Universidad de Buenos Aires
Facultades de Ciencias Económicas, Ciencias Exactas y Naturales e Ingeniería

Maestría en Seguridad Informática

Tesis de Maestría

Criptografía

Impacto de la tecnología blockchain en el armado de Smart Contracts

Autor: Esp. Rick Marcel Cevallos García
Tutor de Trabajo Final: PhD. Pedro Hecht

2020
Cohorte 2017

Declaración Jurada de origen de los contenidos

“Por medio de la presente, el autor manifiesta conocer y aceptar el Reglamento de Tesis vigente y que se hace responsable que la totalidad de los contenidos del presente documento son originales y de su creación exclusiva, o bien pertenecen a terceros u otras fuentes, que han sido adecuadamente referenciados y cuya inclusión no infringe la legislación Nacional e Internacional de Propiedad Intelectual”

_____

Rick Marcel Cevallos García

C.I. 0.925.351.181 Guayaquil, Ecuador

DNI. 90.712.064 CABA., Argentina

Resumen

El presente trabajo de tesis de maestría muestra un análisis holístico de los contratos Inteligentes mejor conocidos por su traducción al inglés Smart Contracts basados en cadena de bloques o Blockchain.

Se comienza con un recorrido minucioso por la tecnología Blockchain detallando tanto sus componentes, como arquitecturas posibles y sus implementaciones existentes, luego se analiza los Smart Contracts buscando incluir la mayor cantidad de desarrollos presentes, tanto aquellos que se encuentran operativos como las apuestas más prometedoras del mercado. El objetivo de la revisión antes mencionada, es conocer la mayor cantidad de opciones posibles para efectuar una prueba de concepto, que permita tener un entendimiento práctico sobre las ventajas y desventajas de esta tecnología las cuales serán una importante base de las conclusiones del trabajo.

Se deja constancia que el costo de las criptodivisas utilizadas en las plataformas que permiten desarrollar contratos inteligentes tiende a sufrir continuas variaciones, por lo cual en el desarrollo se menciona la fecha de las cotizaciones utilizadas tanto en dólares estadounidenses como pesos argentinos para la determinación de los costos por transacción encontrados en la prueba de concepto.

Palabras claves

Prueba de concepto, contratos inteligentes, cadena de bloques, criptodivisas, cotizaciones.

Índice General

Agradecimientos

Agradezco a Dios y a mi familia, en especial a mi hermana Chriss por la ayuda que me han brindado, sin ellos no habría podido estudiar en este País que amo.

A mis profesores y a mis amigos por el conocimiento y el apoyo que me han transmitido durante este proceso.

Índice de figuras

Índice de tablas

Prólogo

Posiblemente Satoshi Nakamoto jamás llegó a imaginar que el trabajo que un día publicó sobre el protocolo Bitcoin, y por consiguiente la tecnología en la cual se basa, Blockchain, llegaría a tener el impacto tanto mediático como técnico inclusive comercial que en la actualidad ha logrado.

Se podría hacer un paralelismo entre la invención del transistor el cual se encuentra prácticamente en todos los aparatos electrónicos existentes, con el ingenio de la cadena de bloques en la informática, para el momento que me encuentro escribiendo este prólogo existen desarrollos basados en Blockchain que han revolucionado áreas tales como, los servicios prestados en la nube activos económicos de resguardo e inclusive pretenden alcanzar el área comercial y el sector legal, especialmente mediante los llamados contratos inteligentes.

A priori conjeturar el alcance del último ejemplo mencionado en el párrafo anterior, posiblemente no llegue a significar ni la mitad de todos los usos que tendrán en los próximos años.

Como opinión personal pienso que esta solución cumple los requisitos necesarios para ser una tecnología capaz de adaptarse a distintas necesidades y llegar a implementarse a gran escala a futuro, por ello es que con mira a las soluciones futuras a procesos actuales seleccioné este tópico como tema de tesis de maestría.

Nómina de abreviaturas

API	Application Program Interface
BaaS	Blockchain as a Service
BIP	Bitcoin Improvement Proposals
DApp	Decentralize Application
DPoS	Delegated Proof of Stake
EdDSA	Edwards-curve Digital Signature Algorithm
EOA	Externally Owned Accounts
EVM	Ethereum Virtual Machine
IDE	Integrated Development Environment
NeoVM	NEO Virtual Machine
PoC	Proof of Concept
PoS	Proof of Stake
PoW	Proof of Work
P2P	Peer to Peer
SaaS	Software as a Service
SCP	Stellar Consensus Protocol
SDK	Software Development Kit
SSC	Stellar Smart Contract
UETA	Uniform Electronic Transaction Act

CAPÍTULO 1: Blockchain

1.1. Definición

Para comprender el impacto de la tecnología blockchain en el armado de los contratos inteligentes (Smart contracts), de los cuales se hablará en el segundo capítulo, es necesario conocer primero, ¿Qué es Blockchain?.

La tecnología Blockchain o cadena de bloques, introducida por Satoshi Nakamoto en el año 2008, puede definirse como una base de datos distribuida que registra todas las transacciones que han ocurrido en la red blockchain, la cual se replica y se comparte entre los participantes de la red. Esta característica ha demostrado permitir a entidades que no son de confianza, comunicarse y enviar transacciones entre sí de forma segura sin la necesidad de un tercero de confianza. [1]

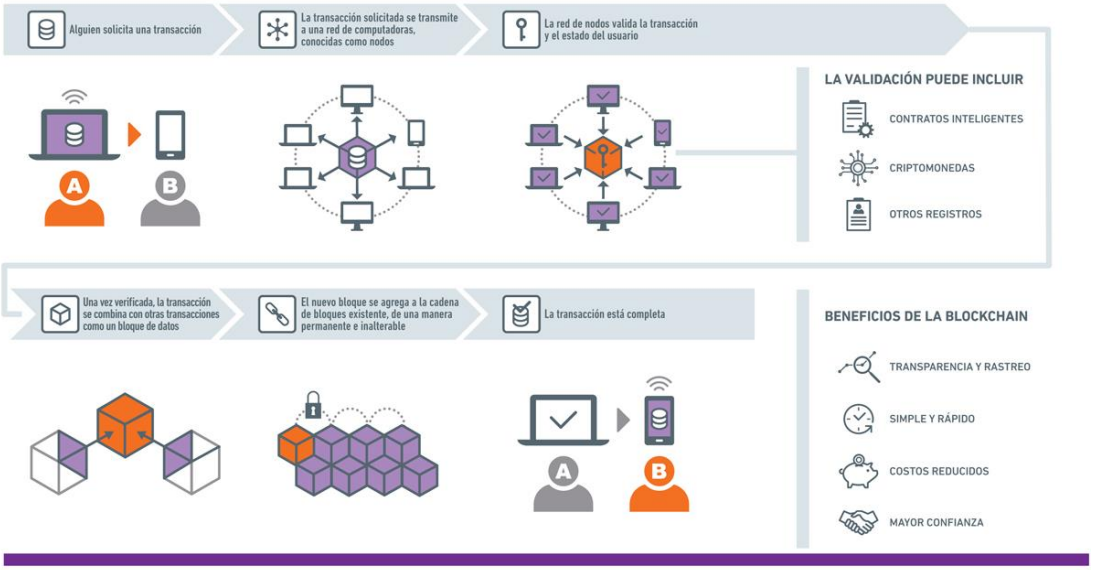

Figura 1.1: Funcionamiento de Blockchain. [2]

A partir de la figura 1.1, se puede describir ciertas características de los sistemas Blockchain:

- Sistemas descentralizados: La tecnología Blockchain está siendo vista como una tecnología poderosa para descentralizar la web. Al no existir un único ente regulador, los usuarios pueden mantener el control de toda su información y transacciones, así como realizar un intercambio sin la intermediación de un tercero, eliminando así el riesgo de un tercero en la operación. [3]
- Transparencia: Los cambios en las cadenas de bloques públicas pueden ser vistos por todas las partes, además cada usuario de la red puede validar las transacciones, lo que permite que exista

transparencia, y todas las transacciones tengan la característica de no poder ser alteradas o eliminadas.

- Resistencia al ataque: Los sistemas descentralizados son más caros de atacar o destruir. Debido a que las redes Blockchain no tienen un punto central de control, poseen una mayor capacidad de sobrevivir a los ataques maliciosos, en una configuración de Blockchain, los bloques de datos se enlazan y se aseguran mediante métodos criptográficos. [4]
- Velocidad de Transacciones: Mientras que en la actualidad sistemas como los bancarios pueden llegar procesar transacciones en horas o incluso días, el procesamiento de transacciones realizadas en una cadena de bloques puede ser realizado en pocos minutos y lograr las mismas garantías de transparencia que las exigidas en la actualidad. [5]
- Doble gasto: Para que se apruebe una transacción y esta se almacene en un “libro mayor universal”, los usuarios del sistema deben aprobar la transacción, los cuales podrán rechazarla si consideran que esta ya se ha realizado, o en el caso de las criptomonedas, si el comprador ha utilizado anteriormente el dinero, esta se rechaza. [6]

1.2. Componentes

Blockchain, como se ha mencionado posee una arquitectura descentralizada, es decir, no depende de una autoridad central. Las transacciones son aprobadas por una gran cantidad de nodos distribuidos llamados mineros y se registran en bloques con marcas de tiempo, donde cada bloque se identifica mediante criptográficamente mediante un hash y se encadena a los bloques anteriores en orden cronológico.

Los componentes necesarios para desarrollar la arquitectura de un sistema Blockchain, son los siguientes:

- Nodo
- Transacción
- Bloque
- Cadena de Bloques
- Protocolo de Consenso

1.2.1. Nodo

Un nodo es un dispositivo en una red de bloques, el cual puede ser cualquier elemento electrónico activo, como por ejemplo una computadora, un teléfono

o incluso una impresora, siempre y cuando esté conectado a Internet y como tal tenga una dirección IP (ver figura 1.2).

La función de un nodo es dar soporte a la red, manteniendo copia de una sección o todos de los registros de las transacciones realizadas en una cadena de bloques (Libro Mayor) y en algunos casos procesar transacciones. Los nodos suelen estar dispuestos en la estructura de los árboles, lo que se conoce como árboles binarios. Como los propietarios de los nodos contribuyen voluntariamente con sus recursos informáticos para almacenar y validar las transacciones, tienen la oportunidad de cobrar las comisiones de transacción y ganar una recompensa en la criptodivisa subyacente por hacerlo. Esto se conoce como minería o forja. [7]

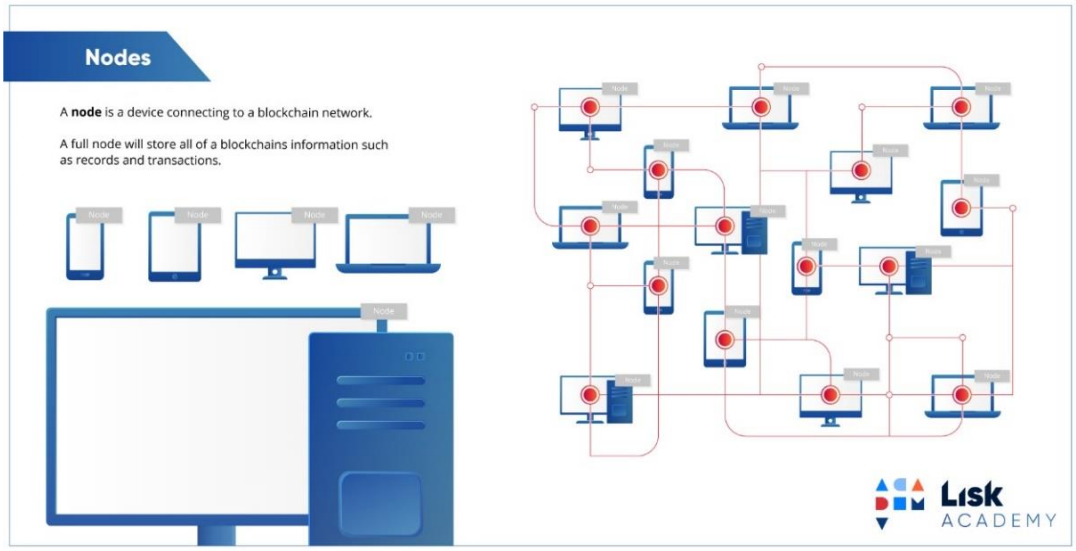

Figura 1.2: Nodos en Blockchain. [7]

Un nodo puede ser un punto final de comunicación o un punto de redistribución de la comunicación, enlazando con otros nodos. Cada nodo de la red se considera igual, sin embargo, ciertos nodos tienen diferentes roles en la forma en que soportan la red. Por ejemplo, no todos los nodos almacenarán una copia completa de una cadena de bloqueo o validarán las transacciones. [7]

1.2.2. Transacción

Las transacciones son los movimientos o acuerdos que se realizan en la red. Estas generalmente están compuestas por una dirección de destinatario, una dirección de remitente y un valor, son consideras como el propósito de una red Blockchain [8]

Las transacciones se agrupan y se entregan a cada nodo en forma de bloque. A medida que las nuevas transacciones se distribuyen por toda la red, son verificadas y "procesadas" de forma independiente por cada nodo. [8]

1.2.3. Bloque

Un bloque es esencialmente un conjunto de transacciones, cada bloque hace referencia al bloque que vino antes, lo que resulta en una cadena. Una vez que se crea un bloque y se agrega a la cadena, las transacciones contenidas en él no pueden ser modificadas o revertidas. Esto para asegurar la integridad de las transacciones y evitar el problema del doble gasto. [1]

Los bloques están constituidos por una cabecera y la lista de transacciones que el minero ha decidido incluir en el bloque que ha creado, el encabezado incluye metadatos que ayudan a verificar la validez de un bloque. Típicamente son 6 los campos que constituyen la cabecera del bloque: [8]

- Versión: indica la versión actual de la estructura de bloques.
- Hash de la cabecera del bloque anterior (Hash of Block N-1): referencia al bloque padre de este bloque
- Merkle root hash: un hash criptográfico de todas las transacciones incluidas en este bloque.
- Tiempo: hora en que se creó este bloque
- Difficulty Target: dificultad actual que se usó para crear este bloque
- Nonce ("número usado una vez"): valor aleatorio utilizado para generar un bloque.
- Estado: elemento opcional para dar mayor información de un bloque, en la figura 1.3, se muestra una cadena de bloques para Ethereum, donde este campo es necesario.

