO               string `json:"co"`
    CO2              string `json:"co2"`
    O2               string `json:"o2"`
    NOxppm          string `json:"noxppm"`
    HCxppm          string `json:"hidrocarburo"`
    Lambda          string `json:"lambda"`
    CreateDate       string `json:"createdate"`
    UpdateDate       string `json:"updatedate"`
    AddedAt          uint64 `json:"addedAt"`
}
```

Figura 4.5: Definición de la estructura “Car” en el contrato inteligente

En la sección anexa del trabajo, se encuentra el código en lenguaje de programación Go que contiene la implementación completa del contrato inteligente utilizado, resaltando algunas funciones vitales para el sistema como lo es:

- “*CreateCar*”: Esta función se encarga de recabar los datos necesarios para la inserción de una nueva entrada al *ledger*. Creando las funciones *hash* necesarias para validar los datos ya albergados en el libro mayor, posteriormente realizar la inserción y distribuir la transacción hacia todos los demás nodos de la red.

Es de resaltar el uso de las funciones *marshalling* y *unmarshalling*¹¹ que son necesarias para el intercambio de información entre el cliente y el servidor.

- “*GetCarById*”: Mediante esta función es que se puede acceder de forma directa a los valores almacenados en el *ledger* sin embargo, como se ha mencionado en secciones anteriores, el uso de una base de datos de solo lectura, viene a reemplazar esta funcionalidad que *Hyperledger* le provee a los administradores de la red.
- “*GetHistoryForAsset*”: Esta función es la encargada de realizar una consulta directa al *ledger* mediante el atributo identificador de la estructura “*Car*” obteniendo como resultado, el histórico de las transacciones realizadas para el identificador en seleccionado, con esta función es que se puede dar seguimiento de manera puntual a transacciones específicas.

Una vez que el contrato inteligente ha sido codificado y analizado para su correcto funcionamiento, es necesario difundirlo xxa todas las organizaciones pertenecientes a la red, las cuales se encargarán de albergar y ejecutar los archivos necesarios para su acreditación. El procedimiento inicia comprimiendo el contrato inteligente con el fin de obtener un archivo tipo “tar”, el cual será enviado a cada uno de los nodos para su descompresión, análisis, ejecución y acreditación.

Cabe mencionar que el uso del mismo contrato inteligente en todas las organizaciones proporciona a la red una capa de seguridad y estabilidad al sistema que le permite generar confianza, tanto en la parte del cliente como en la de los operadores, ya que *Hyperledger* verifica que las transacciones realizadas hayan sido generadas mediante el contrato inteligente más reciente a la fecha actual. Así mismo resulta imposible que se generen

¹¹ Marshalling se refiere a la acción de almacenar el estado de un objeto junto con su código, mientras que serializar es solamente crear copias de objetos como flujos de bytes. El procedimiento inverso de la serialización es la deserialización o unmarshalling. [32]

modificaciones al contrato sin previa autorización, por lo que, si en algún momento es necesario el actualizar las reglas de negocio establecidas en el *smart contract*, este debe ser empaquetado, distribuido, ejecutado y autorizado nuevamente por todas las organizaciones.

4.2.6 Publicación del servicio *Hyperledger Explorer*

Hyperledger Explorer es un servicio que *Fabric* le provee al usuario mediante una interfaz gráfica de consulta global sobre los bloques de datos, hashes, contratos inteligentes y canales generados en la implementación de forma global, enfocada en facilitar el mantenimiento de esta a través de un servicio web.

El servicio de *Explorer*, al igual que cada instancia presentada en esta sección, debe ser alojado en un contenedor *docker* que le permita consumir los recursos establecidos en el archivo de configuración, por lo que una vez montada la red con todos los *peers* y canales de comunicación funcionando, es posible montar el servicio que se abrirá en el puerto 8080 (si la implementación es local).

4.2.7 Creación del servidor para la interfaz gráfica

Hasta este punto, únicamente se había configurado la parte relacionada con el *backend* de la aplicación, sin embargo, una vez que la red *blockchain* se encuentra funcionando, es necesario levantar un servidor capaz de administrar la interacción con el usuario final, para poder manipular, insertar y consultar las transacciones realizadas.

Para esto se utilizaron los siguientes *frameworks*¹² de desarrollo *Vue JS*, *Bootstrap*, *Quasar* y *Node js*. Estos se usaron para realizar el maquetado de la interfaz del sistema de verificación vehicular. Así mismo algunas dependencias propias de *Node JS* como *Axios* se usaron para realizar peticiones HTTP a los servicios previamente montados de *Hyperledger Fabric*. Cabe mencionar que el puerto lógico generado para recibir peticiones a este servicio es el 8081 en un entorno local. En los siguientes capítulos se profundizará de manera detallada en el funcionamiento de la implementación presentada en este capítulo, así

¹² Tipo de software que provee funcionalidades genéricas que ayudan a facilitar la elaboración de alguna tarea, optimizando el tiempo y mejorando la calidad de las herramientas generadas.

mismo se harán pruebas ejemplificando los alcances y seguridad que la herramienta provee al usuario.

Capítulo 5. Resultados

En la primera parte de este capítulo se profundizará sobre el uso de la herramienta desarrollada, con un ejemplo en donde se llevará a cabo el registro de un automóvil en algún centro de verificación vehicular, posteriormente se simularán las pruebas de medición de contaminantes mencionadas en capítulos anteriores para poder ser almacenadas en el libro mayor. Así mismo se validará y asignará un holograma con base en los criterios establecidos en el capítulo 4 para posteriormente replicar la información de esta transacción a todas las organizaciones restantes.

En la segunda parte del capítulo se abordará el uso de la herramienta *Hyperledger Explorer*, la cual ayudará a comprender de forma gráfica el flujo completo por el cual debe someterse una transacción para poder ser creada, validada, almacenada y replicada en la red *blockchain*, resaltando en todo momento los beneficios relacionados seguridad y consistencia de los datos, que la herramienta le provee al sistema.

5.1 Análisis del funcionamiento

Una vez que las entidades involucradas en la implementación de la infraestructura se encuentren configuradas y funcionando, se puede iniciar con la creación de transacciones en la red, siguiendo la lógica del diagrama de flujo mostrado en la Figura 5.1, iniciando por el registro del automóvil, almacenando valores propios de la unidad como el número de placa, NIV y modelo, para así poder consultar si el vehículo cuenta con alguna infracción pendiente de pago en el sistema de verificación de multas.

Posteriormente se asignará un técnico y línea verificadora en donde se realizará la prueba de velocidad y aceleración para poder recabar los datos, se subirá la información al sistema de *blockchain* en donde automáticamente y de forma aleatoria se otorgará la

transacción a algún CVV que será la encargada de analizar los resultados de las pruebas antes mencionadas para poder otorgar o no un holograma al vehículo.

En el momento en el que el CVV remoto hace la actualización de la información en el *ledger*, el CVV origen imprime el comprobante y coloca el engomado en la unidad del ciudadano. Es aquí donde el flujo de verificación vehicular termina.

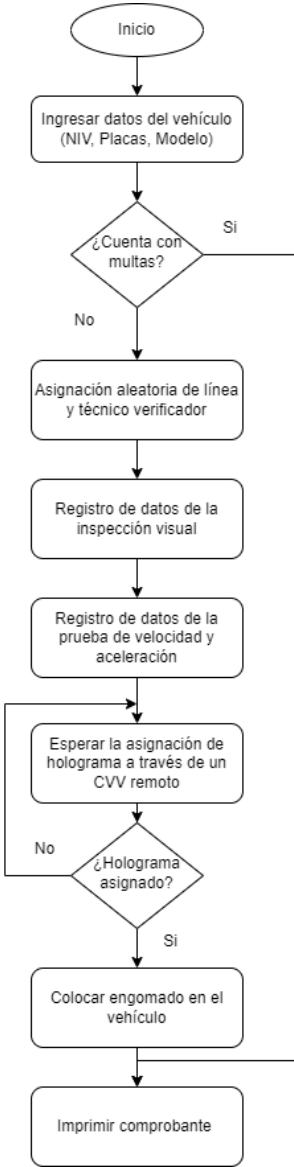


Figura 5.1: Diagrama de flujo a seguir para la verificación de un vehículo particular

A continuación, se presenta un ejemplo práctico utilizando un sistema web desarrollado con de las herramientas y *frameworks* mencionadas en capítulos anteriores, integrando funcionalidades propias de *blockchain* a través de *Hyperledger Fabric*

El primer paso a realizar en el sistema de verificación vehicular es la autenticación de usuarios, actualmente existen 10 usuarios registrados en el sistema, los cuales son los representantes de cada una de las 10 organizaciones que forman parte del *blockchain*. Como se puede apreciar en la Figura 5.2 se realiza una autenticación mediante usuario y contraseña. Cabe mencionar que, para fines prácticos, el nombre de usuario y contraseña es el mismo en todos los casos, iniciando por el usuario CVV_1, CVV_2 de manera ascendente hasta llegar al usuario CVV_10.

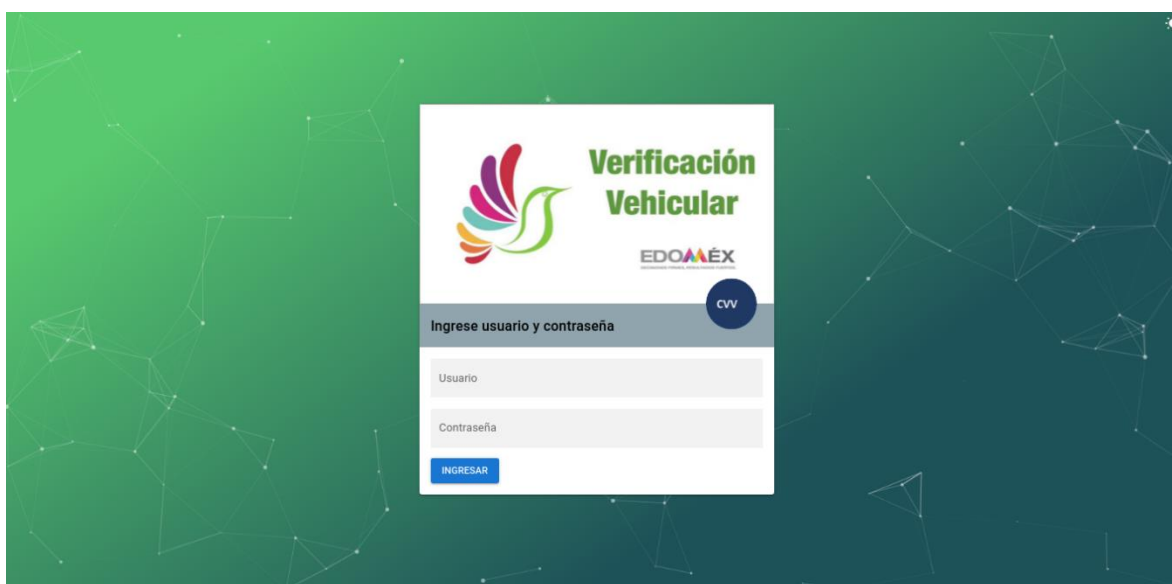


Figura 5.2: Página de autenticación mediante usuario y contraseña

Una vez autenticado en el sistema, se generará un *token* de forma a dinámica que le permitirá al usuario el poder interactuar con el *ledger* en tareas como generar, validar y consultar transacciones. Es de resaltar que este *token* únicamente servirá para la sesión actual. Por cuestiones de seguridad en el sistema, cada vez que un usuario se autentica, se le proporcionará un token nuevo.

Como se puede apreciar en la Figura 5.3, existe un menú lateral en la parte izquierda de la pantalla que le permitirá al usuario el poder navegar sobre las funcionalidades que se

irán presentando a lo largo de este capítulo. En un inicio, el sistema posiciona al usuario en la ventana de “Registrar entrada” en donde es capaz de ingresar una nueva transacción.

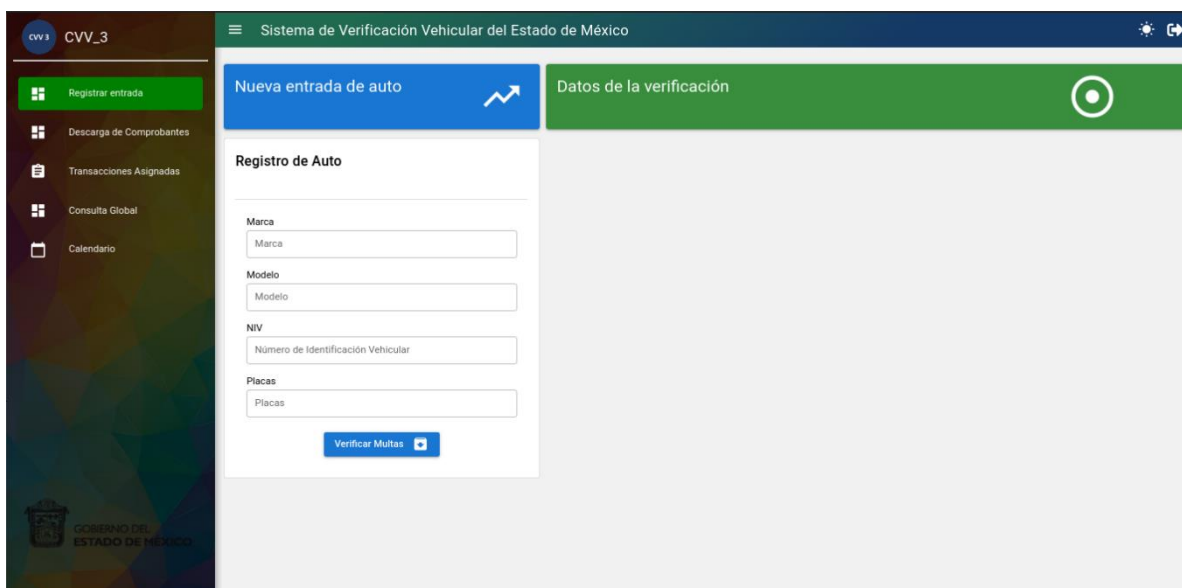


Figura 5.3: Sección “Registrar entrada”

Para ejemplificar, si un ciudadano decide acudir a un CVV para la inspección semestral de su automóvil particular. El personal capacitado en cuestión (técnico capturista), le recibe a la entrada y le solicita los documentos necesarios (tarjeta de circulación, última papeleta de verificación y una identificación oficial) para comenzar con el registro del procedimiento. Es en ese momento cuando el sistema propuesto empieza a registrar los datos necesarios para almacenar la información particular del cliente y su automóvil.

Con ayuda de los datos ingresados, se realizará una consulta al sistema de multas¹³ en el Estado de México para validar si el vehículo cuenta con sanciones pendientes de pago. De ser así se le solicitará al ciudadano pagar el monto total y posteriormente volver a asistir a un CVV, como se puede apreciar en la Figura 5.4. Automáticamente el sistema realizará la inserción de los datos en el *ledger* informando que el auto en cuestión fue rechazado por multas, en ese momento el estado de la transacción cambiará a “Rechazado por multas” y se

¹³ Para fines prácticos, la consulta al sistema de multas es un proceso simulado que arroja datos no reales. Sin embargo, en un ambiente de producción se utilizaría una API que pudiera acceder a la información de esta plataforma.

podrá descargar un comprobante de rechazo mismo que se le proveerá al ciudadano.

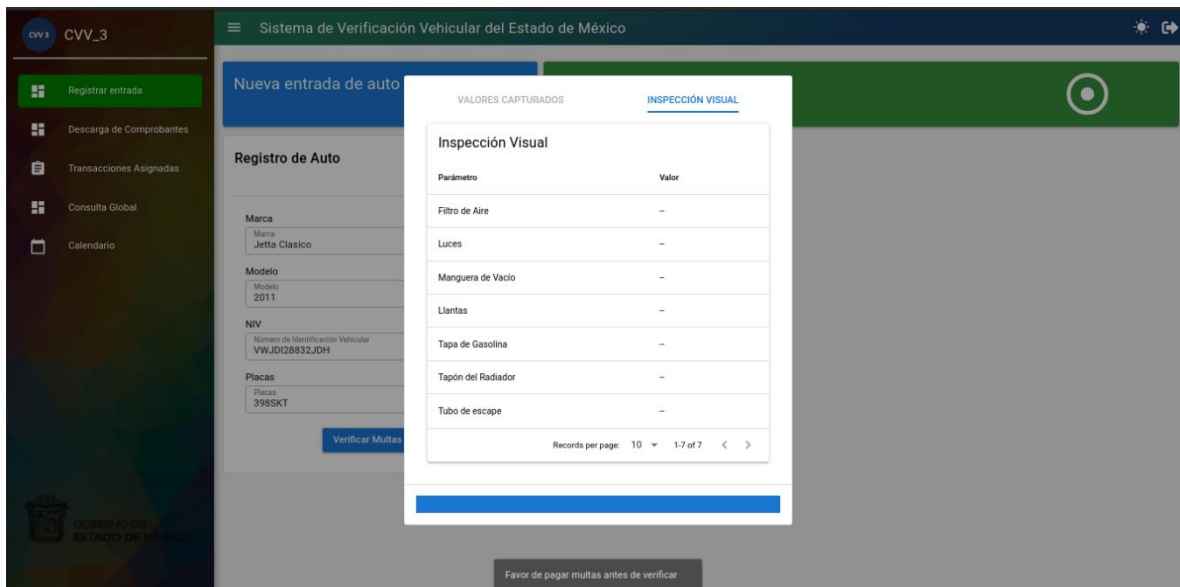


Figura 5.4: Automóvil rechazado por multas

Una vez que se valida que la unidad no cuenta con adeudos de multa pendientes, el sistema establece de forma aleatoria el número de línea o pista en la cual deberá ser inspeccionado el automóvil, el identificador del odómetro y el nombre del técnico que realizará la pruebas, como se puede apreciar en la Figura 5.5.

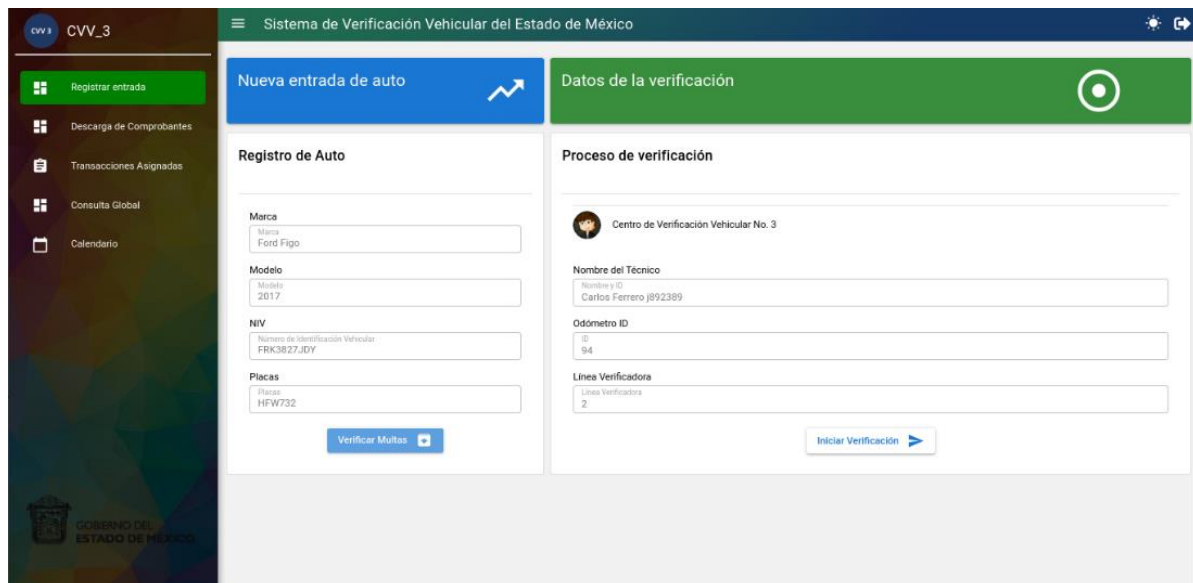


Figura 5.5: Asignación aleatoria de línea, odómetro y técnico verificador

El usuario deberá permanecer en la sala de espera mientras se realiza el procedimiento de verificación. Iniciando por la inspección visual en la que se validará el cumplimiento y estado de deterioro de los siguientes puntos:

- Tapón de gasolina
- Bayoneta de aceite
- Filtro de aire
- Tubo de escape
- Tapón del radiador
- Mangueras de vacío
- Abombamiento de ruedas
- Luces de delanteras y traseras

La Figura 5.6 muestra la forma en que estos valores son almacenados al sistema, ya que son los únicos que podrán ser ingresados de forma manual por el técnico verificador. Al tratarse de una prueba visual, se necesita el criterio de validación por parte del personal. Cabe mencionar que cada uno de estos rubros podrán obtener la etiqueta de “Cumple” o “No cumple” dependiendo los resultados de la prueba.

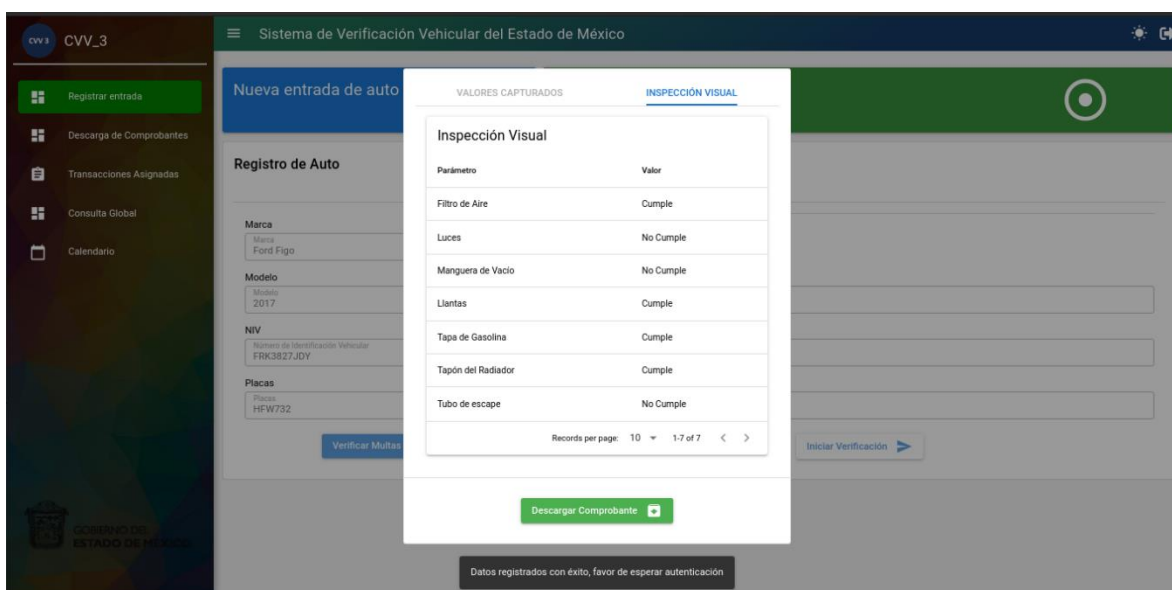


Figura 5.6: Registro de valores para la inspección visual por el técnico

En el siguiente paso, el sistema deduce automáticamente a partir del modelo del automóvil, si es candidato o no a someterse a un escaneo mediante SDB ya que como se mencionó en secciones previas, los vehículos modelo 2000 o anteriores no cuentan con el sistema de diagnóstico a bordo. De ser así, el campo relacionado a esta prueba se completará de forma automática con la leyenda “N/A”. Sin embargo, si la unidad cuenta con el puerto para ser escaneado, el técnico encargado deberá conectar el enchufe SDB al automóvil para su diagnóstico. Una vez realizado los pasos anteriores, se someterá al vehículo a las pruebas de humo, carga y velocidad. El técnico verificador deberá colocar la pipeta en el escape, posteriormente acelerar el vehículo a 24 km/h por un tiempo de 30 segundos, para identificar si la unidad arroja humo negro o azul.

Para la prueba de carga y velocidad, se deberá acelerar el vehículo a 24 km/h durante 60 segundos para que la computadora logre registrar los niveles de monóxido de carbono (CO), dióxido de carbono (CO2), óxidos de nitrógeno (NOxppm), hidrocarburos (HC hppm) y el factor Lambda (Figura 6.7). Los cuáles serán registrados en el sistema *blockchain* para su posterior evaluación. A partir de este momento, el estatus de la transacción tomará el valor de “Por validar” y los únicos atributos que se encontrarán vacíos son: “CVV Validador Id” y “validador Id”.

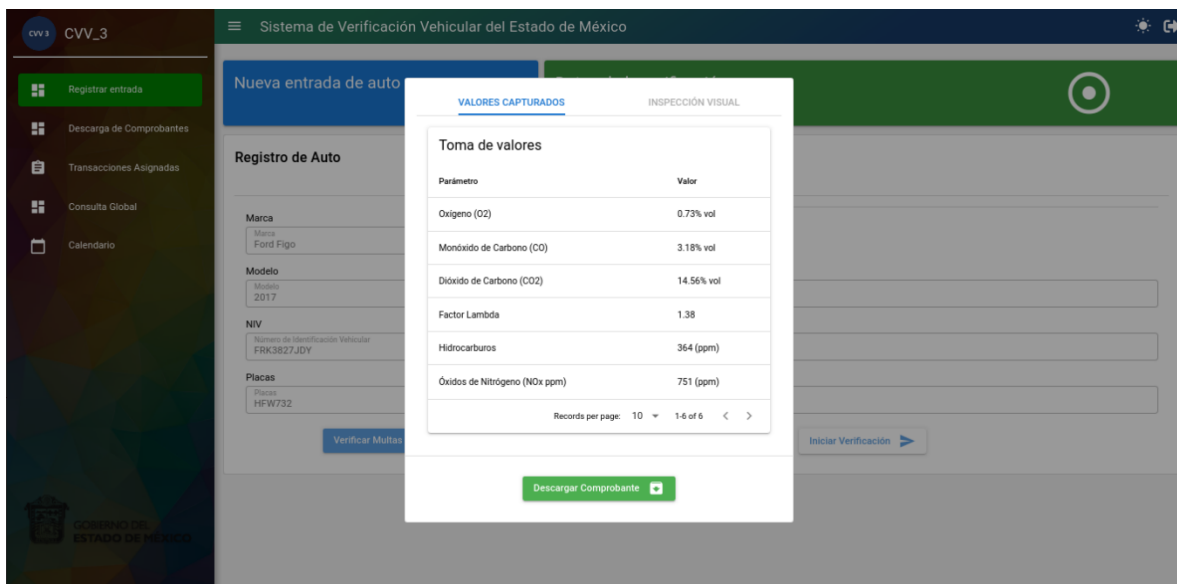


Figura 5.7: Registro de valores para la prueba de carga y velocidad

Como se puede apreciar en la Figura 5.8, a través del *Hyperledger Explorer* podemos ver que los datos recabados en las transacciones anteriores ahora forman parte del *ledger*, generando un nuevo registro que ha sido concatenado al libro mayor mediante una función hash que ayuda mantener la integridad de los registros.