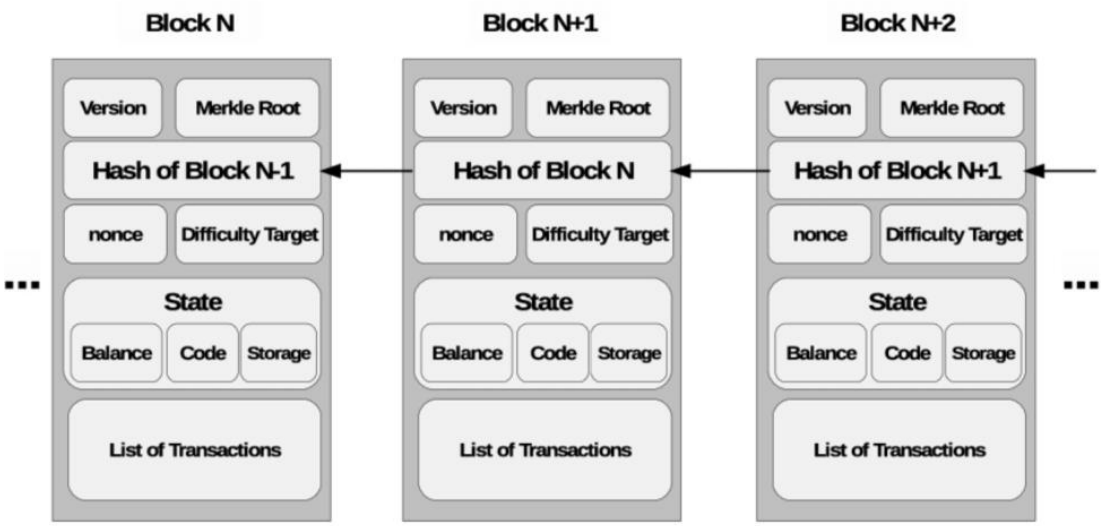

Figura 1.3: Cadena de bloques en Ethereum. [9]

1.2.4. Cadena de bloques

Es el resultado de la concatenación de bloques, es el registro universal de las transacciones realizadas en el Blockchain, el cual es público y se

encuentra parcial o totalmente almacenado en los nodos que componen la red.

Debido a que incluye todas las transacciones registradas, es utilizado por los nodos en la red para validar que una transacción en la lista por validar sea legítima y evitar el problema del doble gasto.

1.2.5. Consenso

En un sistema Blockchain, el protocolo de consenso es el método utilizado por nodos independientes que comprueban y verifican la validez de las transacciones y los bloques. Se aplica a todos los miembros de la red y permite mantener la confiabilidad y la seguridad en la misma. [10]

Debido a que cada sistema Blockchain tiene su propio método de consenso el algoritmo utilizado para llegar al consenso entre los mineros y mantener la santidad de los datos registrados en los bloques no es necesariamente el mismo para dos sistemas diferentes.

Los más conocidos son: Prueba de Trabajo (PoW) y Prueba de Participación (PoS).

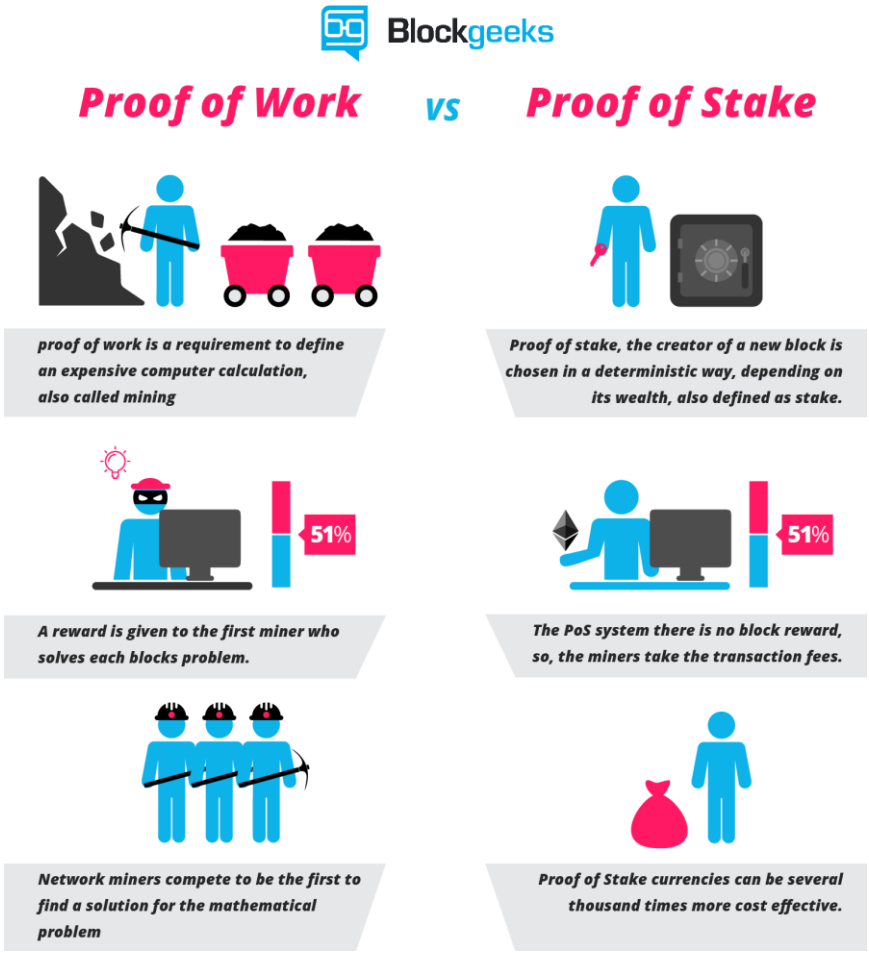

Figura 1.4: Diferencia entre Prueba de Trabajo y Prueba de Participación. [11]

1.2.5.1. Prueba de Trabajo (PoW)

La prueba de trabajo es un protocolo de consenso cuyo objetivo principal es la verificación descentralizada de transacciones, además de disuadir ciberataques del tipo denegación de servicio distribuido (DDoS), el cual tiene por objetivo agotar los recursos de un sistema informático mediante el envío de múltiples solicitudes falsas.

La idea de PoW fue publicada originalmente por Cynthia Dwork y Moni Naor en 1993, pero el término "prueba de trabajo" fue acuñado por Markus Jakobsson y Ari Juels en un documento publicado en 1999. Satoshi Nakamoto lo aplicó en el diseño de su sistema Blockchain para la verificación de transacciones. [12]

Para que un participante de la red pueda cumplir con la PoW impuesta en el sistema, debe buscar la respuesta a un problema matemático el cual requiere un trabajo considerable para poder ser resuelto, pero que es fácilmente verificable una vez alcanzada la solución.

Las PoW más conocidas utilizan las funciones hash, por ejemplo, validar que la función de hash de cada bloque sea menor que un valor específico determinado, como en la figura 1.5. Sin embargo, un sistema Blockchain puede utilizar otros métodos de PoW. Un ejemplo de esto es la búsqueda de un Script como algoritmo de prueba de trabajo en lugar de las funciones hash. [13]

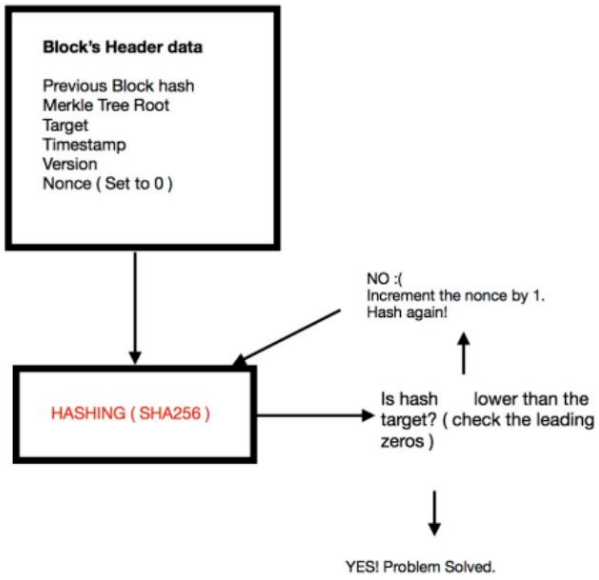

Figura 1.5: Ejemplo de PoW utilizando funciones Hash. [14]

Los sistemas que poseen PoW como método de consenso demandan el uso de gran cantidad de energía, existen propuestas para definir un sistema Blockchain exclusivo para el cálculo científico, donde una solución correcta a un determinado problema podría actuar como un método de validación, de

esta manera, la potencia de cálculo utilizada podría ayudar a resolver problemas científicos y contribuir a la investigación.

1.2.5.2. Prueba de Participación (PoS)

La prueba de participación es un mecanismo de consenso diferente a la prueba de trabajo, permite validar las transacciones basadas en el consenso distribuido. El sistema fue inicialmente sugerido en el foro de bitcointalk en 2011, la primera moneda digital en utilizar este método fue Peercoin en 2012 a continuación, le siguieron otras como ShadowCash, Nxt, BlackCoin, NuShares/NuBits, Qora y Nav Coin. [11]

Las principales ventajas de la PoS son la eficiencia energética y la seguridad. Se conoce que cada transacción de Bitcoin, que utiliza el sistema de PoW, puede requerir tanta electricidad como un hogar holandés promedio en dos semanas. Esto es tanto ineficaz como insostenible. [7]

La PoS impide la centralización, al otorgar parte de la riqueza de la creación de un bloque a varios mineros participantes, de lo contrario el individuo más rico del sistema siempre crearía el siguiente bloque y aumentaría consistentemente su riqueza resultando en el control del sistema.

1.3. Arquitectura

Debido a que Satoshi Nakamoto introdujo la tecnología Blockchain como código abierto, cualquier persona puede usar y modificar la arquitectura de las cadenas de bloques para implementar soluciones acordes a sus necesidades.

Actualmente se pueden distinguir tres tipos de arquitecturas:

- Pública
- Privada
- Consorcio

1.3.1. Pública

En este tipo de Blockchain, la red está completamente abierta y sin restricciones para poder unirse y participar en ella, se considera a la arquitectura pública en la cadena de bloques como irrestricta o *Permissionless.*

Se puede enviar y recibir transacciones de cualquier integrante de la red, además no existe limitantes para él o los miembros que deseen auditar el libro mayor del sistema, es decir, descargar el código y empezar a ejecutar un nodo público en un dispositivo local, validando las transacciones realizadas en la red y participando en el proceso de consenso. [15]

La característica *Permissionless* permite a los Blockchain públicos la existencia de mineros, los cuales son nodos que están dispuestos a

compartir su poder computacional para agregar bloques a la cadena a cambio de alguna recompensa. Para llegar a esta recompensa deben cumplir una prueba de trabajo o una prueba de participación, la forma en que se les recompensa depende de la implementación del protocolo de la cadena de bloques, sin embargo, a menudo implica una cuota (es decir, el nodo que solicitaba añadir datos paga al minero para que lo haga) o la creación de valor (por ejemplo, en el caso de Bitcoin, la cadena de bloques acuña valor y se lo da al minero que ha añadido un bloque). Así, los mineros están en competencia: todos quieren añadir el siguiente bloque, pero sólo uno o pocos de ellos lo lograrán para cada nuevo bloque. [16]

A pesar de la ventaja de evitar el costo de mantener la infraestructura, este tipo de arquitecturas poseen desventajas que en ciertos casos obligan a considerar opciones como Blockchain privados o consorcios. Una de ellas es su total apertura, (dependiendo como se defina la red) los miembros pueden ser anónimos o pseudo-anónimos. Otro inconveniente, es la transparencia de la red, la cual implica poca o ninguna privacidad para las transacciones, además de la gran cantidad de potencia de cálculo necesaria para el mantenimiento del libro mayor. Con tantos nodos y transacciones como parte de la red, este tipo de escala requiere un gran esfuerzo para lograr el consenso. [16]

Las implementaciones como: Bitcoin, Ethereum, Monero, Litecoin, entre otras poseen esta arquitectura. [17]

1.3.2. Privada

Como se mencionó en el punto anterior, buscando evitar problemas como: privacidad de información, escalabilidad del volumen de transacciones, capacidad de respuesta del sistema, entre otras. Se han desarrollado Blockchain que no encajan en la definición de público, uno de ellos la arquitectura privada.

Su principal diferencia como se puede intuir por su nombre es la privacidad de los datos, este cambio que puede parecer simple, sin embargo, impacta en varias características que definen a un Blockchain público.

Para tener acceso a una red privada de Blockchain, debe ser invitado y luego validado por el iniciador de la red o por reglas específicas que fueron establecidas, lo cual elimina cualquier rastro de anonimidad de los nodos participantes.

Debido a que la arquitectura ya no depende necesariamente de los nodos, las opciones de ver o validar las transacciones pueden ser restringidas por condiciones que definió el iniciador o creador de la red. Un ejemplo, es que el autor de la cadena de bloques defina grupos y participantes que puedan verificar las transacciones internamente, esto modificaría las condiciones de consenso para las transacciones y crearía riesgos de seguridad nuevos a los existentes en un sistema público. [15]

La arquitectura privada, es una de las principales opciones para empresas que necesitan utilizar la tecnología Blockchain, pero no quieren hacer publica su información. Las implementaciones como Monax, Multichain, entre otras poseen esta arquitectura. [17]

1.3.3. Consorcio

Los Blockchain de arquitectura de consorcio se asemejan a los Blockchain privados como se puede observar en la tabla 1.1, sin embargo, su principal distinción es su esquema de gobierno e infraestructura.

Para ciertas corporaciones o grupos corporativos el esquema privado puede quedar pequeño para sus necesidades, los Blockchain de arquitectura de consorcio permiten comunicar a departamentos y empresas, a la vez que permiten a los responsables de la toma de decisiones limitar el acceso a los datos y a la validación. [18]

Los implementadores pueden aprovechar las ventajas en el gobierno del sistema, mientras que en las cadenas de bloques privadas una entidad gobierna toda la red, la arquitectura de consorcio permite definir miembros que compartan la autoridad entre ellos.

En consecuencia, la infraestructura está centralizada en el caso de las Blockchain privadas, pero de forma similar a las bases de datos distribuidas comunes, los datos se replican en múltiples nodos que pertenecen al único propietario. Por el contrario, las cadenas de bloques de consorcio se despliegan de forma descentralizada en múltiples hardware gestionados por diferentes propietarios (o empresas). Además, no es necesario que los datos sean homogéneos entre los nodos del consorcio, ya que algunas cadenas de bloques permiten transacciones privadas que conducen a la fragmentación del conocimiento (es decir, las transacciones privadas son compartidas por subconjuntos de participantes). [16]

Las implementaciones como R3 (Bancos), EWF (Energía), Corda, entre otras poseen esta arquitectura. [17]

Propiedad	Público	Privado	Consorcio
Tipo de Gobierno	Protocolo de consenso público	El consenso es administrado por un conjunto de participantes	El consenso es administrado por un solo propietario
Validación de Transacción	Cualquier nodo minero	Una lista de nodos autorizados (o validadores)	
Algoritmo de Consenso	Sin permiso (PoW, PoS, PoET, etc.)	Con permiso (PBFT, Tendermint, PoA, etc.)	
Permiso de lectura de Transacciones	Cualquier nodo	Cualquier nodo (sin permiso) o una lista de nodos predefinidos (con permiso)	

Alteración de transacciones	Modificar transacciones es imposible	Alterar transacciones es posible	
Tasa de transacciones	Baja	Alta	
Escalabilidad de red	Alta	Baja a media (unas pocas docenas / cientos de nodos)	
Infraestructura	Altamente descentralizado	Descentralizada	Distribuida
Ventajas	• Resistencia a la censura • No regulado y transfronterizo • Soporte de activos nativos • Identidades anónimas • Arquitectura de red escalable	• Aplicable a negocios altamente regulados (identidades conocidas, estándares legales, etc.) • Eficiente rendimiento de las transacciones • Transacciones sin comisiones • Las normas de infraestructura son más fáciles de gestionar • Mejor protección contra las perturbaciones externas	
Ejemplos de Tecnologías	Bitcoin, Ethereum, Ripple, etc.	Monax, Multichain, etc.	R3, EWF, Corda, etc.

Tabla 1.1: Comparación de arquitecturas en Blockchain. [16] [17]

1.4. Implementaciones

Cuando se piensa en Blockchain es común imaginar las implementaciones de criptodivisas, en las cuales cualquier persona puede minar o adquirir una moneda para realizar transacciones o como método de inversión. Sin embargo, los sistemas Blockchain puede ser utilizados de diversas maneras, y para fines distintos a las criptodivisas, como se puede observar en la figura 1.6.

Por ser Blockchain una tecnología relativamente nueva, en el futuro probablemente existan nuevas formas de aplicar las cadenas de bloques y áreas que por el momento no se encuentran en la figura 1.6.

Actualmente las principales formas de desarrollar este tipo de tecnología son:

- Blockchain como un Servicio (Baas).
- Blockchain Primero.
- Plataformas de desarrollo.

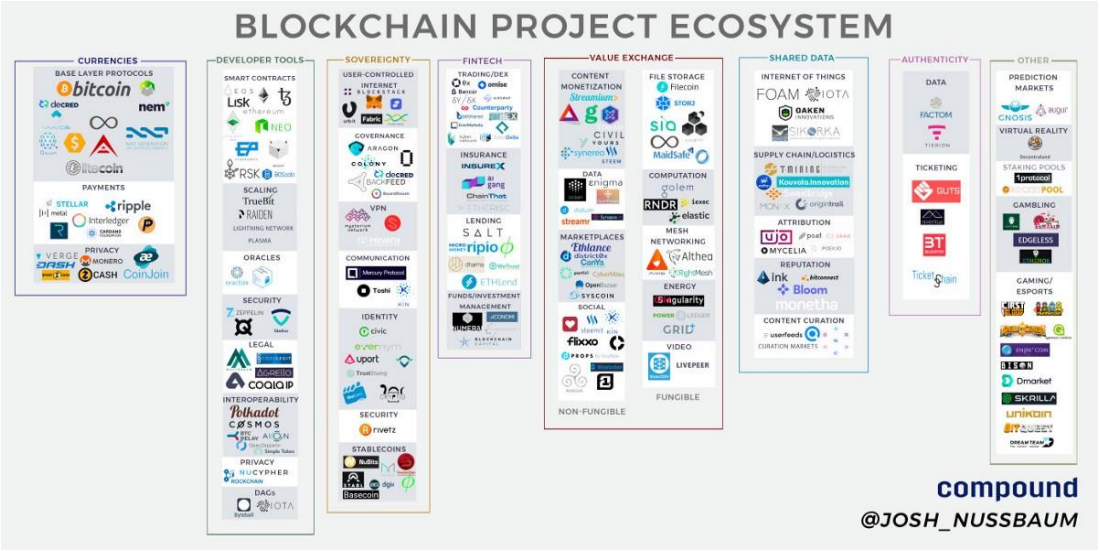

Figura 1.6: Áreas en las cuales, existen aplicaciones de Blockchain. [19]

1.4.1. Blockchain como un Servicio (Baas)

La implementación Blockchain como un Servicio (BaaS), se basa en y funciona de manera similar al concepto de Software como un Servicio (SaaS).

Es una opción que permite a un cliente aprovechar soluciones basadas en la nube para construir, alojar y utilizar sus propias aplicaciones de Blockchain, mientras que el proveedor gestiona todas las tareas y actividades necesarias para mantener la infraestructura ágil y operativa. [20]

Debido a la importancia que ha tomado Blockchain como tecnología para desarrollo de soluciones, las grandes empresas en especial las dedicadas a TI, han desarrollado opciones de BaaS para sus clientes, a continuación, se enuncian algunas de las más importantes:

- Microsoft: fue uno de los primeros proveedores de software en ofrecer BaaS cuando lanzó Azure Blockchain Service en 2015, cuya arquitectura se observa en la figura 1.7, para el año siguiente anunció una colaboración con Blockstack Labs, ConsenSys y una variedad de desarrolladores, en un sistema de identidad de código abierto basado en cadenas de bloques que permite a las personas, productos, aplicaciones y servicios interoperar entre cadenas de bloques, proveedores de cloud y organizaciones. En el año 2017 lanzó Enterprise Smart Contracts, que proporciona a los usuarios el esquema, la lógica, las contrapartes, las fuentes externas, el libro mayor y la vinculación de contratos para crear sus propios servicios de cadena de bloques. [21]
- Corda: es una plataforma de Blockchain con versiones de pago y de código abierto, creada por R3, un consorcio financiero con más de 70

socios que incluyen entidades como BBVA, Bank of America y HSBC. Funciona como un libro mayor especializado en procesar las transacciones financieras, el cuál posee libros de contabilidad interoperables, por lo que las aplicaciones de software pueden comunicarse, intercambiar datos y utilizar los datos intercambiados. [22]

- Fabric: es una implementación plug-and-play diseñada por IBM como base para desarrollar aplicaciones de Blockchain de gran escala con un grado flexible de permisos. Fabric se diseñó para proporcionar un marco con el que las empresas puedan crear sus propias redes de cadenas de bloqueo individuales que puedan escalar rápidamente a más de 1.000 transacciones por segundo. [23]
- Deloitte: Una de las consultoras Big Four Deloitte, tiene una solución para empresas llamada Rubix Core. Rubix es una infraestructura de cadena de bloques de pila completa compatible con Ethereum, que se centra en la interoperabilidad, escalabilidad, rendimiento y seguridad. Rubix está específicamente diseñado para permitir a los desarrolladores crear e implementar aplicaciones descentralizadas en un entorno de cadena de bloques que puede personalizarse fácilmente para los requisitos únicos de la industria y el negocio. [24]

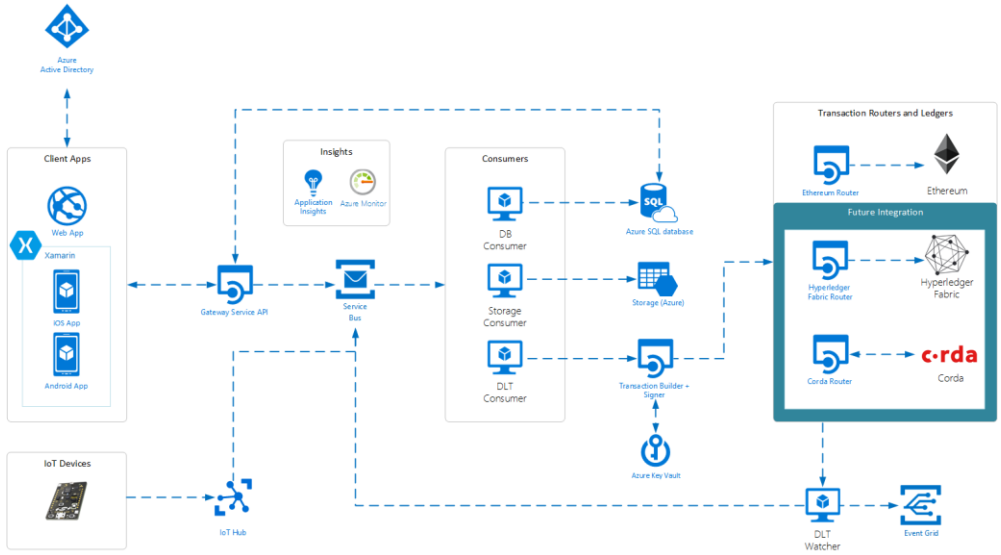

Figura 1.7: Arquitectura de Azure Blockchain. [25]

1.4.2. Blockchain Primero

En esta forma de implementación, las cadenas de bloques poseen mayor relevancia que la manera anterior. Antes Blockchain se ve como un servicio ofertado por un proveedor a un cliente, mientras que en Blockchain primero las aplicaciones se diseñan de forma tal, que toda la lógica de negocio y el almacenamiento de datos ocurre principalmente en una base, cuya tecnología es de tipo cadena de bloques y actúa como autoridad central.

Todos los demás sistemas y participantes tendrán acceso a esta lógica y datos, los cuales están de acuerdo en que sólo las actualizaciones en el Blockchain de base son válidas para su procesamiento posterior. La desventaja de este tipo de implementación, es que la tecnología de cadena de bloques está en constante desarrollo y una falla en la autoridad central, afecta a todas las aplicaciones y participantes que dependen de la misma. Una de las implementaciones más conocidas es OpenBazaar. [26]

OpenBazaar es un proyecto de código abierto, creado con el fin de formar una red descentralizada para el comercio entre pares en línea utilizando criptodivisas, el cual no tiene tarifas ni restricciones como se puede observar en la figura 1.8. Para utilizar este proyecto, se debe descargar el programa cliente de OpenBazaar en el dispositivo, el cual permite crear una lista de productos disponibles para la venta, esta lista es accesible mediante la red distribuida punto a punto (P2P) a todos los usuarios que poseen el programa de cliente de OpenBazaar, los cuales al buscar algún producto que se encuentre en alguna de las listas, obtendrán todas las listas que poseen ese producto para la venta. Cuando un usuario compre un artículo ofertado, el programa crea un contrato entre las partes con sus firmas digitales, el comprador deposita el pago en una cuenta garante, y si el comprador recibe el producto y está conforme con el artículo, se libera el pago. [27]

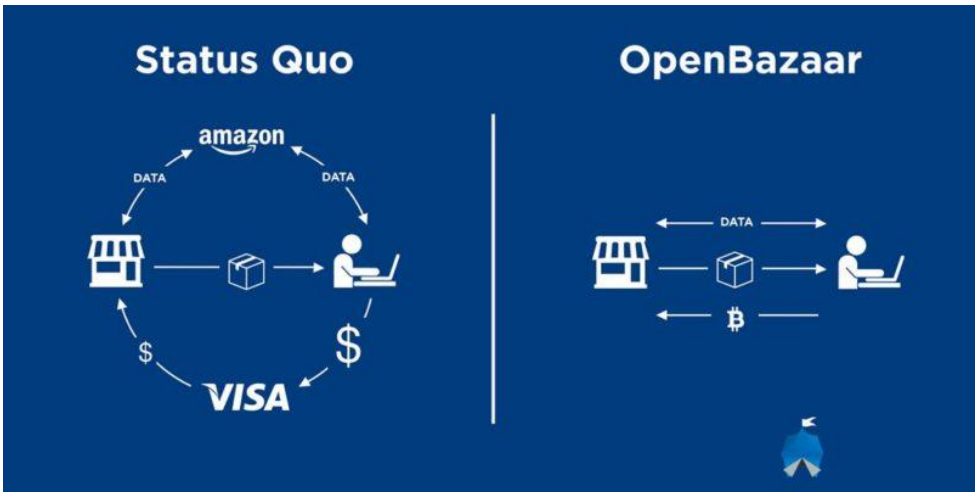

Figura 1.8: Esquema de OpenBazaar. [28]

1.4.3. Plataformas de desarrollo

Son plataformas creadas específicamente para desarrollar aplicaciones Blockchain, algunas de las cuales pueden estar o no en la nube (BaaS). Poseen la ventaja de facilitar la adopción de la tecnología Blockchain en las empresas.