El siguiente paso a seguir consiste en colocar el vehículo en la zona de espera, mientras que internamente la implementación a través del protocolo de consenso RAFT, busca a algún CVV candidato de forma aleatoria para asignarle la validación de los datos recabados. A diferencia del protocolo actual, en donde la asignación de holograma se hacía de forma local (en el mismo CVV) ahora con ayuda de *blockchain*, la tarea de analizar y asignar un estampado será delegada a uno de los 137 verificentros restantes (sin contar el CVV actual) esto con el objetivo de evitar colusiones entre los usuarios y el personal encargado de realizar las pruebas de verificación.

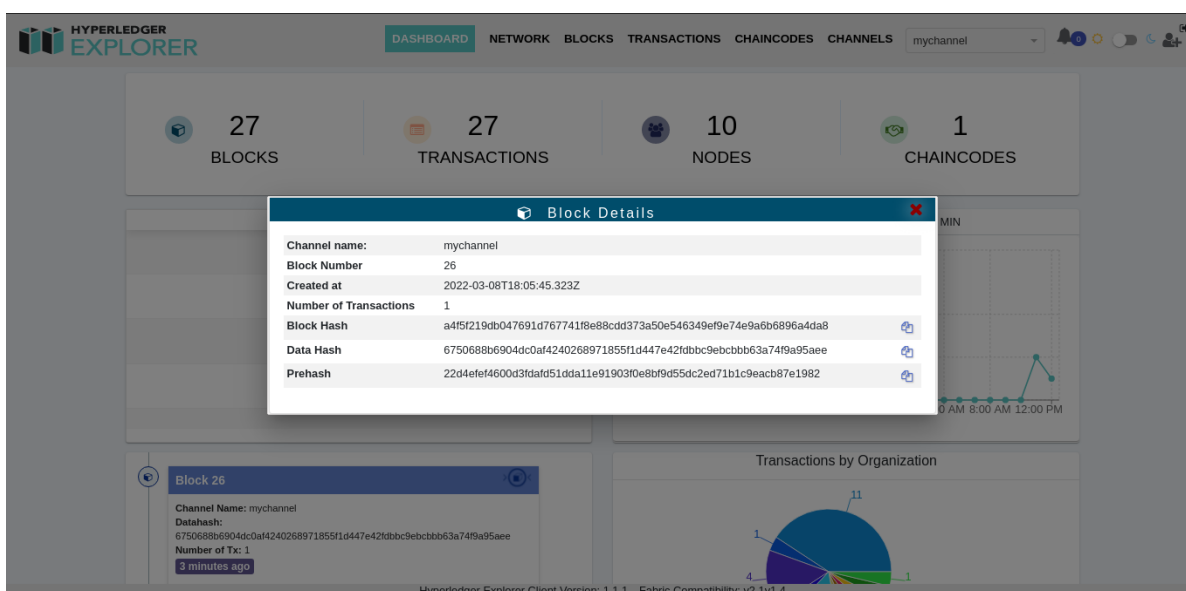


Figura 5.8: Visualización del bloque insertado a través de la interfaz de *Hyperledger Explorer*

Aunado a esto, existe una capa adicional de anonimato entre los verificentros que evitará en la medida de lo posible, la corrupción en el proceso, ya que el remitente de la transacción no podrá saber en ningún momento cual CVV en específico será el encargado de validar su transacción generada. Así mismo, el receptor tampoco podrá saber cuál centro de verificación vehicular envió la transacción por validar, ya que ninguna de estas entidades tienen acceso a consultar estos atributos hasta que el estatus de la transacción aparezca como “Finalizado” y se le haya otorgado un holograma al vehículo.

En este caso en particular, la transacción registrada fue asignada al centro de verificación vehicular número 1, el cual ya se ha identificado previamente en otra instancia del sistema (dentro de su propio navegador web) y se ha posicionado en la sección de "Transacciones Asignadas". Esto se puede apreciar en la Figura 5.9, en donde se muestra el resumen de todas las transacciones validadas y pendientes. Así el usuario asignado, deberá evaluar los datos presentados en la Figura 5.10 conforme a los parámetros establecidos de las tablas 4.2, 4.3 y 4.4 para así poder asignar el estampado correspondiente y de ser necesario, capturar algún comentario como se puede muestra en la Figura 5.11.

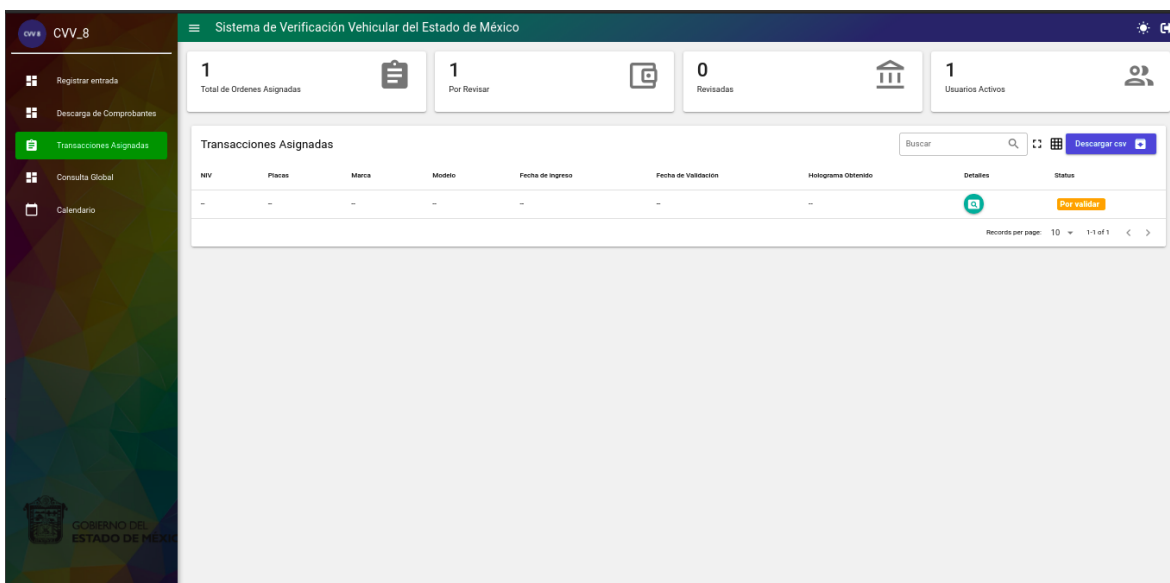


Figura 5.9: Sección de “Transacciones Asignadas”

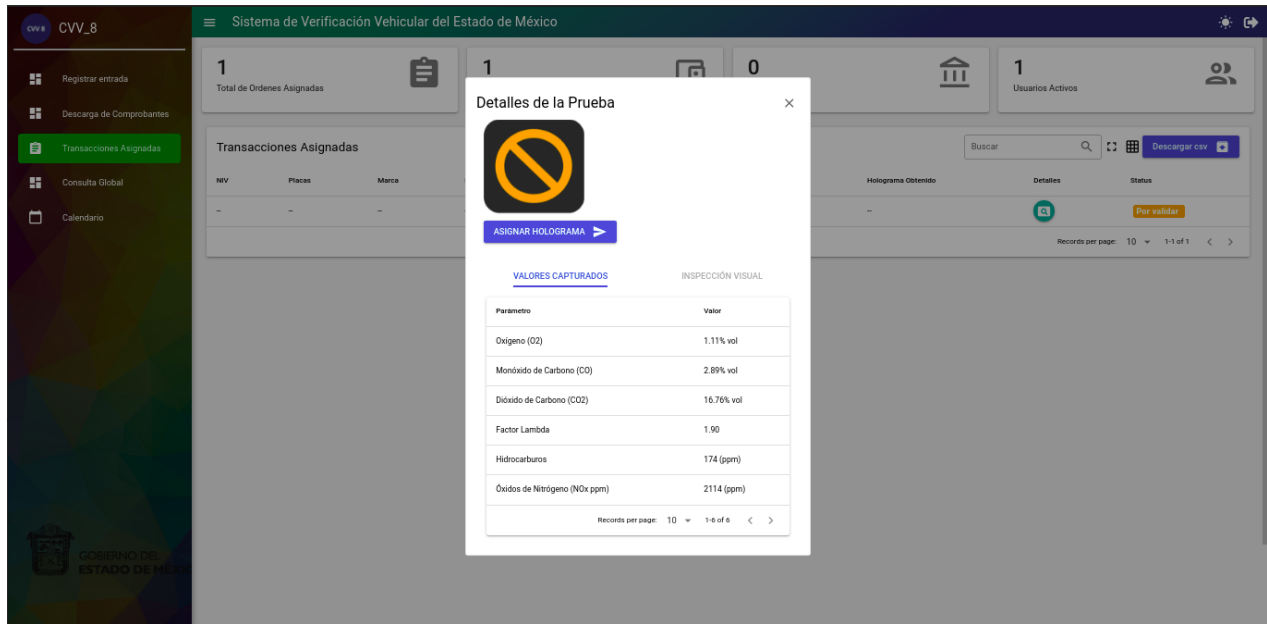


Figura 5.10: Resumen de los parámetros obtenidos en la prueba de verificación, en espera de ser evaluados por el validador.

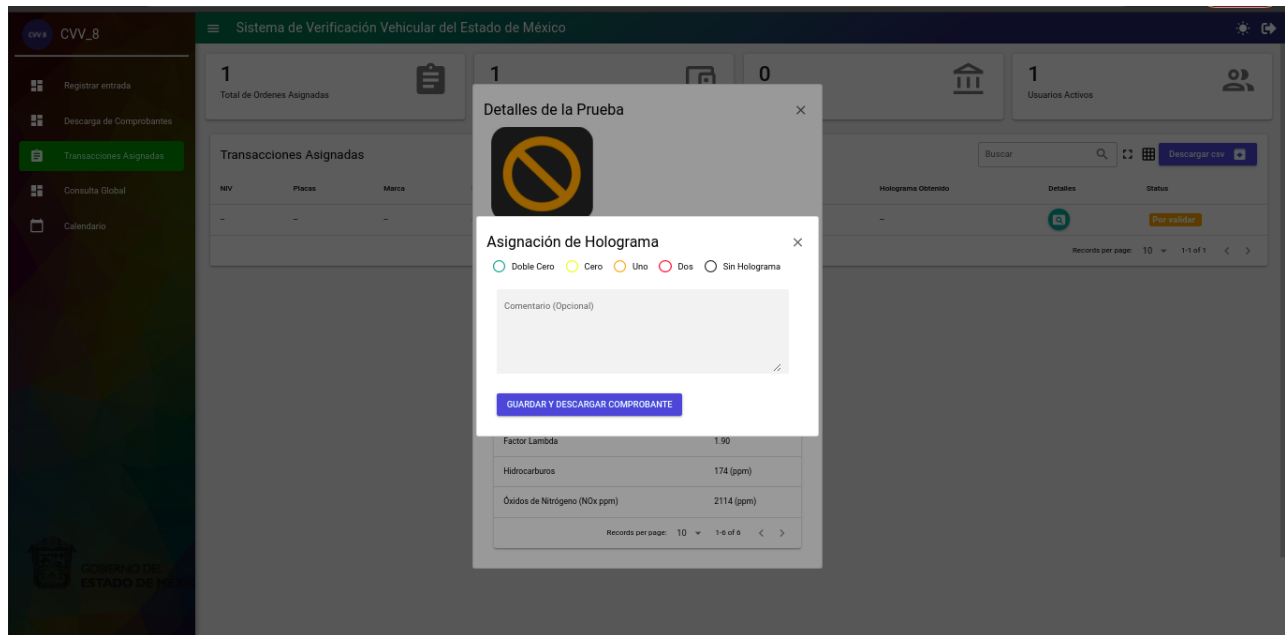


Figura 5.11: Menú de asignación de holograma

Una vez analizada esta transacción y establecido el holograma correspondiente desde un CVV remoto, internamente se actualiza el registro en el *ledger* para su correcta propagación a todas las organizaciones, como se puede apreciar en la Figura 5.12. Resulta imposible modificar un bloque previamente almacenado en el *ledger*, por lo que es necesario

crear una nueva transacción con el mismo identificador, parámetros y únicamente añadir los campos de “estatus”, “CCVValId” y “ValidadorId”.

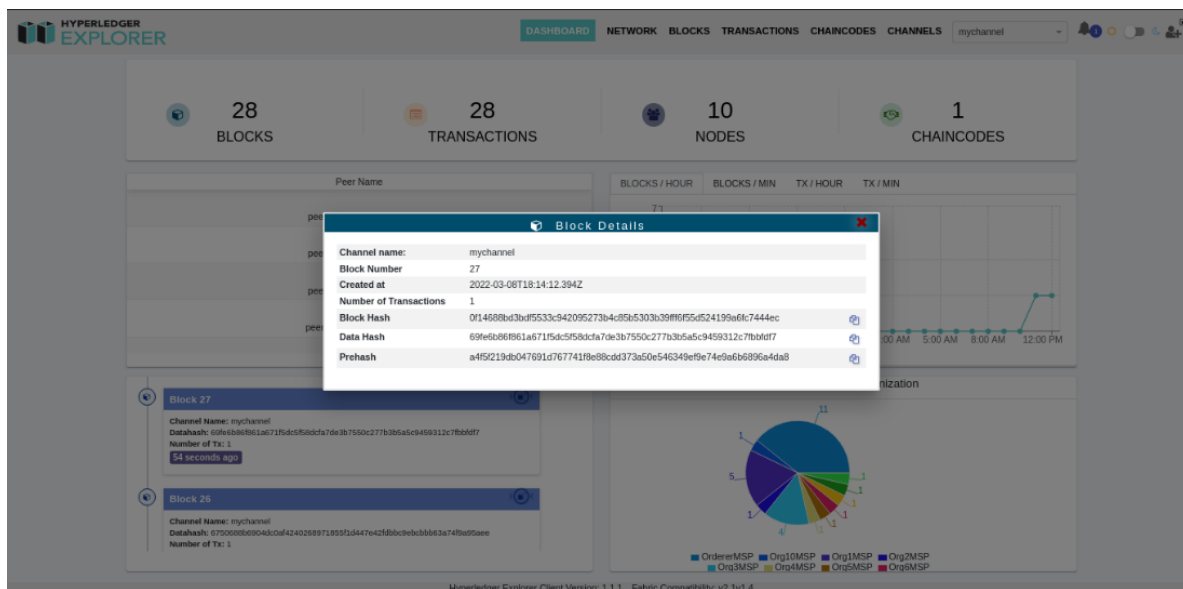


Figura 5.12: Información del nuevo bloque generado a partir de la asignación de holograma (Nótese que el hash es diferente al de la Figura 6.8)

El procedimiento de propagación, validación y actualización de la información toma alrededor de 5 minutos, mismos que el ciudadano deberá aguardar en la sala de espera. Una vez que el procedimiento de verificación haya llegado a su fin el técnico capturista recibirá la respuesta del tipo de holograma otorgado como se muestra en la sección de "Consulta Global" en la Figura 5.13. Este holograma deberá ser colocado en el medallón del vehículo así mismo se deberá imprimir el resultado de la prueba de verificación para proveer al usuario de un comprobante oficial como el de la Figura 5.14. Es en este punto donde se le da acceso al ciudadano de poder recoger su unidad portando el nuevo holograma obtenido.

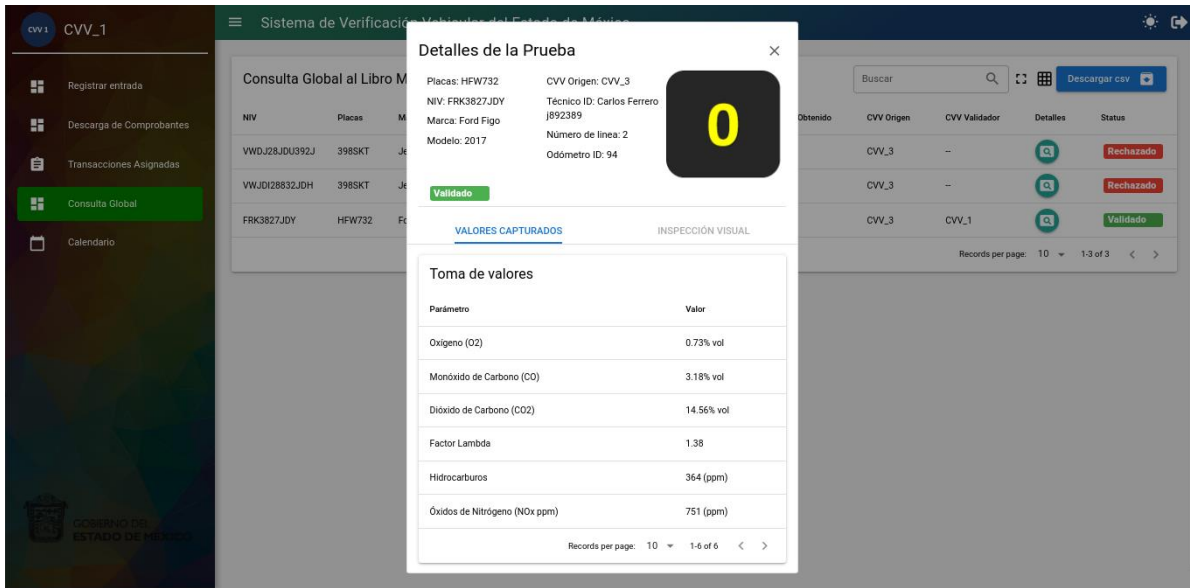


Figura 5.13: Sección de consulta global de la transacción ya validada y con holograma asignado



Figura 5.14: Comprobante oficial obtenido una vez que la transacción es validada

Las secciones de "Descarga de Comprobante" y "Calendario" son apartados que ayudarán a los operadores de cada uno de los centros de verificación vehicular a mejorar los procedimientos y facilitar las auditorías a estas dependencias, ya que como su nombre lo indica en la sección de "Descarga de Comprobante", es posible hacer una búsqueda en el libro mayor a través del NIV para posteriormente obtener el documento oficial que avale el resultado de las pruebas aplicadas a los automóviles. Esto se puede apreciar en la Figura 5.15.

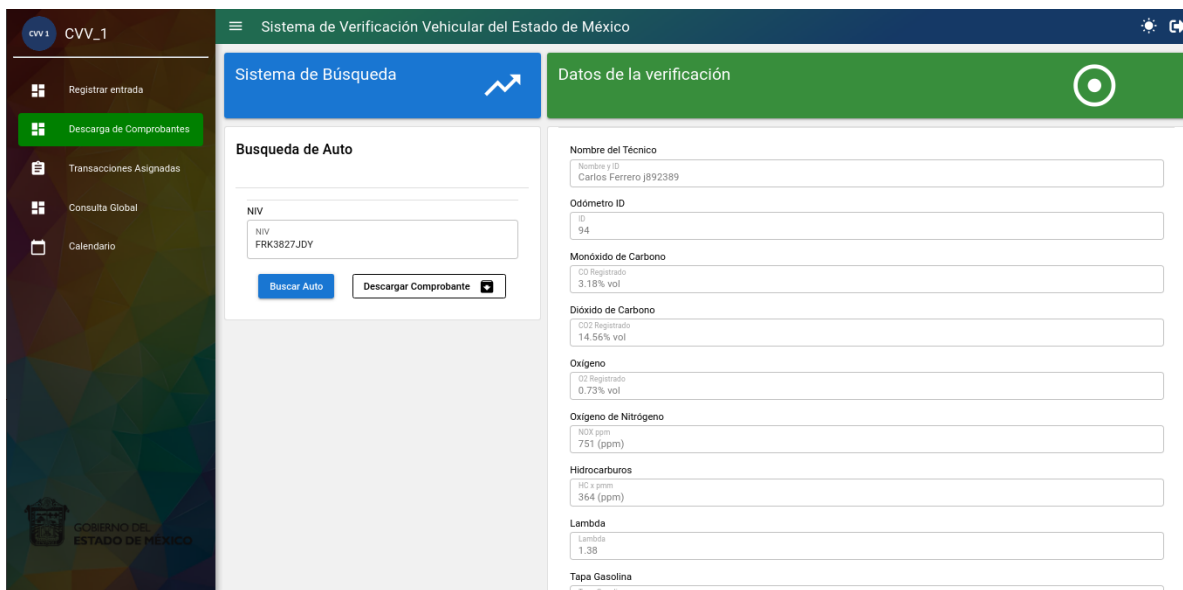


Figura 5.15: Sección de “Descarga de Comprobante”

Por otro lado, la sección de "Calendario" es únicamente para informar a los operadores de los CVV las fechas relevantes en las que se realizarán auditorías, mantenimientos, juntas y visitas de técnicos como se puede apreciar en la Figura 5.16.

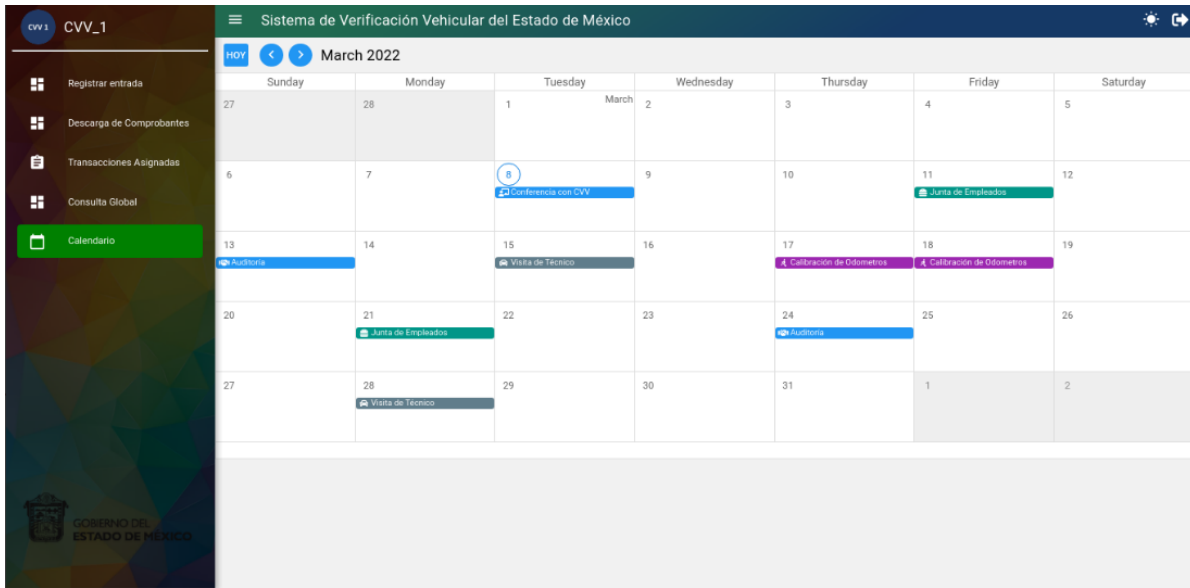


Figura 5.16: Sección de “Calendario”

5.2 Hyperledger Explorer

Como se mencionó en secciones anteriores, el servicio de *Hyperledger Explorer*, es una de las herramientas más utilizadas para ilustrar de forma gráfica las tareas que se generan integrante en el *ledger* al momento de actualizar y añadir transacciones. Es de resaltar que este servicio es únicamente es de consulta, por lo que resulta imposible el modificar algún bloque de datos, canal u organización a través de él. Sin embargo, en esta sección se ejemplificará de forma gráfica el cómo es que se generan los bloques internamente en la red *blockchain*, el uso de las funciones hash y algunos otros datos relevantes que la herramienta proporciona al usuario.

El servicio de Hyperlesger Explorer es una aplicación local que se encuentra alojada en el puerto 8080, y requiere de una previa autenticación para poder acceder a ella. Como se muestra en la Figura 5.17, se deberá ingresar el nombre de la red a la cual se va a conectar (“*First-network*” en este caso), así como el usuario y la contraseña previamente establecidas en los archivos de configuración de *Hyperledger Fabric*.

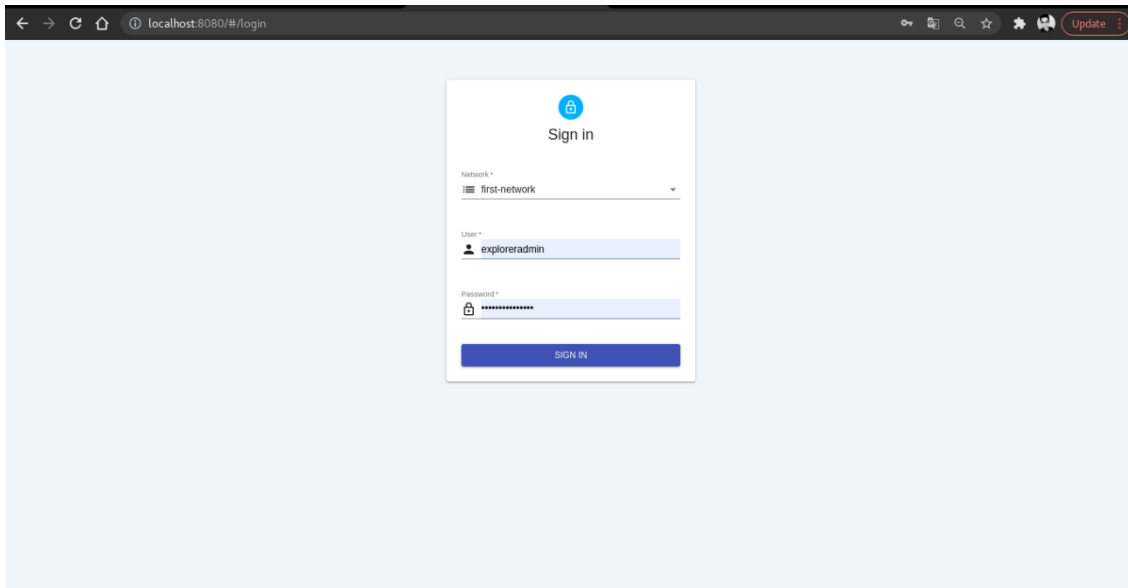


Figura 5.17: Interfaz de autenticación en Hyperledger Explorer

Una vez autenticado, es posible navegar dentro del sistema mediante el menú superior que indica el posicionamiento actual del usuario, en particular la figura 5.18 muestra la sección del *dashboard*, en donde se muestra un resumen del número total de bloques, transacciones, nodos y contratos inteligentes utilizados hasta el momento, así como dos diferentes tipos de gráficas las cuales contienen información relevante para el administrador. En la gráfica de líneas, se muestra el número de bloques generados por hora o minuto, y en la gráfica de pastel se indica el nombre de las organizaciones y el número de transacciones que han generado cada una de estas entidades. Así mismo en la parte central del *dashbord*, se muestra nombre de los *peers* generados hasta el momento. En este caso en particular únicamente se cuenta con 10 *peers* que son los representantes de cada centro de verificación vehicular.

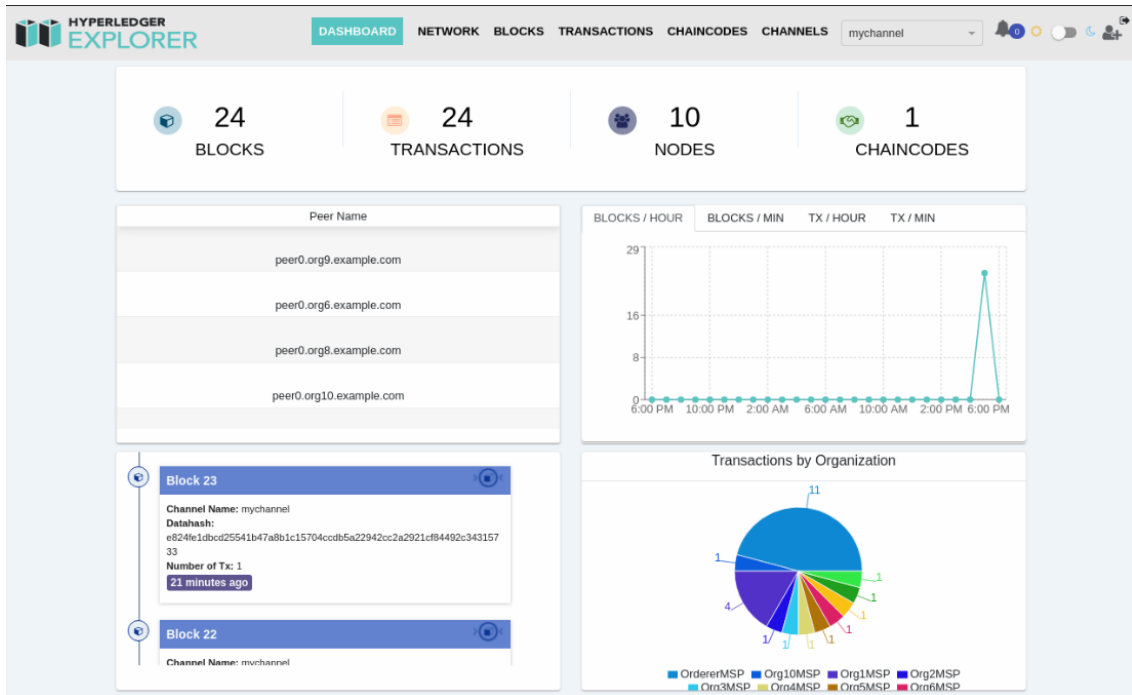


Figura 5.18: Sección “Dashboard” en *Hyperledger Explorer*

En la sección de *Network* (Figura 5.19) se puede apreciar una tabla con el número total de *peers* presentes en la implementación, así como sus respectivos servicios de ordenamiento que permiten realizar inserciones a la red.