Entre las opciones de Plataformas de desarrollo más conocidas están:

- BlockApps
- Blockstream
- Eris

CAPÍTULO 2: Contratos inteligentes

2.1. Antecedentes

En la cultura popular, un contrato es un “acuerdo legal que no se puede romper”, pero a lo largo del tiempo se han definido distintos tipos de contratos, los cuales han ido evolucionando y adaptando a las necesidades de las partes. Las primeras transacciones que poseían un sustento legal y de las que se tiene registro, aparecen en la antigua Babilonia, en las cuáles se registraba la formación de familias mediante un contrato, en el cual el novio concedía al suegro un regalo nupcial y el suegro concedía a la hija una dote. [29]

Más tarde los romanos definieron diferentes tipos de contratos en el derecho romano, estos se pueden clasificar en: Nominado e Innominado, ambos casos podían ser orales o escritos. [30]

Dejando de lado la antigüedad, los contratos hoy en día pueden ser suscritos por medios diferentes a la firma convencional o a sellos, un claro ejemplo de esto es que desde hace más de 15 años, la mayor parte de países han desarrollado legislatura relacionada a contratos firmados mediante firma electrónica o digital, en el caso de la Argentina la Ley 25.506 promulgada en el 2001, tiene por objeto reconocer *“el empleo de la firma electrónica y de la firma digital y su eficacia jurídica en las condiciones que establece la presente ley.”* [31]

Otro ejemplo, son los contratos suscritos por voz, los cuales son comunes por parte de empresas aseguradoras, telefónicas, entre otras. Existen desarrollos que buscan apegar estos contratos a las leyes locales, este es el caso de Biometric Vox, el cual utiliza una plataforma de servicios para poder firmar un contrato mediante la voz, además afirma cumplir el reglamento europeo “eIDAS”. [32]

Los nuevos tipos de contratos mencionados se han desarrollado con el propósito de flexibilizar los medios para firmar estos, además de reducir costos. Lamentablemente no se puede evitar situaciones como la necesidad de un tercero que avale la transacción, el cual debe ser de confianza entre las partes, esto expone a los contratos a riesgos en su seguridad, como un único punto de falla. Afortunadamente desde los años 90 se comenzó a hablar de contratos inteligentes, los cuales persiguen eliminar o cuanto menos mitigar algunos riesgos actuales, además de reducir costos.

El término contratos inteligentes (Smart contracts) se acuñó en el año 1993, cuando Nick Szabo habló por primera vez de la idea de un tipo de contrato criptográfico en el que la verificación y las obligaciones contractuales se ejecutan a través de un código informático autoimpuesto. [33]

2.2. Definición

No existe una definición formal aceptada universalmente sobre los contratos inteligentes, pero se pueden describir como programas informáticos que pueden ejecutarse de manera consistente por una red de nodos que desconfían mutuamente, sin el arbitraje de una autoridad confiable. Debido a su resistencia a la manipulación, los contratos inteligentes son atractivos en muchos escenarios, especialmente en aquellos que requieren transferencias de dinero para respetar ciertas reglas acordadas (como los servicios financieros y los juegos). [34]

Para el momento en que Nick Szabo propuso su idea de contratos inteligentes no existía la tecnología necesaria para hacerla realidad, fue hasta la invención del Blockchain, que se pudo convertir los contratos en código informático y codificar en cadenas de bloques, lo cual también permite su descentralización. Este aspecto es simplifica el proceso y elimina la necesidad de un tercero, debido a que el ejecutor es ahora el código, y no es imprescindible un abogado para asegurarse de que el contrato se ejecuta correctamente. [35]

Debido a que los Smart Contracts poseen las características:

- Auto verificable.
- Auto ejecutable.
- Resistente a las manipulaciones.

Son capaces de realizar un seguimiento del rendimiento en tiempo real y pueden suponer un enorme ahorro de costo. Además, el cumplimiento y el control se realizan sobre la marcha. [36]

2.3. Funcionamiento

Como se ha mencionado los contratos inteligentes permiten realizar intercambios sin la necesidad de un intermediario. En la figura 2.1 se ilustra los pasos que son necesarios desde el inicio hasta el final para el funcionamiento de un Smart Contract, los cuales son comunes inclusive para un contrato tradicional:

1. **Términos del Contrato. -** Para el primer paso las partes definen las condiciones que se cumplirán y los detalles del contrato que permitirán su ejecución en el tercer paso, entre estas condiciones están: eventos que realizarán las partes, fecha de cumplimiento, divisa a utilizar, entre otras. En el caso que las partes utilicen un contrato pre-definido este paso será obviado.
2. **Eventos. -** Aquí suceden las condiciones de cumplimiento definidas en el paso anterior, es decir, las transacciones que las partes acordaron para el cumplimiento del contrato. Puede ser la transferencia de un bien, la prestación de un servicio, o inclusive un evento externo a las partes, unos

ejemplos de esto último fueron conocidos casos, en que mediante la plataforma Augur basada en Ethereum, se crearon listas de la muerte para predecir la muerte de famosos. [37]

3. **Ejecución. -** A partir del cumplimiento o no de los eventos en el segundo paso, ocurre la ejecución del contrato de acuerdo con lo establecido, y las partes reciben lo que firmaron en el contrato.
4. **Asentamiento. -** En este paso se envía el contrato a la cadena de bloques y se registra en ella, en el caso que se utilice criptodivisas o bienes digitales, se asienta en el libro mayor el cambio en las cuentas de las partes. En el caso que el medio de pago sea físico y no digital, las cuentas de las partes se modificarán de acuerdo a lo establecido.

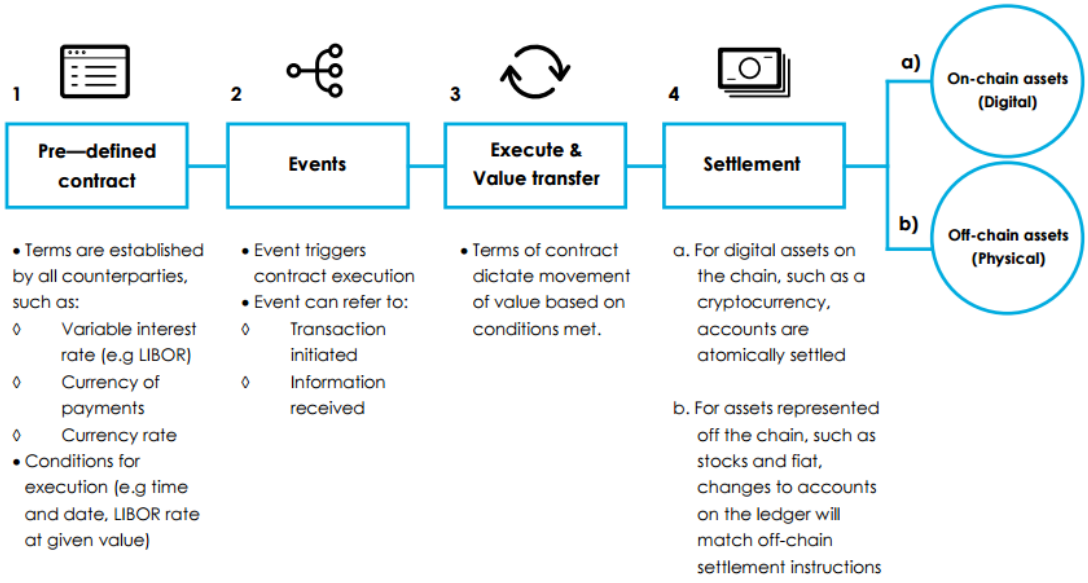

Figura 2.1: Ejemplo del funcionamiento de un Smart Contract. [38]

Debido a que los Smart Contract pueden requerir información de agentes externos, se pueden clasificar los contratos inteligentes en dos tipos:

- Deterministas
- No Deterministas

Un contrato inteligente determinista es aquel que cuando se ejecuta, no requiere ninguna información de una parte externa (de fuera de la cadena de bloques). Un contrato inteligente no determinista por el contrario depende de la información (llamada oráculos o fuentes de datos) de una parte externa. Por ejemplo: un Smart Contract que requiera información de la muerte de un famoso, la cual no está disponible en la cadena de bloques, se considera como determinista. [1]

2.4. Desarrollos Actuales

A pesar de que el desarrollo de Contratos Inteligentes en la práctica es reciente, existen varias opciones que pueden ser utilizadas, las principales son enunciadas aquí y una de ellas se utilizada en el siguiente capítulo en la Prueba de Concepto, además de ello existen otras opciones que no son enunciadas pero permiten el desarrollo de Smart Contracts como: EOS, Hiperledger Fabric, Stratis, Waves, Nem, Aion Aeternity, Zen, Rootstock, RChain, Qtum, Ark, Neblio, DFINITY, BOSCoin, Agoras Tauchain, Burst, iOlite, ByteBall, XTRABYTES, PolkaDot, Radix, Exonum, Universa, Urbit, Soil, Expanse, entre otros.

2.4.1. Bitcoin

Es la plataforma de Blockchain más conocida del mundo, utiliza una Cadena de Bloques pública para registrar el historial completo de las transacciones en la red. Los nodos de la red Bitcoin utilizan un PoW basado en hash como algoritmo de consenso para establecer cómo añadir un nuevo bloque de transacciones a la red Bitcoin, estos nodos poseen diferentes opciones de software para el mantenimiento de la red y la aprobación de transacciones.

2.4.1.1. Transacciones

Al igual que cualquier sistema basado en Blockchain, el motivo fundamental de la existencia de la red, es la realización de transacciones, las cuales buscan la transferencia de divisas y no la creación de Contratos Inteligentes.

Como se puede observar en la figura 2.2, las transacciones en la red Bitcoin contienen esencialmente la siguiente información: [39]

- ID: Identificador de transacción único.
- Entrada: Las direcciones que identifican las fuentes de los bitcoins a transferir. Éstos son generalmente la salida de una transacción anterior y se utilizan para verificar el remitente y verificar el saldo disponible.
- Monto: El número de bitcoins a transferir.
- Salida: La dirección del receptor.

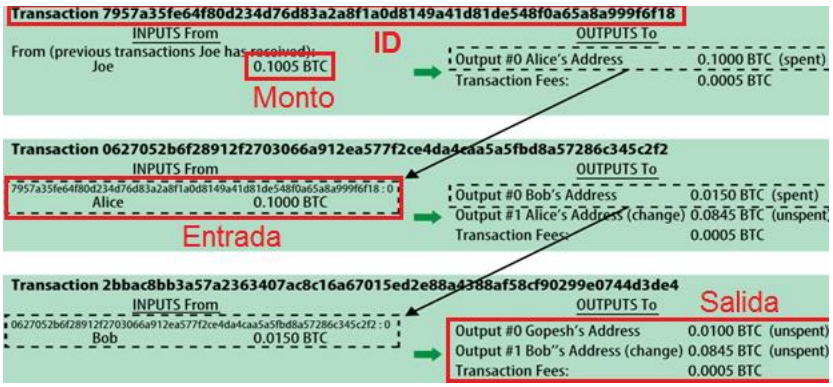

Figura 2.2: Transacción en la red Bitcoin. [39]

Las salidas de una operación pueden utilizarse como entradas para otra operación. Esto crea una cadena de propiedad a medida que el valor del bitcoin se traslada de una dirección a otra. [39]

2.4.1.2. Contratos Inteligentes

Como se ha mencionado anteriormente, Bitcoin es una red que no fue diseñada para la realización de Smart Contracts, sin embargo, en la actualidad la popularidad de esta plataforma ha propiciado el desarrollo de nuevas funcionalidades a la versión original de Satoshi Nakamoto, haciendo posible el desarrollo de Smart Contracts en esta cadena de bloques.