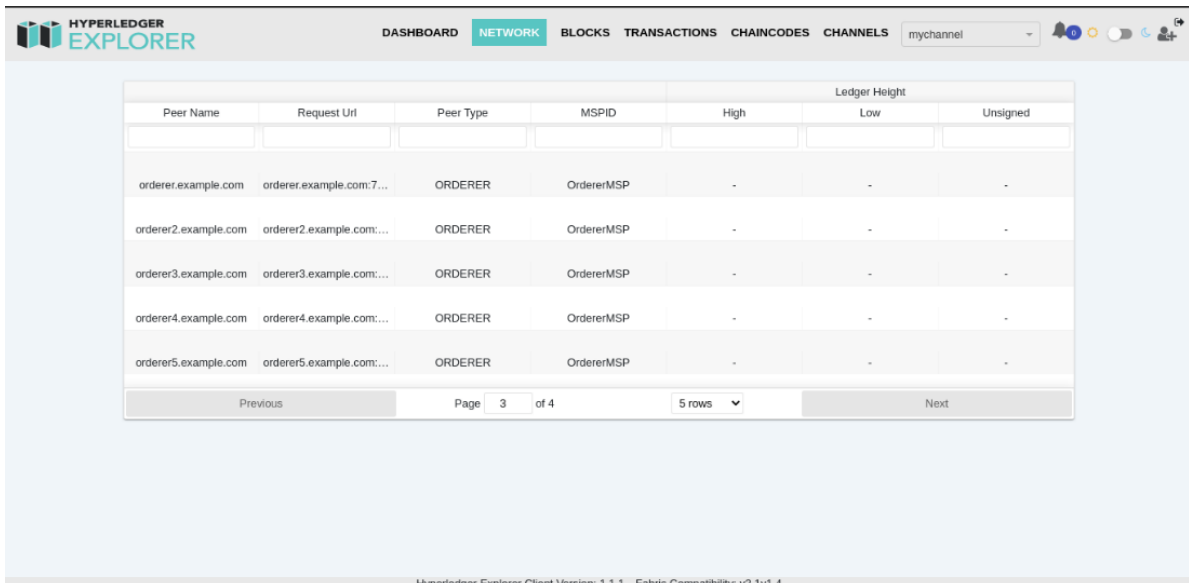


Figura 5.19: Sección “Network” en *Hyperledger Explorer*

Una de las funcionalidades más relevantes que provee *Hyperledger Explorer* se encuentra en la sección de *Blocks*, ya que como se puede apreciar en la Figura 5.20, la herramienta muestra información de cada uno de los bloques que forman parte del *ledger*, como lo son: el número de bloque generado, el nombre del canal, el resultado a la función hash generado, la función hash del bloque anterior y el tamaño en kilobytes del bloque. Así mismo, es posible realizar búsquedas mediante el filtro que la herramienta le provee al usuario.

Block Number	Channel Name	Number of Tx	Data Hash	Block Hash	Previous Hash	Transactions	Size(KB)
1	mychannel	1	021971 ...	62e5d6 ...	5bcec6 ...		28
4	mychannel	1	047a93 ...	0c3138 ...	fat4e4 ...		28
11	mychannel	1	0c7bb3 ...	6b6fb0 ...	8e4b40 ...	2e08df ...	5
19	mychannel	1	0f5c98 ...	2c1265 ...	33776c ...	e107ac ...	5
14	mychannel	1	128551 ...	4ba394 ...	af2ccf ...	17facb ...	5
18	mychannel	1	1516fc ...	33776c ...	3f340e ...	d23975 ...	5
0	mychannel	1	20b688 ...	5bcec6 ...			27

Hyperledger Explorer Client Version: 1.1.1 Fabric Compatibility: v2.1v1.4

Figura 5.20: Sección “*Blocks*” en *Hyperledger Explorer*

La Figura 5.21 muestra información detallada sobre el último bloque realizada por la organización uno, resaltando algunos datos como: el número de transacción generado, la política de validación presentada, la organización que insertó el registro y la hora exacta de la inserción.

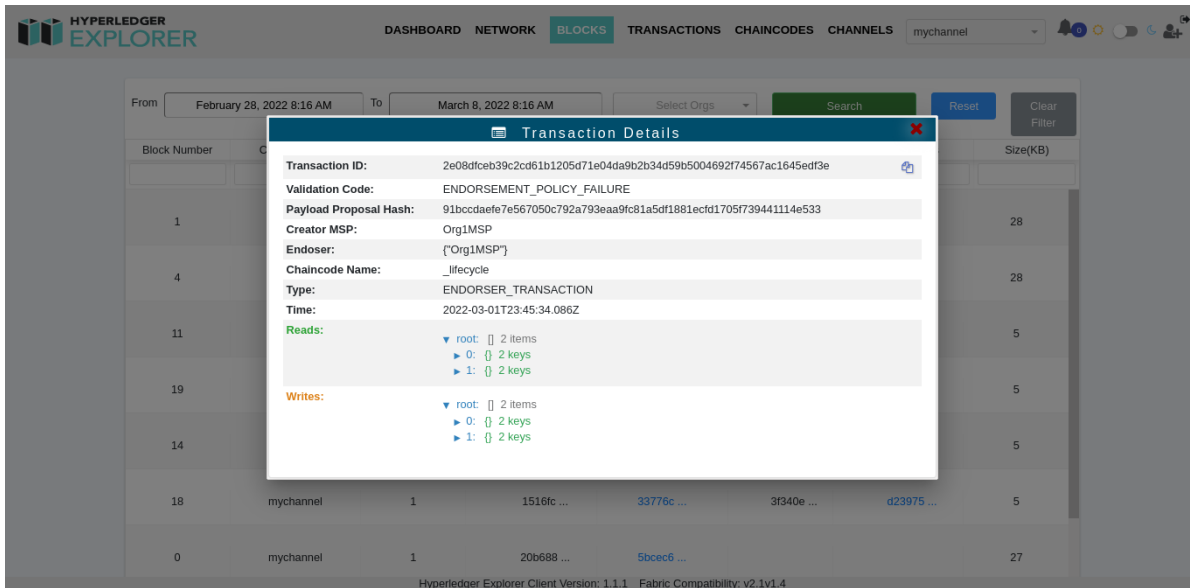


Figura 5.21: Detalles del último bloque agregado al libro mayor visto desde *Hyperledger Explorer*

La sección de *Transactions* es muy similar a la de *Blocks* ya que la información es presentada de la misma forma sin embargo algunas columnas contienen información propia de la transacción como lo es el nombre de la organización que realiza la inserción, el identificador de la transacción, el nombre del contrato inteligente utilizado y la fecha exacta en la que se creó. Recordando que un conjunto de transacciones puede formar parte de un bloque, la implementación presentada, almacena una transacción dentro de un bloque que a su vez es insertado en el *ledger* de mediante el servicio de ordenamiento propio de cada organización.

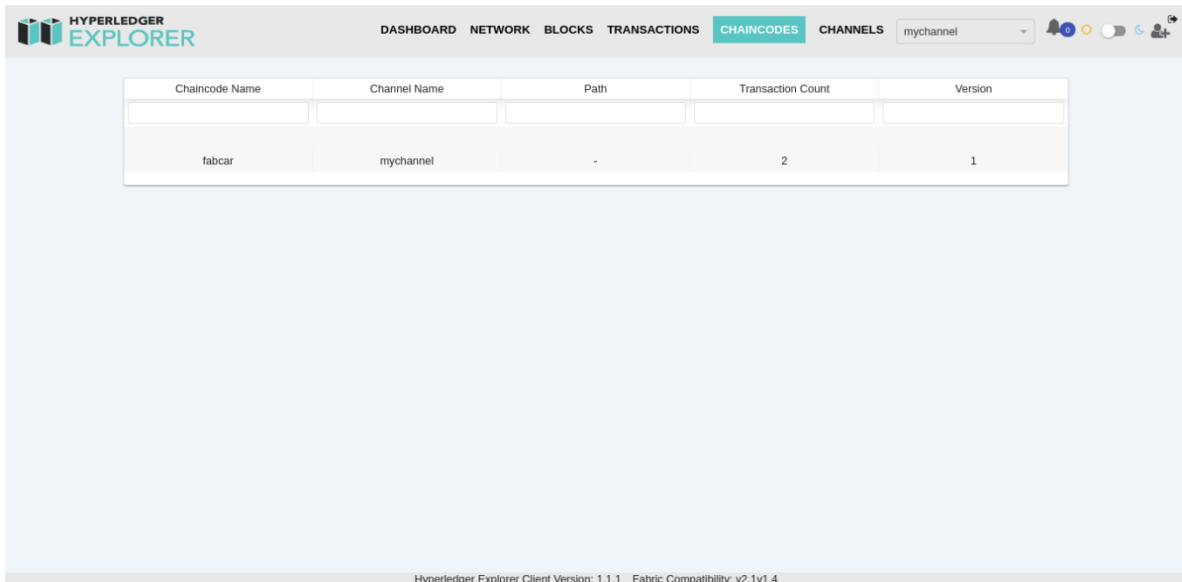
Creator	Channel Name	Tx Id	Type	Chaincode	Timestamp
OrdererMSP	mychannel		CONFIG		2022-03-01T23:45:14.000Z
Org1MSP	mychannel	5a96ct...	ENDORSER_TRANSACTION	fabcar	2022-03-01T23:46:17.745Z
Org1MSP	mychannel	9d3802...	ENDORSER_TRANSACTION	fabcar	2022-03-01T23:46:10.024Z
Org1MSP	mychannel	bdf4b1...	ENDORSER_TRANSACTION	_lifecycle	2022-03-01T23:46:06.022Z
Org10MSP	mychannel	4ac69f...	ENDORSER_TRANSACTION	_lifecycle	2022-03-01T23:46:03.352Z
Org9MSP	mychannel	e107ac...	ENDORSER_TRANSACTION	_lifecycle	2022-03-01T23:46:00.124Z
Org8MSP	mychannel	d23975...	ENDORSER_TRANSACTION	_lifecycle	2022-03-01T23:45:56.922Z
Org7MSP	mychannel	32649d...	ENDORSER_TRANSACTION	_lifecycle	2022-03-01T23:45:53.590Z

Figura 5.22: Sección “Transactions” en *Hyperledger Explorer*

Al momento de desplegar la información de una transacción como en la Figura 5.23, es posible visualizar los datos relevantes de la operación como lo son: el identificador, nombre de la política de validación, la organización que creó la transacción, el nombre del contrato inteligente utilizado y la fecha de creación. Mismos que ayudan a los administradores a mantener la red en operación y entender de forma sencilla el cómo es que se crean y albergan los datos en el libro mayor.

Figura 5.23: Detalles de la última transacción agregada al libro mayor visto desde *Hyperledger Explorer*

Por último, en las Figuras 5.24 y 5.25 muestran el nombre del contrato inteligente y el canal de comunicación utilizado.

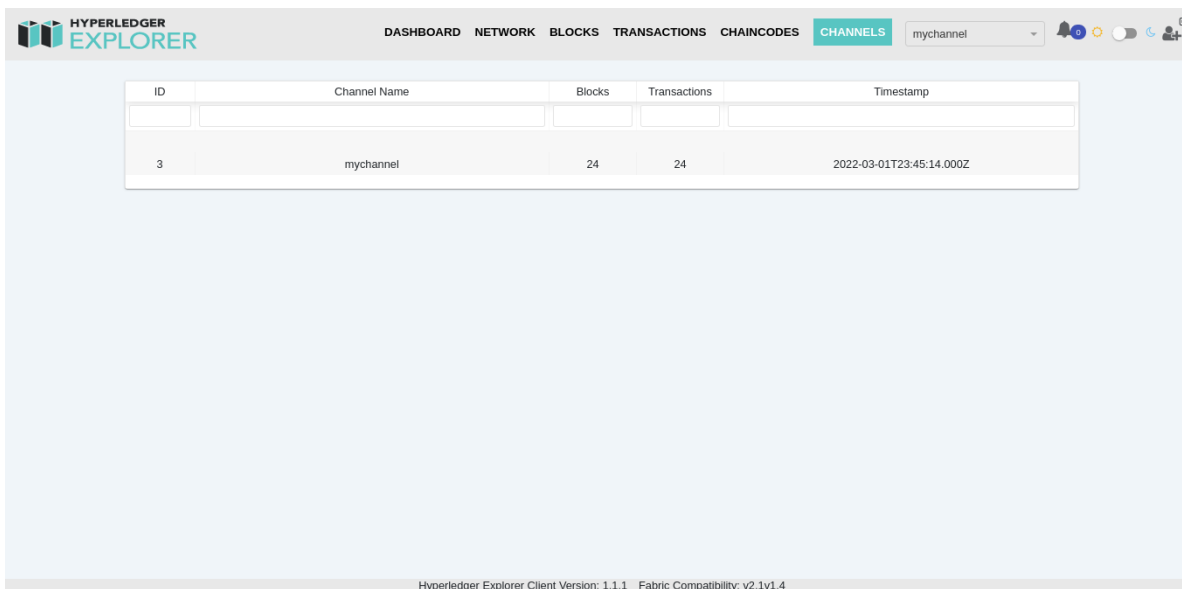


The screenshot shows the Hyperledger Explorer interface with the 'CHAINCODES' tab selected. A table displays the following data:

Chaincode Name	Channel Name	Path	Transaction Count	Version
fabcar	mychannel	-	2	1

At the bottom of the interface, the text reads: "Hyperledger Explorer Client Version: 1.1.1 Fabric Compatibility: v2.1v1.4"

Figura 5.24: Sección “Chaincode” en *Hyperledger Explorer*



The screenshot shows the Hyperledger Explorer interface with the 'CHANNELS' tab selected. A table displays the following data:

ID	Channel Name	Blocks	Transactions	Timestamp
3	mychannel	24	24	2022-03-01T23:45:14.000Z

At the bottom of the interface, the text reads: "Hyperledger Explorer Client Version: 1.1.1 Fabric Compatibility: v2.1v1.4"

Figura 5.25: Sección “Channels” en *Hyperledger Explorer*

Sin duda, la herramienta *Hyperledger Explorer*, resulta de mucha ayuda al momento de administrar las entidades y reglas de negocio específicas para el sistema de verificación vehicular, ya que es posible visualizar de forma gráfica la aplicación y el cómo es que se generan las funciones *hash* concatenadas unas a otras hasta lograr construir el libro mayor que contiene la lógica de un *blockchain* privado. Así como la verificación de fallas en la creación de los contratos inteligentes, comunicación entre entidades, tipos de canales. Es importante resaltar que el uso de *Hyperledger Explorer* es únicamente informativo, ya que resulta imposible manipular características previamente establecidas en la red.

Capítulo 6. Conclusiones y trabajo futuro

Uno de los principales problemas con los que la mayoría de los mexicanos hemos tenido que lidiar es la corrupción, desafortunadamente México es conocido por ser uno de los países con más alto índice de corrupción a nivel mundial, según el Índice de Percepción de Corrupción (IPC) [29], el país se encuentra en la posición 31 de 100 donde 0 es el nivel más alto corrupción. Lo que nos da una idea de la cantidad de actos ilegales que a diario se suscitan en los diferentes niveles de gobierno, instituciones y servidores públicos.

Si bien el proceso de verificación vehicular nació como una propuesta a favor del medio ambiente para contrarrestar la emisión de partículas dañinas, este fue rápidamente apoderado de actos ilegales mediante la manipulación de protocolos y resultados que beneficiaban a los ciudadanos a cambio de una cuota monetario que no precisamente se destinaba al gobierno. Por lo que el gobierno propuso implementar una serie de protocolos más rigurosos que ayudaban a evitar la colusión entre el personal de los centros de verificación y el ciudadano, sin embargo, la mayoría de estos protocolos se encuentran obsoletos y resulta muy difícil dar seguimiento a algún acto de corrupción suscitado dentro de las instalaciones.

Es por lo que nació la idea de implementar un sistema robusto basado en la filosofía de *blockchain* que permitiera realizar operaciones de forma descentralizada en los más de 100 centros de verificación vehicular del Estado de México, lo que ayudaría a generar confianza entre los usuarios y operadores del sistema. Así mismo involucrar a todos los centros de verificación en la validación de transacciones de manera colectiva, lo que permite erradicar la colusión entre entidades, sin mencionar que indirectamente se genera un ambiente de transparencia en la que cualquier CVV es capaz de ver las transacciones que se realizan en otros CVV remotos.

Uno de los retos más desafiantes en el desarrollo de la presente implementación fue elegir el tipo de *blockchain* a utilizar, ya que como se menciona a lo largo del capítulo 2, existen arquitecturas que cubren necesidades propias que la lógica del negocio impone, en

este caso en particular y con ayuda del análisis de requerimientos realizados como resultado de la investigación previa, fue que se decidió optar por un *blockchain* privado no permissionado que aporta una capa de seguridad a la información sensible albergada en un libro mayor garantizando que todas las entidades participantes se encuentran previamente registradas y verificadas por una entidad certificadora como lo es el “*orderer*”.

Por otro lado, el uso de un contrato inteligente fue parte clave en la creación y verificación de transacciones en tiempo real, lo que ayudó de manera significativa en optimizar el tiempo total en el que una transacción es procesada, ya que a diferencia del protocolo actual en donde las tareas de registro, toma de datos, pruebas OBS, evaluación de resultados y asignación de holograma, eran tareas que se repartían entre los integrantes del centro de verificación vehicular. Ahora la ejecución del *smart contract* se realiza de forma automática sin posibilidad de manipular el contenido de los resultados obtenidos en la toma de valores para cada unidad. Esta idea se encuentra sustentada gracias al uso de funciones hash y firmas digitales vistas en capítulos anteriores.

Otro punto a resaltar es el uso de la red *blockchain* en un sistema distribuido real, ya que la implementación presentada es un prototipo local que funciona simulando las organizaciones en contenedores Docker independientes. Sin embargo si se desea utilizar la aplicación en un entorno real se deberá alojar cada uno de estos contenedores en un servidor especializado para mantener la operación durante las horas en las que se provee el servicio. Así mismo se deberá asignar al personal necesario para poder validar las transacciones asignadas a cada una de las organizaciones, siguiendo las normas del protocolo RAFT presentado en el capítulo 2.

Por lo tanto, como conclusiones particulares se destaca lo siguiente:

1. Se logró abstraer, diseñar, e implementar la arquitectura de un sistema altamente concurrente como lo es el sistema de verificación vehicular, a una tecnología distribuida capaz de albergar datos de forma segura y eficiente, dando prioridad en todo momento al cumplimiento del objetivo principal del trabajo, que es generar transparencia en los procesos de verificación.

2. Con ayuda de un contrato inteligente y mediante el protocolo de consenso RAFT se asegura que se cumplan los procedimientos adecuados para la evaluación de los vehículos verificados en los CVV, ya que en el supuesto caso que algún empleado u organización quisiera coludirse con la persona encargada de validar sus propias transacciones, este tendría que convencer a los más de 100 verificentros (debido a que no siempre es el mismo CVV quien valida) para poder llevar a cabo un acto de corrupción, por lo que no sería una acción redituable para llevar a cabo.

3. Por otra parte, al trabajar con un *blockchain* privado, se asegura en todo momento que las entidades que aportan información al *ledger* son legítimas. Así mismo la arquitectura permite escalar de forma fácil cuando sea necesario el agregar más organizaciones, *peer*, canales de comunicación, contratos inteligentes y entidades certificadoras.

4. Finalmente, se logró cumplir de forma éxito los objetivos presentados, teniendo como base, la lógica que utiliza una de las tecnologías más explotadas de la historia moderna de la computación, además, se logró obtener el conocimiento suficiente para comprender otros temas relacionados a *blockchain*, como lo son las criptomonedas y los sistemas de rastreo que empiezan a tomar relevancia en los procesos con los que a diario nos rodean.

6.1 Trabajo Futuro

Si bien los resultados obtenidos con la aplicación desarrollada ayudan a generar transparencia en los procesos de verificación vehicular, así como reducir los tiempos de procesamiento en cada uno de los centros de verificación, existen áreas de este trabajo que pueden ser mejoradas a partir del trabajo presentado. A continuación, se presenta una lista de propuestas para su futuro desarrollo.

- Creación de estadísticos y uso de datos en proyecciones de autos en horas pico, lo que ayudaría a organizar las plantillas de empleados en cada uno de los CVV, dependiendo la zona donde se ubiquen, la demanda de usuarios, así como los días más concurridos en estos establecimientos, para así contar con el recurso humano suficiente en cada unidad.
- Inspección visual mediante inteligencia artificial. Si bien el sistema presentado le provee la capacidad al técnico verificador de llenar los datos correspondientes a la

prueba visual (revisión de neumáticos, tapón de gasolina, varilla de aceite, entre otros) uno de los trabajos futuros más ambiciosos es el poder validar el cumplimiento de los parámetros mencionados a través del uso de las cámaras que actualmente se encuentran funcionando en los CVV. Esto con ayuda de inteligencia artificial y reconocimiento de imágenes, en donde el sistema sea capaz de detectar si el automóvil cuenta con tapón de gasolina, varilla de aceite, el estado de los neumáticos, entre otros parámetros.

- Automatización en la asignación de hologramas. Si bien el sistema actual genera los datos mediante un *software* conectado a los barómetros ubicados en las líneas de verificación, estos deben ser validados y evaluados por el personal calificado basándose en los parámetros establecidos por la Secretaría del Medio Ambiente en las Tablas 4.2, 4.3 y 4.4. Sin embargo, esta tarea pudiera ser automatizada mediante un sistema de validación de datos y toma de decisiones a partir de los parámetros obtenidos en las pruebas.
- Sistema de auditoría. Con ayuda de los datos recabados en el libro mayor, es posible hacer informes de auditoría de forma detallada, teniendo la capacidad de rastrear el históricos de cada uno de los vehículos que se hayan sometido a las pruebas de verificación, dando seguimiento a denuncias impuestas por corrupción en los más de 100 centros de verificación.

Bibliografía

- [1]. COMUNICADO DE PRENSA NÚM. 496/21 24 DE AGOSTO DE 2021 PÁGINA 2/2 EVALUACIÓN DE SERVICIOS PÚBLICOS BÁSICOS Y DE INFRAESTRUCTURA Se anexa Nota Técnica. (n.d.). Retrieved September 9, 2022, from <https://www.inegi.org.mx/programas/encrige/2020/>
- [2]. SUMAN 28 VERIFICENTROS SUSPENDIDOS POR IRREGULARIDADES Y CORRUPCIÓN. (2016). https://www.ipomex.org.mx/recursos/ipo/files_ipo/2015/21/11/2c26eaa96a5d4a81f878a754f9174b15.pdf
- [3] Qué son los “smart contracts” o contratos inteligentes | BBVA. (n.d.). Retrieved February 9, 2022, from <https://www.bbva.com/es/smart-contracts-los-contratos-basados-blockchain-no-necesitan-abogados/>
- [4]. Pahlajani, S., Kshirsagar, A., & Pachghare, V. (2019). Survey on Private Blockchain Consensus Algorithms. Proceedings of 1st International Conference on Innovations in Information and Communication Technology, ICIICT 2019. <https://doi.org/10.1109/ICIICT1.2019.8741353>