La mayor parte de las condiciones existentes para el desarrollo de los Contratos Inteligentes en la red, se encuentran en las transacciones. Cada transacción puede tener un tiempo de bloqueo asociado, esto permite que la transacción esté pendiente y sea reemplazable hasta un momento futuro acordado. Además de ello, cada entrada y salida de las transacciones tiene una pequeña función asociada a ella llamada script, existen implementaciones que se basan en estos scripts para desarrollar Smart Contract, un ejemplo de ello es “Particl”. [40]

La plataforma de Particl utiliza el opcode de Bitcoin *“CHECKLOCKTIMEVERIFY”*, que fue introducida por Peter Todd en el Bitcoin Improvement Proposal (BIP) 65. Esta característica permite escribir scripts que evitan que se gasten fondos en una cartera con múltiples firmas hasta que se implemente un determinado patrón de firmas o hasta que transcurra un cierto tiempo. La desventaja de desarrollar scripts complejos en las entradas y salidas de las transacciones, es que no todos los nodos de Bitcoin pueden manejar esta clase de scripts, debido a que, existen versiones antiguas de software utilizadas por los nodos que no poseen compatibilidad con estas características. [40]

Otro ejemplo de implementación de Smart Contract sobre Bitcoin, es Counterparty, que será estudiada más adelante, esta solución utiliza la red de Bitcoin para transmitir Contratos Inteligentes.

2.4.2. NXT

Nxt es una plataforma pública basada en Blockchain, la cual tiene diversos usos, no se encuentra focalizada en Smart Contracts, pero posee una selección de plantillas para contratos inteligentes, a los cuales llama Smart Transactions (Transacciones Inteligentes).

Entre las principales razones para utilizarlo están:

- Nxt es un sistema autosuficiente. No depende de una cadena de bloques implementada y mantenida por un tercero como, por ejemplo: Bitcoin. Nxt es un sistema completo y autónomo en sí mismo. [41]
- Como se ha mencionado, Nxt es una plataforma de código abierto.

- Los desarrolladores de Nxt pueden crear rápida y fácilmente nuevas características mientras mantienen un sistema coherente, sin necesidad de consultar con un proveedor externo de Blokchain. [41]
- Existen aplicaciones para Blockchain que ya se encuentran desarrolladas en este sistema como los Contratos Inteligentes.
- Posee arquitectura modular.

La última característica puede ser vista tanto una ventaja como una desventaja, dependiendo de la aplicación para la cual se utilice Nxt. Para aprovechar al máximo la versatilidad de Nxt, sus desarrolladores han creado un sistema de plug-in que permite a los usuarios interesados construir aplicaciones y compartirlas con otros usuarios de Nxt.

En el caso de los Contratos inteligentes, la tecnología modular restringe el desarrollo de nuevas clases de contratos y permite el uso únicamente de aquellas funcionalidades ya desarrolladas, las cuales se conocen como Transacciones Inteligentes.

Las Transacciones inteligentes no requieren ningún procesamiento de scripts o de entrada/salida de transacciones por parte de los nodos de la red; los scripts ya están incrustados, el código que se ejecuta es el software real que se ejecuta en el servidor del nodo. [42]

Cuando un usuario quiere expresar su opinión en una encuesta, comprar un artículo en el mercado o vender acciones, la transacción que el usuario presenta contiene sólo los parámetros necesarios para la transacción, y el ID de la funcionalidad que desea utilizar, manteniendo el consenso de la mayoría de los nodos como prueba absoluta de que la salida, guardada en el siguiente bloque, es el resultado genuino de esa transacción.

Otra diferencia es que una aplicación informática tiene diez veces más potencia y posibilidades que un script interpretado en una máquina virtual, como es el caso de los Smart Contracts utilizados en Ethereum. [42]

2.4.3. Counterparty

Counterparty es una plataforma de código abierto construida sobre bitcoin, es decir, no posee Blockchain propio, Counterpaty pretende en palabras del desarrollador líder Adam Krellenstein, "Democratizar las finanzas de la misma manera que Internet democratizó la creación y el intercambio de información". Esta plataforma permite el uso de cualquier Smart Contract escrito en Solidity o Serpent, lenguajes de programación utilizados en los contratos inteligentes en Ethereum.

Debido a que los datos de los Smart Contracts viajan sobre el Blockchain de Bitcoin, los nodos de Bitcoin reciben e ignoran los datos de Counterparty y tan solo los nodos de Counterparty pueden reconocer, interpretar y ejecutar los contratos inteligentes.

2.4.3.1. Smart Contract

Counterparty describe en su página los pasos para la implementación de un Smart Contract en su plataforma, los cuales son los siguientes:

- Escribir el código del Contrato Inteligente (usando Solidity o Serpent) y compilar en una forma más compacta (bytecode). [43]
- Counterparty creará y emitirá una transacción de publicación para integrar el código del contrato en el Blockchain de Bitcoin.
- Una vez publicado, el Smart Contract se aloja en una dirección.
- Mediante Counterparty se puede crear y transmitir una operación de ejecución para llamar una función (transacción) en el código del contrato inteligente. [43]
- Se ejecuta la transacción y es confirmada por un minero de Bitcoin, cada nodo Counterparty en ejecución recibirá esta solicitud y ejecutará ese método. [43]
- Se modifica el estado del contrato almacenado en la base de datos de Counterparty, a medida que se ejecuta el Smart Contract. [43]

Es importante destacar dos características: Es posible asociar activos de Counterparty al Smart Contract, que los almacenará y podrá utilizarlos en futuras ejecuciones de llamadas. Y la ejecución de un Smart Contract, generalmente se ejecuta tan rápido como el nodo puede procesarlo. [43]

2.4.4. Stellar

Es un sistema distribuido complejo orientado a las finanzas, que permite el desarrollo de billeteras móviles, herramientas bancarias, como se puede observar en la figura 2.3, y entre otras cosas, la elaboración de Smart Contracts, a través de los Stellar Smart Contract (SSC), nombre con el que se conoce a los contratos inteligentes en Stellar. Un SSC se puede definir como una composición de transacciones que se conectan y ejecutan utilizando varias restricciones.

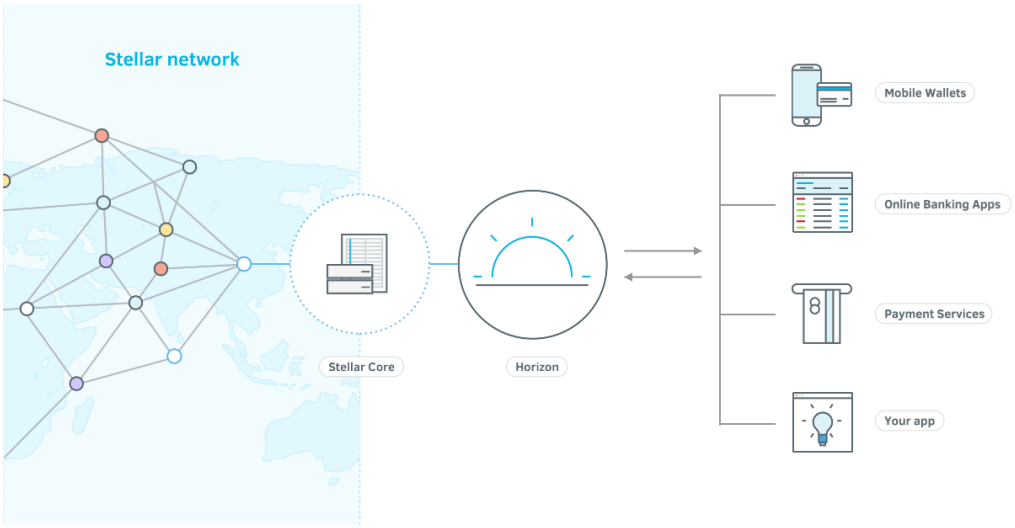

Figura 2.3: Red Stellar.

El protocolo central de la red Stellar continua en desarrollo, y actualmente no existe ninguna garantía con respecto a la compatibilidad tanto hacia atrás como hacia delante, para los Smart Contracts de larga duración.

2.4.4.1. Arquitectura de Red

La arquitectura de la red Stellar, mostrada en la figura 2.3, es aquella sobre la que se desarrollan todas las soluciones financieras que permite, entre ellas los Contratos Inteligentes, los siguientes elementos constituyen las piezas fundamentales de la red:

- **API Horizon:** La mayoría de las aplicaciones interactúan con la red Stellar a través de Horizon. Horizon ofrece una forma sencilla de enviar transacciones, cuentas de cheques y suscribirse a eventos, debido a que utiliza el protocolo HTTP, Horizon permite la interacciónmediante el uso de navegadores web, así como, herramientas sencillas de línea de comandos como: cURL o inclusive alguno de los SDK (kit de desarrollo de software) de Stellar. Actualmente Stellar.org mantiene SDKs basados en JavaScript y Java, además existen SDKs mantenidos por la comunidad de desarrollo de Stellar para Ruby, Python y C#.
- **Stellar Core / Núcleo de Stellar:** Es la red de Backbone de Stellar, donde se conectan todos los servidores Horizon. El software Stellar Core realiza el arduo trabajo de validar y acordar con otras instancias de Core el estado de cada transacción, a través del Protocolo de Consenso de Stellar (SCP). La red Stellar es en sí misma una colección de Núcleos Stellar conectados, los cuales se encuentran administrados por varios individuos y entidades alrededor del mundo. En algunos casos poseen un servidor Horizon con el que puede comunicarse, mientras que otros sólo existen para añadir fiabilidad a toda la red.

Como se ha mencionado, la red Stellar es una colección mundial de núcleos estelares, cada uno mantenido por diferentes personas y organizaciones. La naturaleza distribuida de la red la hace fiable y segura.

Todos estos núcleos estelares, acuerdan en conjunto las transacciones. Cada transacción en la red cuesta una pequeña cuota: 0.00001 XLM. Esta tarifa ayuda a evitar que los malos actores envíen spam a la red.

2.4.4.2. Restricciones

Como se mencionó, los SSC permiten restricciones que ayudan a mejorar el desarrollo y el uso de los mismos, las restricciones posibles en los SSC hasta el momento son:

- **Multifirma:** Este concepto desarrollado en Stellar, permite que un SSC requiera de múltiples partes para firmar transacciones derivadas de una cuenta.

- **Batch/Atomicidad:** Este concepto permite que un SSC dependa de más de una transacción para generar su accionar. La atomicidad es la garantía de que, dada una serie de operaciones, al ser sometidas a la red si falla una operación, falla toda operación de la transacción.
- **Secuencia:** Este concepto es una alternativa al concepto de Atomicidad, utilizando números de secuencia en la manipulación de transacciones, se puede garantizar que transacciones específicas no tienen éxito si se presenta una transacción alternativa.
- **Límite de tiempo:** Este concepto introduce limitaciones en el período de tiempo durante el cual una transacción es válida.

2.4.4.3. Moneda

Stellar utiliza, al igual que todas las soluciones vistas en el capítulo, su propia moneda para las transacciones en la red, el lumen (XML). En 2014 la red Stellar lanzó 100 mil millones de “Stellars”, el nombre original de la moneda de la red, el cual se cambió al siguiente año por “Lumen”.

Los lúmenes cumplen dos funciones importantes en la red:

- **Evitar transacciones spam:** Como se ha mencionado, los lúmenes son necesarios para las tarifas de transacción, además las cuentas en la red Stellar tienen un saldo mínimo (0.5 XML), esto como medida de seguridad, al mitigar los ataques DoS que intentan generar un gran número de transacciones o consumir grandes cantidades de espacio en el libro mayor del Blockchain.
- **Facilitar las transacciones en varias divisas:** Los lúmenes actúan como puente, facilitando las operaciones entre pares de divisas entre las cuales su intercambio resulta complejo. Esta función es posible cuando hay un mercado líquido entre el lumen y cada moneda involucrada.

2.4.4.4. Seguridad

Para garantizar la confiabilidad en las transacciones realizadas en la red, Stellar utiliza criptografía asimétrica como medida de seguridad. Para ello se utiliza una clave pública la cual es segura para compartir con otros usuarios que necesitan identificar una cuenta y mantener el no repudio de una transacción, así como una semilla secreta, la cual es privada y se utiliza para determinar unívocamente el propietario de una cuenta.

En Stellar se utiliza una semilla para dar acceso completo a una cuenta, debido a que permite generar tanto la clave pública como la privada, al ser un elemento de tal importancia, la semilla debe mantenerse en secreto. En la actualidad, el proceso que se utiliza para crear las claves de una cuenta invoca la función “ed25519”, la cual utiliza el esquema de firma EdDSA (Edwards-curve Digital Signature Algorithm) con SHA-512 y la criptografía de curva elíptica de 128 bits “Curve25519”.

2.4.5. Monax

Esta solución se diferencia del resto, al ser desarrollada por una compañía que posee su propia infraestructura y centrarse en el sector legal, Monax es pionera en la tecnología de Blockchain y fue la primera empresa en comercializar un diseño de Cadena de Bloques compatible con el desarrollo de contratos inteligentes, Burrow. Lanzado en 2014, Burrow proporciona un cliente de Blockchain modular con un intérprete de Smart Contracts autorizado, construido en parte según las especificaciones de la máquina virtual Ethereum (EVM). [44]

Burrow permite a los usuarios crear Blockchain privados y definir políticas de autorización para acceder a ellas. Su protocolo de consenso está organizado en rondas, en las que un participante propone un nuevo bloque de transacciones, y los demás votan por el bloque. Cuando un bloque no se aprueba, el protocolo pasa al siguiente, donde otro participante se encargará de proponer los bloques. Un bloque se confirma cuando es aprobado por al menos dos tercios del total de los votos.