- [5]. La blockchain : fundamentos, aplicaciones y relación con otras tecnologías disruptivas. - Dialnet. (n.d.). Retrieved July 21, 2021, from <https://dialnet.unirioja.es/servlet/articulo?codigo=6207510>
- [6]. Pablo, J., & Romero, C. (n.d.). Trabajo Fin de Máster Aplicación de Blockchain para el uso de transportes Blockchain application for transport use.
- [7]. Nakamoto, S. (n.d.). Bitcoin: A Peer-to-Peer Electronic Cash System. www.bitcoin.org
- [8]. La blockchain : fundamentos, aplicaciones y relación con otras tecnologías disruptivas. - Dialnet. (n.d.). Retrieved July 21, 2021, from <https://dialnet.unirioja.es/servlet/articulo?codigo=6207510>
- [9]. Groth, P., and Moreau, L. Recording Process Documentation for Provenance. IEEE Transactions on Parallel and Distributed Systems 20, 9 (Sept. 2009), 1246–1259.

- [10]. Aldeco-Pérez, R., and Moreau, L. Provenance-based Auditing of Private Data Use. In International Academic Research Conference, Visions of Computer Science (BSC 08) (London, UK, 2008), BCS, pp. 141–152
- [11]. Rocío Aldeco-Pérez , Gualberto Aguilar Torres, Nareli Cruz Cortés, Luis J. Domínguez Perez, Ponciano J. Escamilla Ambrosio, Gina Gallegos García, Miguel A. León Chavez, Raúl Monroy Borja, Lil M. Rodríguez Henríquez, Francisco J. Rodríguez Henríquez, Abraham Rodríguez Mota, Moisés Salinas Rosales, Alejandra G. Silva Trujillo, "Introducción a la Ciberseguridad y sus aplicaciones en México", Academia Mexicana de Computación, A. C., México, Noviembre 2020, ISBN 78-607-98941-2-2.
http://amexcomp.mx/files/libro/Intro_a_la_Ciberseguridad_y_sus_Aplicaciones_en_Mexico.pdf
- [12]. Menezes, A., van Oorschot, P., Vanstone, S., Rosen, K. (1997). "Handbook of Applied Cryptography". Boca Raton: CRC Press.
- [13]. Blockchain pública vs privada: ¿cuál es la diferencia? | Logística | Apuntes empresariales | ESAN. (n.d.). Retrieved August 2, 2021, from <https://www.esan.edu.pe/apuntes-empresariales/2019/12/blockchain-publica-vs-privada-cual-es-la-diferencia-1/>
- [14]. Allende, M. (2018, June). Conoce los distintos tipos de blockchain. <https://blogs.iadb.org/conocimiento-abierto/es/tipos-de-blockchain/>
- [15]. Raikwar, M., Gligoroski, D., & Krlevska, K. (2019). SoK of Used Cryptography in Blockchain. IEEE Access, 7, 148550–148575.
<https://doi.org/10.1109/ACCESS.2019.2946983>
- [16]. E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yel-lick, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in Proceedings of the Thirteenth EuroSys Conference, ser. EuroSys '18. New York, NY, USA: ACM, 2018, pp. 30:1–30:15.
- [17]. Sankar, L. S., Sindhu, M., & Sethumadhavan, M. (2017, August 22). Survey of consensus protocols on blockchain applications. 2017 4th International Conference on Advanced Computing and Communication Systems, ICACCS 2017.
<https://doi.org/10.1109/ICACCS.2017.8014672>

- [18]. Ismail, L., & Materwala, H. (2019). A Review of Blockchain Architecture and Consensus Protocols: Use Cases, Challenges, and Solutions. *Symmetry* 2019, Vol. 11, Page 1198, 11(10), 1198. <https://doi.org/10.3390/SYM11101198>
- [19]. Introducción — documentación de hyperledger-fabricdocs - master. (n.d.). Retrieved August 3, 2021, from <https://hyperledger-fabric.readthedocs.io/es/latest/whatis.html>
- [20]. ¿Qué es una licitación pública y cómo funciona? (n.d.). Retrieved August 26, 2021, from <https://www.latamcompra.com/service/Noticias/%C2%BFSabes-qu%C3%A9-es-una-licitaci%C3%B3n%3F>
- [21]. X-Road® Technology Overview — X-Road® Data Exchange Layer. (n.d.). Retrieved February 9, 2022, from <https://x-road.global/x-road-technology-overview>
- [22]. IBM Food Trust - Blockchain for the world's food supply | IBM. (n.d.). Retrieved August 30, 2021, from <https://www.ibm.com/blockchain/solutions/food-trust>
- [23]. SEDEMA | Centros de verificación vehicular. (n.d.). Retrieved August 31, 2021, from <http://www.data.sedema.cdmx.gob.mx/centros-verificacion-vehicular/historia4.html>
- [24]. Verificación Vehicular. (n.d.). Retrieved August 31, 2021, from <https://sedema.cdmx.gob.mx/programas/programa/verificacion-vehicular>
- [25]. Informe Integral 023-0013-2018, a las visitas técnicas de supervisión a Centros de Verificación Vehicular autorizados en el Estado de México. (2018). <http://transparenciafiscal.edomex.gob.mx/sites/transparenciafiscal.edomex.gob.mx/files/files/auditorias/2019/023-0013-2019.pdf>
- [26]. ¿Qué es Docker? (n.d.). Retrieved September 9, 2021, from <https://www.redhat.com/es/topics/containers/what-is-docker>
- [27]. Blockchain network — hyperledger-fabricdocs master documentation. (n.d.). Retrieved February 13, 2022, from <https://hyperledger-fabric.readthedocs.io/en/release-2.2/network/network.html>
- [28]. Servicio de Registro de Citas de Verificación Vehicular. (n.d.). Retrieved June 3, 2021, from <https://citaverificacion.edomex.gob.mx/RegistroCitas/>
- [29]. México mejora su posición en Índice de Percepción de Corrupción | Capital 21 | NOTICIAS. (n.d.). Retrieved March 1, 2022, from <https://www.capital21.cdmx.gob.mx/noticias/?p=9335>
- [30]. Centro Nacional de Metrología. (s/f). ¿Qué es un dinamómetro vehicular y cómo se usa para la medición de emisiones contaminantes de los vehículos? gob.mx. Recuperado el 26 de abril de 2022, de <https://www.gob.mx/cenam/articulos/que-es-un-dinamometro->

vehicular-y-como-se-usa-para-la-medicion-de-emisiones-contaminantes-de-los-vehiculos?idiom=es

[31]. rest-apis. (s/f). Ibm.com. Recuperado el 26 de abril de 2022, de <https://www.ibm.com/mx-es/cloud/learn/rest-apis>

[32]. Ruelas, U. (s/f). ¿Qué es la serialización o marshalling? Codingornot.com. Recuperado el 26 de abril de 2022, de <https://codingornot.com/que-es-la-serializacion-o-marshalling>

[33]. A new blockchain solution for the certification of vehicle compliance at European level will be implemented - Renault Group. (s/f). Renaultgroup.Com. Recuperado el 3 de mayo de 2022, de <https://www.renaultgroup.com/en/news-on-air/news/xceed-a-new-blockchain-solution-for-renault-plants-in-europe/>

[34]. Dobson, N. (2020, septiembre 22). Bank guarantees: Jumping from paper to blockchain. IBM Supply Chain and Blockchain Blog. <https://www.ibm.com/blogs/blockchain/2020/09/bank-guarantees-jumping-from-paper-to-blockchain/>

[35]. la Integridad, A. P. (s/f). MEDICIONES INTERNACIONALES. BANCO DE BUENAS PRÁCTICAS INTERNACIONALES EN EL COMBATE A LA CORRUPCIÓN. Recuperado el 26 de junio de 2022, de <https://banco.sesna.gob.mx/mediciones-internacionales/>

[36]. Instituto Nacional de Estadística y Geografía. (s/f). Encuesta Nacional de Calidad e Impacto Gubernamental (ENCIG) 2021 [Data set].

[37]. de la Megalópolis, C. A. (s/f). IMECA: Índice METropolitano de la Calidad del Aire. gob.mx. Recuperado el 26 de junio de 2022, de <https://www.gob.mx/comisionambiental/articulos/imeca-indice-metropolitano-de-la-calidad-del-aire?idiom=es>

[38]. Así se mide la corrupción en México. (2020, 19 diciembre). Deloitte México. Recuperado 26 de junio de 2022, de <https://www2.deloitte.com/mx/es/pages/dnoticias/articles/asi-se-mide-corrupcion-mexico.html>

Anexo A. Código

Código de contrato inteligente