Pero Monax no se quedó allí, a pesar de ser uno de los desarrolladores y miembros que da mantenimiento a Burrow, en el año 2018 lanzó su plataforma Monax en versión beta, cuya pantalla principal se observa en la figura 2.4. Esta plataforma es descrita por la compañía como un “espacio de trabajo colaborativo para empresas, profesionales del derecho y de la tecnología, con plantillas de contratos inteligentes listas para el mercado y disponibles para uso individual o comercial”. [45]

Como se ha mencionado Monax está centrada en el ámbito legal, y se espera que pronto (en un año o dos máximo) salga al mercado con su propia plataforma de plantillas de contratos inteligentes dirigida a abogados que quieren aprovechar la tecnología, lo cual sería una buena noticia para las firmas de abogados que desean explorar esta área.

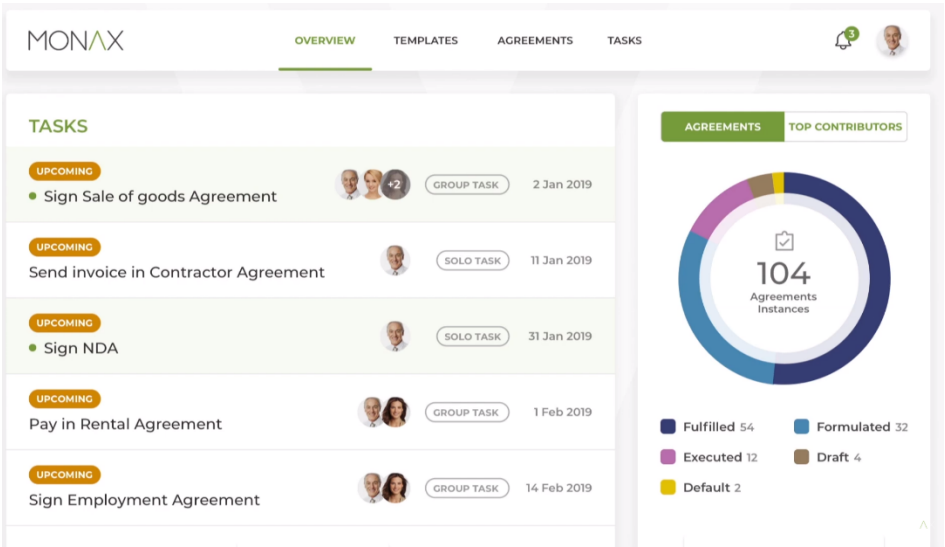

Figura 2.4: Pantalla de inicio de Monax Plataform. [45]

La Plataforma Monax consta de dos elementos:

- **Monax Deal Space:** permite a los usuarios delegar y coordinar tareas de equipo, automatizar tareas repetitivas, hacer un seguimiento del estado de desarrollo de las obligaciones contractuales, realizar auditorías de las obligaciones legales y acceder a productos legales de calidad que se adapten a una variedad de necesidades. [45]
- **Monax Legal Products Studio:** permite a los usuarios crear, probar y comercializar sus productos de trabajo en una comunidad de ideas afines. [45]

2.4.6. Lisk

A pesar de estar considerada como una posible solución para el desarrollo de Contratos Inteligentes, es necesario aclarar que Lisk no es una plataforma dedicada a Smart Contracts.

Lisk es una plataforma de Blockchain pública, que posee su propia criptomoneda y que busca simplificar la integración e implementación de otras cadenas de bloques generadas de forma personalizada para un usuario, el cual puede ser un individuo, una pyme o inclusive una entidad financiera como un banco. Lisk ejecuta cada una de las aplicaciones como se muestra en la figura 2.5, en una cadena lateral (Sidechain) completamente separada y aislada, en el caso que un sidechain falle, la responsabilidad recae directamente en el desarrollador que ejecuta el sidechain. [46]

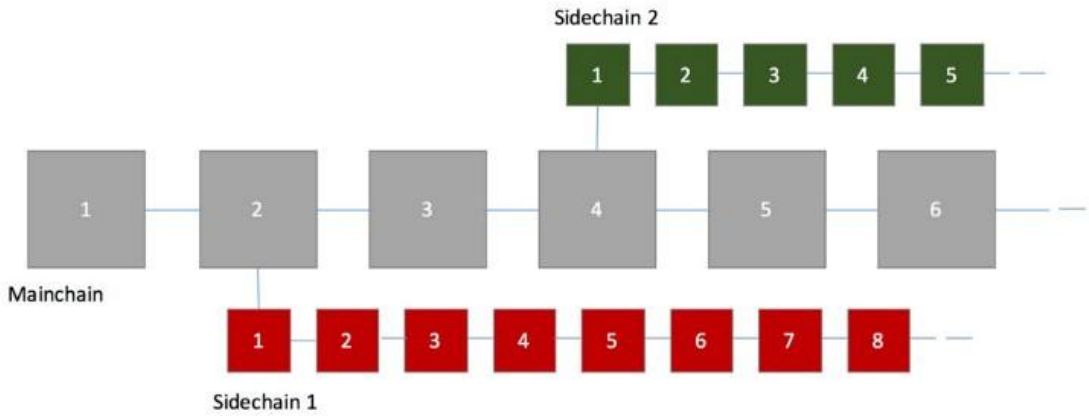

Figura 2.5: Arquitectura de ejecución de Lisk. [46]

2.4.6.1. Características

Al igual que el resto de soluciones para Smart Contracts, Lisk pretende en el futuro convertirse en el estándar del sector, para ello busca diferenciarse de las demás plataformas mediante cualidades que permitan su uso a gran escala, entre las cuales están:

- **Escalabilidad:** Como la distribución de las aplicaciones se realiza en cadenas laterales (Sidechain), Lisk no tiene problemas con la escalabilidad, por lo tanto, la velocidad de su funcionamiento es mucho mayor, y la probabilidad de cualquier fallo o de que se produzcan es mínima. [47]
- **Tolerancia a fallos:** Como se ha mencionado, la plataforma Lisk utiliza Sidechain la cual disminuye el impacto de un fallo en una de las aplicaciones implementadas, al no afectar directamente a la cadena principal (Main Chain).
- **Facilidad de uso:** La plataforma permite escribir aplicaciones de Blockchain en Javascript, lo cual elimina la necesidad de aprender un lenguaje nuevo para los desarrolladores de las aplicaciones. [47]
- **DPoS (Prueba de Aceptación Delegada):** Este sistema es más rápido y consume mucha menos energía que la Prueba de Trabajo (PoW). A continuación, se explica con mayor detalle que es una DPoS.

2.4.6.2. DPoS

La prueba de participación delegada (DPoS) utiliza el voto en tiempo real combinado con un sistema social de reputación para lograr el consenso. Este sistema puede considerarse menos centralizado en comparación que a otros como el PoW, debido a que es más inclusivo con las entidades de la red. [48]

Cada poseedor de una criptomoneda puede ejercer cierto grado de influencia sobre lo que sucede en la red, mediante el voto de los delegados activos, como se muestra en la figura 2.6. Es importante mencionar, que los delegados se eligen teniendo en cuenta la capacidad que poseen para desarrollar su rol, el cuál corresponde al mantenimiento de la red funcionando sin problemas y de forma segura.

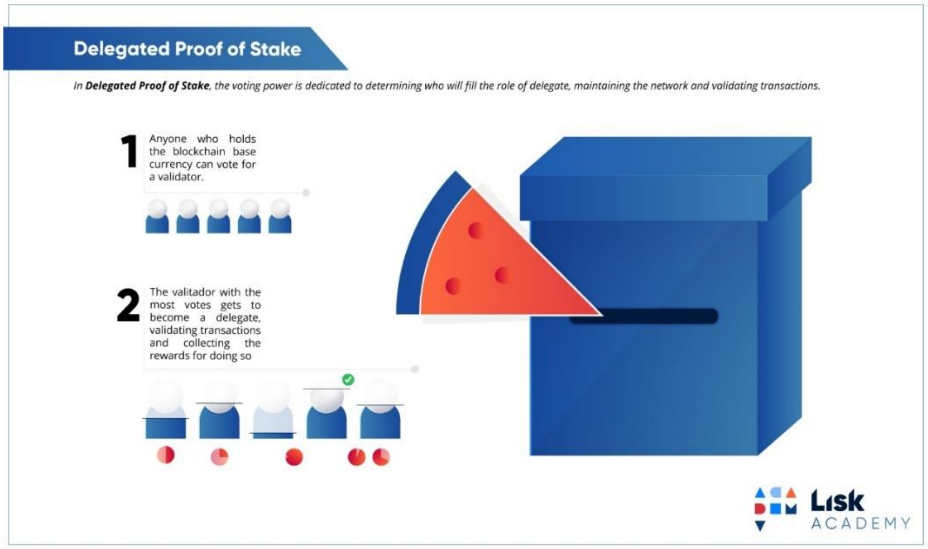

Figura 2.6: Elección de validador para DPoS. [48]

En algunas versiones de DPoS, el delegado debe demostrar su compromiso depositando sus fondos en una cuenta de seguridad (que se confisca en caso de comportamiento malicioso). Esta versión de DPoS se denomina a menudo prueba de participación basada en el depósito. [48]

Los delegados no tienen el poder de cambiar los detalles de las transacciones. Sin embargo, como son validadores, teóricamente podrían excluir ciertas transacciones en un bloque, debido al diseño del proceso de creación de un bloque, la omisión de transacciones tiene poco efecto para las mismas, puesto que el siguiente bloque creado incluirá estas transacciones, dando al siguiente delegado las cuotas asociadas con su validación.

En el caso de Lisk, emplea un sistema de recompensa inflacionaria. Por cada bloque generado se crean nuevos tokens de LSK para recompensar a los que están forjando. Durante el primer año, las recompensas de forja se fijaron en 5 LSK por bloque, reduciéndose la recompensa en 1 LSK cada 3.000.000 de bloques validados, concluyendo en 1 LSK después de aproximadamente 5 años. Suponiendo un aumento en el precio de las fichas en el tiempo, en el año 1 recibir 5 fichas de LSK que valían $1 cada una significaba un retorno de $5, mientras que recibir 1 ficha de LSK en el año 5 que valía $100 significa un retorno considerablemente más alto. [48]

Mientras que, en un sistema de prueba de trabajo, como el empleado por Bitcoin, la validación de bloques se conoce como "minería", en el caso de la prueba delegada de participación este proceso se denomina "forja". [48]

2.4.7. Cardano

Es una plataforma desarrollada en Blockchain por la empresa IOHK, Cardano es considerada como una de las grandes apuestas del futuro, se encuentra en fase de desarrollo, a pesar de esto existen demostraciones de la eficacia de los Smart Contracts desarrollados en ella. Una característica importante, es que el equipo que se encuentra en desarrollo de esta plataforma está integrado por ingenieros expertos e investigadores, muchos de los cuales tienen experiencia en otra de las plataformas aquí descritas: Ethereum, esta es una de las razones por las que se considera a Cardano no como competidor de Ethereum, sino como una evolución. [49]

Un claro ejemplo de la importancia del equipo que posee Cardano para su desarrollo, es el profesor Aggelos Kiayias uno de los criptógrafos más importantes del momento, el cual creó el algoritmo Ouroborus que según Cardano, es la única PoS que ha demostrado matemáticamente ser completamente segura, además de buscar aumentar la escalabilidad de la red.

Su arquitectura se basa en capas y hasta el momento el proyecto es de código abierto.

2.4.7.1. Moneda

El token utilizado para las transacciones en la cadena de bloques de Cardano se llama ADA, para poder utilizar esta moneda virtual es necesario descargar la cartera de criptomonedas “Daedelus”. Esta cartera, al igual que Cardano, se encuentra en continuo desarrollo, sin embargo, posee una serie de características perfeccionadas, entre las cuales están: [49]

- Claves privadas cifradas y contraseñas de gasto, que ofrecen protección contra amenazas como el malware.
- Las billeteras se pueden exportar a certificados en papel.
- Monitoreo configurable del nivel de aseguramiento de transacciones, permitiendo a los usuarios estar seguros cuando las transacciones se vuelven irreversibles.

Como se ha mencionado, Daedelus se encuentra en constante evolución, por ello existen características propuestas que aún están en progreso, por ejemplo: Incluir soporte para Ethereum Classic y Bitcoin, Desarrollar una versión de la cartera para Android e IOS, entre otros. [49]

2.4.7.2. Lenguaje de Programación

Otra importante innovación de Cardano, es la posibilidad de usar más de un lenguaje de programación (Plutus y Marlowe), para desarrollar contratos inteligentes. [50]

- **Plutus:** Está basado en el lenguaje Haskell, y es de tipo funcional. Para animar a los desarrolladores a desarrollar contratos inteligentes utilizando Plutus, IOHK proporciona un entorno fácil de usar llamado Plutus Playground. El entorno de pruebas pone los contratos inteligentes de Plutus en un emulador de cadena de bloques, por lo tanto, los contratos inteligentes que se están probando están listos para su despliegue en la propia cadena de bloques Cardano.
- **Marlowe:** Al contrario de Plutus, esta herramienta se encuentra enfocada para los “no programadores”, debido a que proporciona una forma sencilla de generar código automáticamente y crear productos de software sin necesidad de grandes conocimientos en programación. Esta herramienta es la respuesta a los futuros usuarios como los profesionales de la industria financiera, ya que no necesitan un lenguaje de programación para implementar contratos financieros en la cadena de bloques. Marlowe es una herramienta fácil de usar y viene con su propio entorno de pruebas llamado Meadow. En la figura 2.7, se observa un Smart Contract desarrollado en Marlowe, donde un usuario deposita 100 ADA y un segundo usuario deposita 20 ADA.