```
package main
import (
    "bytes"
    "encoding/json"
    "fmt"
    "strconv"
    "time"

    "github.com/hyperledger/fabric-contract-api-go/contractapi"
    "github.com/hyperledger/fabric/common/flogging"
)
type SmartContract struct {
    contractapi.Contract
}
var logger = flogging.MustGetLogger("fabcar_cc")
type Car struct {
    ID string `json:"id"`
    HologramaObtenido string `json:"hologramaObtenido"`
    Niv string `json:"niv"`
    Comentarios string `json:"comentarios"`
    Marca string `json:"marca"`
    Modelo string `json:"modelo"`
    Placas string `json:"placas"`
    VerificentroId string `json:"verificentroid"`
    TecnicoId string `json:"tecnicoid"`
    OdometroId string `json:"odometroid"`
    CCWValId string `json:"ccvvalid"`
    ValidadorId string `json:"validadorid"`
    LineaVerifica string `json:"lineaverifica"`
    Status string `json:"status"`
    TapaGasolina string `json:"tapagasolina"`
    BayonetaAceite string `json:"bayonetaaceite"`
    FiltroAire string `json:"filtroaire"`
    TuboEscape string `json:"tuboescape"`
    TaponRadiador string `json:"taponradiador"`
    MangueraVacio string `json:"mangueravacio"`
    Ruedas string `json:"ruedas"`
    LucesTyD string `json:"lucestyd"`
    CO string `json:"co"`
    CO2 string `json:"co2"`
    O2 string `json:"o2"`
    NOxppm string `json:"noxppm"`
    HCxppm string `json:"hidrocarburo"`
    Lambda string `json:"lambda"`
    CreateDate string `json:"createdate"`
    UupdateDate string `json:"updatedate"`
    AddedAt uint64 `json:"addedAt"`
}
```