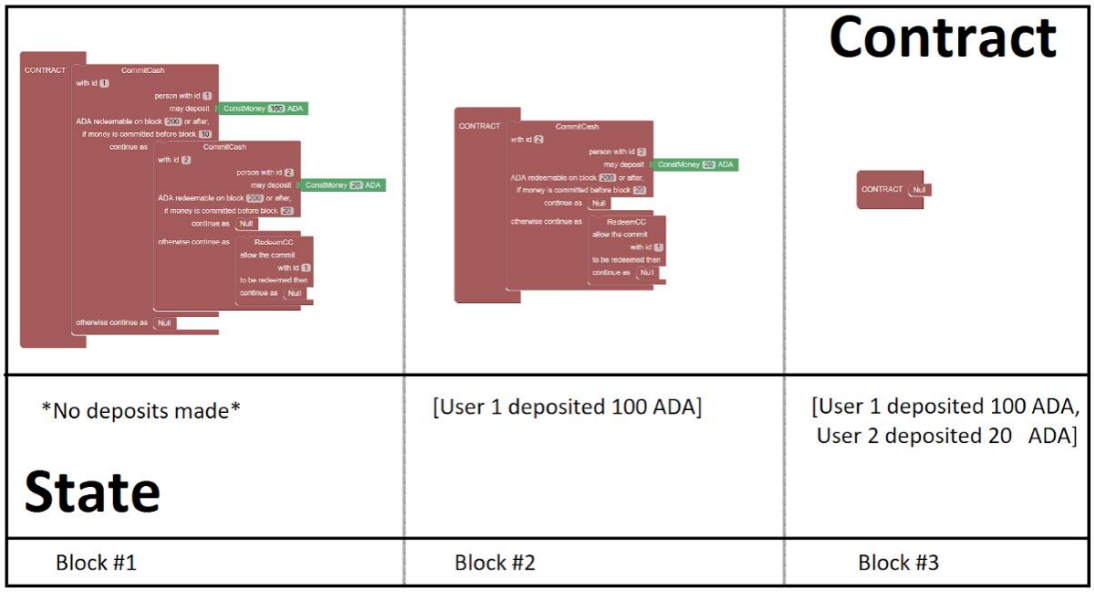

Figura 2.7: Smart Contract desarrollado en Marlowe. [51]

Como se mencionó Plutus es un lenguaje funcional, por lo que posee varias ventajas sobre un lenguaje imperativo, entre ellas se puede mencionar:

- Creación de código de alta seguridad, es más fácil demostrar matemáticamente cómo se va a comportar el código.
- Aumenta la legibilidad y la mantenibilidad, cada función está diseñada para realizar una tarea específica. Las funciones también son independientes del estado.
- Cualquier cambio en el código es más sencillo de implementar, esto facilita el desarrollo reiterativo.
- Las funciones individuales se pueden aislar fácilmente, lo que hace que sean más fáciles de probar y depurar.

2.4.8. Neo

La solución NEO, vista por algunos como la contraparte china de Ethereum, se basa en el uso de tecnología de Blockchain y el concepto de la identidad digital, con el fin de lograr una “economía inteligente”. A pesar de que incorpora conceptos y características que la hacen atractivas por sobre otras opciones, en la actualidad muchas características continúan en desarrollo, y aunque lamentablemente la comunidad de desarrolladores de esta plataforma es aún pequeña comparable con otras como: Ethereum, presenta en su página oficial un reporte mensual del estado de desarrollo de la plataforma.

Con respecto a la identidad digital, se refiere a la información de identidad de individuos, organizaciones y otras entidades que existen en forma electrónica. NEO, implementa un conjunto de estándares de identidad digital compatibles con X.509. Este conjunto de estándares de identidad digital, además del modelo compatible de emisión de certificados de nivel X.509,

también soporta el modelo de emisión de certificados punto a punto de Web Of Trust.

Para lograr su cometido de desarrollar una “economía inteligente”, la red proporciona múltiples funciones como las capacidades de activos digitales, NeoAsset, Smart Contracts, NeoContract y la identidad digital, NeoID, lo que permite a los usuarios participar fácilmente en negocios digitales, y ya no se limitan a la emisión de tokens nativos en la cadena de bloques.

2.4.8.1. Características

Actualmente la versión NeoContracts 2.0 se encuentra en desarrollo, esta versión mejorará características que la versión actual ya posee como:

- Acoplamiento
- Alto Desempeño
- Escalabilidad

Acoplamiento: Es una medida de la dependencia entre dos o más entidades. El sistema NeoContract utiliza un diseño de bajo acoplamiento, que se ejecuta en el entorno de los contratos inteligentes, y se comunica con los datos que no pertenecen a la cadena de bloques a través de la capa de servicio interoperable. Como resultado, la mayoría de las actualizaciones de las funciones inteligentes de los contratos pueden realizarse mediante la API de los servicios interoperables.

Alto Desempeño: Para analizar el desempeño de una plataforma, es necesario revisar el entorno de ejecución, para ello existen dos indicadores que son críticos:

- Velocidad de ejecución de la instrucción
- Velocidad de inicio del entorno de ejecución

Debido a la configuración de los Smart Contracts en Neo, cada vez que se llama al contrato inteligente éste debe iniciar el entorno en el cuál se ejecutan. Por lo tanto, la velocidad de ejecución del entorno tiene un gran impacto en el rendimiento del sistema inteligente de contratos.

Los NeoContract utilizan una máquina virtual como entorno de ejecución NeoVM (NEO Virtual Machine), que fue diseñada para el consumo de pocos recursos mediante a su configuración didáctica. Esta configuración proporciona una serie de instrucciones criptográficas para optimizar la eficiencia de ejecución de los algoritmos criptográficos en los contratos inteligentes.

Escalabilidad: Para analizar la escalabilidad de un sistema, existen dos características principales que son tomadas en cuenta:

- Escalado vertical
- Escalado horizontal

En el primer escalado, se refiere a la optimización del flujo de trabajo, es decir, busca aprovechar al máximo la capacidad de proceso de los equipos existentes en la red. Mientras que el horizontal se basa en el incremento de equipos, lo cual hace a este sistema de escalado dependiente directamente de la cantidad de dispositivos de la red.

Las cadenas de bloques pueden procesar programas en paralelo, por lo tanto, si los Smart Contracts no interactúan entre sí, o si el contrato no modifica los mismos datos de estado al mismo tiempo, su ejecución no es secuencial y pueden actuar simultáneamente. Como resultado, los Smart Contracts pueden procesarse en paralelo, ya que el orden secuencial es irrelevante para el resultado.

2.4.8.2. Clases de contratos

Al igual que Neo plantea ventajas que lo diferencian de las demás plataformas en cuanto a las características de sus Smart Contracts, también lo hace en los tipos de contrato que define, para distintos usos:

- Contratos de aplicación
- Contratos de función
- Contratos de validación

Contratos de aplicación: se desencadenan mediante una transacción especial, que puede acceder y modificar el estado global del sistema y el estado privado del contrato en tiempo de ejecución. Por ejemplo, pueden crear un activo digital global en un contrato, votar, guardar datos e incluso crear dinámicamente un nuevo contrato, cuando el contrato está en ejecución. La ejecución del contrato de solicitud requiere el cobro por instrucción.

Contratos de función: en el caso de este tipo de contrato, se utiliza para proporcionar algunas funciones públicas o de uso común, es decir, aquellos que pueden ser llamadas por otros contratos. El código de contrato inteligente puede ser reutilizado, de modo que los desarrolladores puedan escribir una lógica de negocio cada vez más compleja.

Para el correcto uso de los contratos de función, deben ser ejecutados antes del contrato que lo utilizará en la cadena de bloques, y retirado del Blockchian por una función de sistema "autodestructor". Los datos del contrato antiguo se pueden migrar automáticamente a otro subcontrato antes de que se destruya, utilizando herramientas de migración de contratos.

Contratos de verificación: finalmente, los contratos de verificación o validación se utilizan como reemplazo al sistema de cuentas de clave pública utilizado en Bitcoin, a cada cuenta en la plataforma le corresponde un contrato de verificación, y el valor hash del contrato de verificación es la dirección de la cuenta; la lógica del programa del contrato de verificación controla la propiedad de la cuenta. Al realizar una transferencia desde una

cuenta, en primer lugar debe ejecutarse el contrato de verificación para esa cuenta. Un contrato de validación puede aceptar un conjunto de parámetros (normalmente una firma digital u otros criterios) y devolver un valor booleano después de la verificación, indicando el éxito de la verificación al sistema.

2.4.8.3. Tasas

NEO proporciona un entorno de ejecución creíble para los contratos inteligentes, y la ejecución de los contratos requiere el consumo de recursos informáticos para cada nodo, por lo que los usuarios deben pagar por la ejecución de los contratos inteligentes. La tarifa es determinada por los recursos computacionales consumidos en cada ejecución, y el precio unitario es también en GAS (NeoGas es uno de los tokens que utiliza Neo para su funcionamiento). Si la implementación del contrato inteligente falla por falta de GAS, no se devolverá el coste del consumo, lo que evita ataques maliciosos al consumo de energía de la red.

La mayoría de los contratos pueden ser ejecutados de forma gratuita, siempre y cuando los costes de ejecución se mantengan por debajo de 10 GAS, reduciendo así en gran medida los costes para el usuario.

2.4.8.4. Lenguaje

Una de las principales ventajas que apunta NEO, es la idea de que los desarrolladores puedan participar directamente en la programación de un contrato inteligente NEO sin necesidad de aprender un nuevo idioma. Por ello se busca la posibilidad de programar contratos inteligentes en los lenguajes de programación de alto nivel más comunes de la actualidad.

En la versión NEO Contract 2.0, uno de los objetivos es proporcionar compiladores y plug-ins para los lenguajes que admite, el primer compilador será para CIL (Common Intermediate Language), por lo que teóricamente cualquier lenguaje de red y cualquier idioma que pueda ser traducido a CIL será soportado inmediatamente.

Los idiomas que son admitidos actualmente son:

- C#, VB.Net, F#
- Java, Kotlin
- Python

Mientras que entre los idiomas que se planifica admitir, se incluyen:

- C, C++, Golang
- Javascript

2.4.8.5. Entorno

Como se ha mencionado, el entorno en el cual se ejecutan los contratos inteligentes es una máquina virtual, NeoVM, en la figura 2.8 se aprecia la

arquitectura de la misma, donde el despliegue en el cuadro de puntos es el núcleo de la máquina virtual.

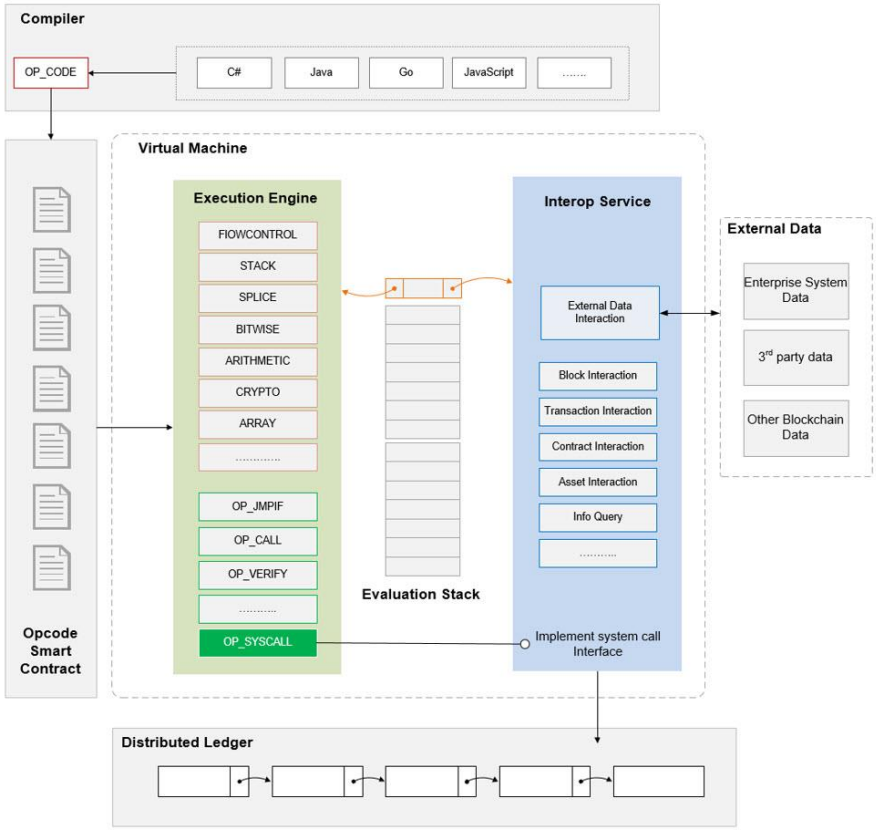

Figura 2.8: Arquitectura de la máquina virtual NeoVM.