```

func (s *SmartContract) CreateCar(ctx contractapi.TransactionContextInterface,
carData string) (string, error) {
    if len(carData) == 0 {
        return "", fmt.Errorf("Please pass the correct car data")
    }

    var car Car
    err := json.Unmarshal([]byte(carData), &car)
    if err != nil {
        return "", fmt.Errorf("Failed while unmarshaling car. %s",
err.Error())
    }
    carAsBytes, err := json.Marshal(car)
    if err != nil {
        return "", fmt.Errorf("Failed while marshaling car. %s",
err.Error())
    }
    ctx.GetStub().SetEvent("CreateAsset", carAsBytes)

    return ctx.GetStub().GetTxID(), ctx.GetStub().PutState(car.ID,
carAsBytes)
}
func (s *SmartContract) GetHistoryForAsset(ctx
contractapi.TransactionContextInterface, carID string) (string, error) {
    resultsIterator, err := ctx.GetStub().GetHistoryForKey(carID)
    if err != nil {
        return "", fmt.Errorf(err.Error())
    }
    defer resultsIterator.Close()
    var buffer bytes.Buffer
    buffer.WriteString("[")
    bArrayMemberAlreadyWritten := false
    for resultsIterator.HasNext() {
        response, err := resultsIterator.Next()
        if err != nil {
            return "", fmt.Errorf(err.Error())
        }
        if bArrayMemberAlreadyWritten == true {
            buffer.WriteString(",")
        }
        buffer.WriteString("{\"TxId\":")
        buffer.WriteString("\")")
        buffer.WriteString(response.TxId)
        buffer.WriteString("\")")

        buffer.WriteString(", \"Value\":")
        if response.IsDelete {
            buffer.WriteString("null")
        } else {
            buffer.WriteString(string(response.Value))
        }
        buffer.WriteString(", \"Timestamp\":")
        buffer.WriteString("\")")
        buffer.WriteString(time.Unix(response.Timestamp.Seconds,
int64(response.Timestamp.Nanos)).String())

```

```

        buffer.WriteString("\\"")

        buffer.WriteString(", \"IsDelete\":")
        buffer.WriteString("\\"")
        buffer.WriteString(strconv.FormatBool(response.IsDelete))
        buffer.WriteString("\\"")

        buffer.WriteString("}")
        bArrayMemberAlreadyWritten = true
    }
    buffer.WriteString("]")
    return string(buffer.Bytes()), nil
}

func (s *SmartContract) GetCarById(ctx contractapi.TransactionContextInterface,
carID string) (*Car, error) {
    if len(carID) == 0 {
        return nil, fmt.Errorf("Please provide correct contract Id")
        // return shim.Error("Incorrect number of arguments. Expecting 1")
    }
    carAsBytes, err := ctx.GetStub().GetState(carID)
    if err != nil {
        return nil, fmt.Errorf("Failed to read from world state. %s",
err.Error())
    }
    if carAsBytes == nil {
        return nil, fmt.Errorf("%s does not exist", carID)
    }
    car := new(Car)
    _ = json.Unmarshal(carAsBytes, car)
    return car, nil
}

func (s *SmartContract) DeleteCarById(ctx
contractapi.TransactionContextInterface, carID string) (string, error) {
    if len(carID) == 0 {
        return "", fmt.Errorf("Please provide correct contract Id")
    }

    return ctx.GetStub().GetTxID(), ctx.GetStub().DelState(carID)
}

func (s *SmartContract) GetContractsForQuery(ctx
contractapi.TransactionContextInterface, queryString string) ([]Car, error) {
    queryResults, err := s.getQueryResultForQueryString(ctx, queryString)
    if err != nil {
        return nil, fmt.Errorf("Failed to read from ----world state. %s",
err.Error())
    }
    return queryResults, nil
}

func (s *SmartContract) getQueryResultForQueryString(ctx
contractapi.TransactionContextInterface, queryString string) ([]Car, error) {
    resultsIterator, err := ctx.GetStub().GetQueryResult(queryString)

```

```

    if err != nil {
        return nil, err
    }
    defer resultsIterator.Close()
    results := []Car{}

    for resultsIterator.HasNext() {
        response, err := resultsIterator.Next()
        if err != nil {
            return nil, err
        }
        newCar := new(Car)
        err = json.Unmarshal(response.Value, newCar)
        if err != nil {
            return nil, err
        }
        results = append(results, *newCar)
    }
    return results, nil
}

func (s *SmartContract) GetDocumentUsingCarContract(ctx
contractapi.TransactionContextInterface, documentID string) (string, error) {
    if len(documentID) == 0 {
        return "", fmt.Errorf("Please provide correct contract Id")
    }
    params := []string{"GetDocumentById", documentID}
    queryArgs := make([][]byte, len(params))
    for i, arg := range params {
        queryArgs[i] = []byte(arg)
    }
    response := ctx.GetStub().InvokeChaincode("document_cc", queryArgs,
"mychannel")
    return string(response.Payload), nil
}

func (s *SmartContract) CreateDocumentUsingCarContract(ctx
contractapi.TransactionContextInterface, functionName string, documentData
string) (string, error) {
    if len(documentData) == 0 {
        return "", fmt.Errorf("Please provide correct document data")
    }
    params := []string{functionName, documentData}
    queryArgs := make([][]byte, len(params))
    for i, arg := range params {
        queryArgs[i] = []byte(arg)
    }
    response := ctx.GetStub().InvokeChaincode("document_cc", queryArgs,
"mychannel")
    return string(response.Payload), nil
}

func main() {

```

```
chaincode, err := contractapi.NewChaincode(new(SmartContract))
if err != nil {
    fmt.Printf("Error create fabcar chaincode: %s", err.Error())
    return
}
if err := chaincode.Start(); err != nil {
    fmt.Printf("Error starting chaincodes: %s", err.Error())
}
}
```

Código fuente de la implementación

- Fronted: <https://github.com/Jehosua97/Front-Tesis>
- Backend: <https://github.com/Jehosua97/Fabric>