En el núcleo observado en la figura 2.8, se pueden distinguir tres secciones de la NeoVM:

- Motor de ejecución
- Pila de evaluación
- Capa de servicio interoperable

Motor de ejecución (Execution Engine): El motor de ejecución de la máquina virtual es el equivalente a la CPU. Puede ejecutar instrucciones comunes como control de flujo, operaciones de pila, operaciones de bit, operaciones aritméticas, operaciones lógicas, métodos criptográficos, etc. Puede interactuar con la capa de servicio Interoperable a través de llamadas al sistema.

Pila de evaluación (Evaluation Stack): La pila de evaluación es equivalente a la memoria, en la actualidad hay dos maneras en las que se puede desarrollar la memoria en una máquina virtual: Pila o registro.

Debido a que el comportamiento por defecto de una VM basada en una pila es obtener datos de la pila de operandos, no hay necesidad de especificar un operando. Por ejemplo, la operación adición "ADD" operará directamente

en la pila de operandos. Los datos se pueden extraer directamente y el resultado se almacena en el tope de la pila.

Capa de servicio interoperable (Interop Service): La capa de servicio interoperable de la máquina virtual es el equivalente a los periféricos. En la actualidad, la capa de servicios interoperable proporciona algunas API para acceder a los datos del Blockchain del contrato inteligente. Puede acceder a información de bloques, información de operaciones, información de contratos, información de activos fijos, etc.

2.4.9. Ethereum

Esta plataforma de código abierto diseñada para la implementación de Smart Contracts fue lanzada en el año 2015, por Vitalik Buterin, quién vio en la tecnología Blockchain la herramienta necesaria para desarrollar la idea de Nick Szabo.

Ethereum posee su propia criptomoneda, el Ether (ETH), la cual ocupa el segundo lugar en términos de capitalización monetaria, como en la mayoría de implementaciones sobre Blockchain los participantes publican transacciones en la red que son agrupadas en bloques por distintos nodos (llamados mineros) y añadidos a la cadena de bloques, a partir de un mecanismo de consenso (PoW).

Para programar y ejecutar un Smart Contract es necesario una Máquina Virtual Ethereum (EVM), los cuales poseen un costo que el emisor del mismo añade, ese costo es entregado a los mineros que añaden la transacción a un bloque, el cual se añade al Blockchain de Ethereum.

En el caso que los contratos necesiten representar un bien que no esté expresado necesariamente como una cantidad de Ether, Ethereum proporciona la posibilidad de usar tokens para esto, los cuales poseen además su estándar, el más conocido ERC-20.

2.4.9.1. EVM

El entorno en el que se ejecutan los Smart Contracts en Ethereum es la Máquina Virtual Ethereum (EVM), debido a como se encuentra conformada el código del Contrato Inteligente que se ejecuta en la EVM no tiene acceso a la red, al sistema de archivos, ni a otros procesos. Los contratos inteligentes incluso tienen un acceso limitado a otros contratos inteligentes.

Los contratos se almacenan en el Blockchain de Ethereum en un formato binario específico de la plataforma (EVM bytecode), sin embargo, los contratos se escriben típicamente en un lenguaje de alto nivel, se compilan en código de bytes utilizando un compilador EVM y finalmente se cargan en la cadena de bloques utilizando un cliente Ethereum. [52]

Como el núcleo de la EVM es una máquina basada en pilas, el conjunto de instrucciones en EVM bytecode consiste principalmente en instrucciones

estándar para operaciones de pila, aritmética, saltos y acceso a la memoria local. Además de estas típicas instrucciones para pila, existen opcodes para diversas funciones como:

- Uso de hash.
- Acceso al entorno en el que se llamó al contrato.
- Acceso y modificación del almacenamiento de la cuenta que actualmente está ejecutando el código.
- Realización de llamadas internas.

2.4.9.2. Minería

En la plataforma Ethereum como se describió en los componentes de un Blockchain en el capítulo anterior, existen nodos que cumplen la función de mineros, los cuales agrupan las transacciones enviadas por los usuarios en bloques, e intentan añadirlos al Blockchain para cobrar las tarifas asociadas. Sólo los bloques que satisfacen una PoW (Ethash) se añaden. La dificultad del PoW se actualiza dinámicamente para que la tasa media de extracción sea de 1 bloque cada 12 segundos. [53]

Cuando un minero resuelve el Ethash y transmite un nuevo bloque válido a la red, los otros mineros descartan sus intentos, actualizan su copia local de la cadena de bloques añadiendo el nuevo bloque y comienzan a "minar" encima de él. El minero que resuelve el PoW es recompensado con los honorarios de las transacciones en el nuevo bloque. [53]

La PoW Ethash se basa en el algoritmo de Dagger Hashimoto, tiene el objetivo de ser rápidamente verificable con una sobrecarga de memoria baja y de rápido minado. Los pasos de minado en el algoritmo Ethash, los cuales se observan en la figura 2.9, son: [54]

1. Se generan conjuntos de datos para la verificación (16 MB de caché) y para la minería (1 GB). Los conjuntos de datos se derivan con la función hash Keccak (primero se realiza el hash de los encabezados de los bloques para obtener una semilla, luego hash de la semilla para obtener una caché y finalmente hash del caché para obtener un gran conjunto de datos). Los conjuntos de datos se generan una sola vez al inicio de las operaciones mineras y se actualizan una sola vez cada 30000 bloques. Por lo tanto, la generación de datos no afecta significativamente a la eficiencia de la cadena de bloques Ethereum.
2. Como se observa en la figura 2.9, se combina y hace hash (al estilo de Hashimoto) de un nonce aleatorio, un hash del encabezado del bloque y rebanadas aleatorias del conjunto de datos. Esta función se mantiene en bucle hasta que se alcanza un valor que satisface el umbral de dificultad (y se encuentra el PoW). Para cada ronda se incrementa el valor de nonce y se obtienen nuevas rebanadas de la memoria. La principal carga de trabajo en Ethash se encuentra en esta fase.

3. Un bloque con la prueba de trabajo es enviado a la red Ethereum.

La cantidad de rondas de corte se ajusta dinámicamente. El número de rondas depende del parámetro de dificultad, que a su vez depende de la frecuencia de los bloques procesados por la red Ethereum. En consecuencia, la cantidad de trabajo y la eficiencia de la minería Ethereum varía ligeramente con el tiempo. [54]

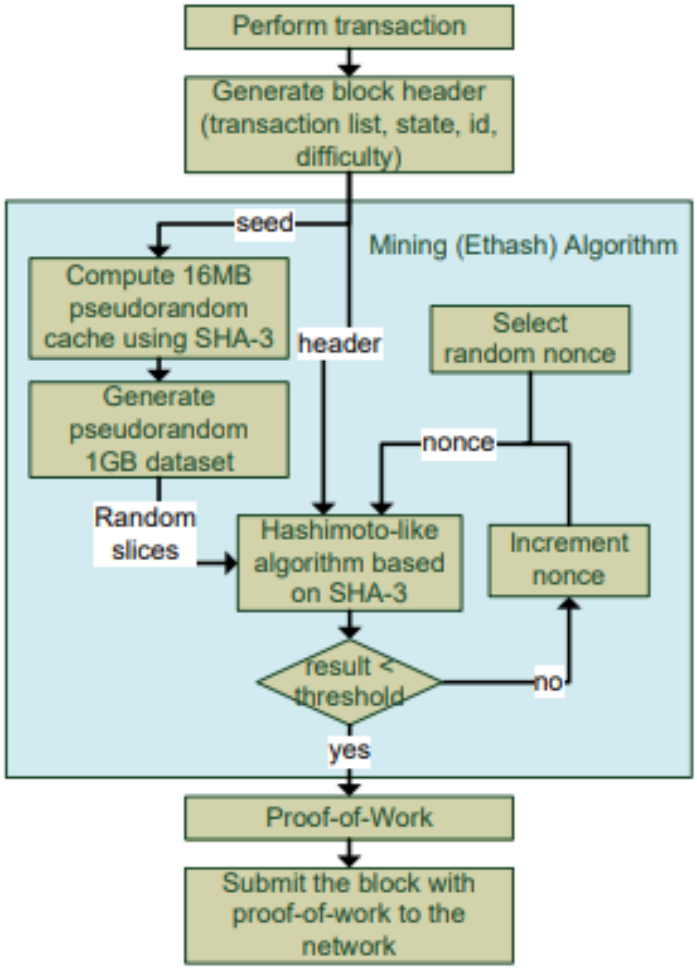

Figura 2.9: Fases del proceso de minería de Ethereum. [54]

2.4.9.3. Tipo de Cuentas

En la plataforma Ethereum, existen dos tipos de cuentas:

- **Cuentas de propiedad externa (EOAs)**: son controladas por claves privadas.
- **Cuentas de contrato:** son controladas por su código de contrato y sólo pueden ser "activadas" por un EOA.

La mejor forma de diferenciar los tipos de cuentas en Ethereum es la siguiente los usuarios humanos controlan las EOAs, debido a que controlan

las claves privadas. Mientras que, las cuentas de contratos se rigen por su código interno. [52]

Ethereum permite implementar un Smart Contract en su Blockchain, mediante una cuenta de contrato, la cual se ejecuta cuando se envía una transacción a esa cuenta. Esto siempre se cumple, debido a que, las cuentas de contrato sólo realizan una operación cuando se lo ordena un EOA. Por lo tanto, no es posible que una cuenta de contrato realice operaciones como la generación de números aleatorios o llamadas a una API, a menos que se le pida un EOA. [52]

En la figura 2.10, se muestran los cuatro elementos que componen el estado de una cuenta (Account State):

- **Nonce:** Si la cuenta es de propiedad externa (EOA), este valor representa el número de transacciones enviadas desde la dirección de la cuenta. Sin embargo, si se trata de una cuenta de contrato, entonces el nonce significa el número de contratos creados por la cuenta.
- **Saldo:** La cantidad de Wei que posee la dirección de la cuenta.
- **Code Hash:** Valor hash del código de la Máquina Virtual Ethereum (EVM) para la cuenta correspondiente, este es el código que se ejecuta si esta dirección recibe una llamada.
- **Storage Hash:** Un hash de 256 bits del nodo raíz de un árbol Merkle que codifica el contenido almacenado de la cuenta.

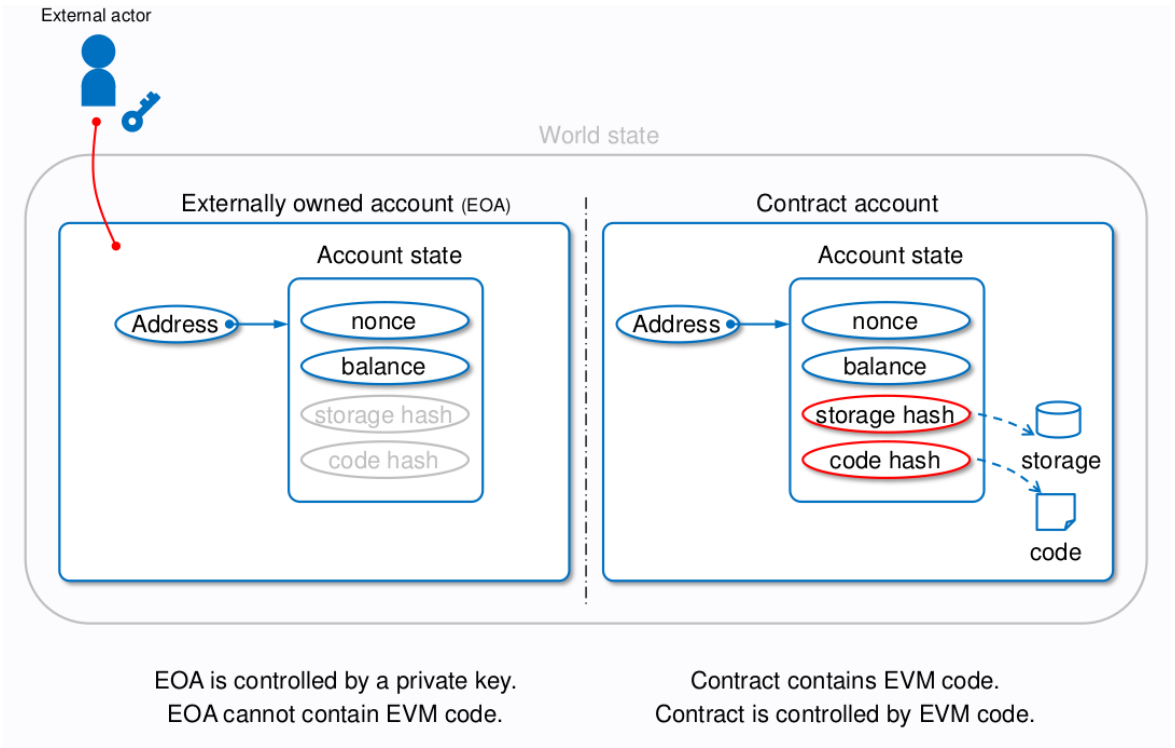

Figura 2.10: Tipos de cuenta en Ethereum. [55]

2.4.9.4. Lenguaje

Para programar Smart Contracts en la plataforma Ethereum, existen lenguajes de alto nivel como Serpent o Solidity, los cuales son de tipo Turing completo.

Solidity es el lenguaje estándar *de facto*, al ser el de mayor uso por parte de desarrolladores, así como el que posee una mayor comunidad detrás. Es llamado como un lenguaje de programación "orientado a contratos", debido a que utiliza el concepto de clase de los lenguajes orientados a objetos para la representación de contratos. [56]

Los contratos especifican campos y métodos para instancias de contratos. Los campos pueden ser vistos como almacenamiento persistente de un contrato (instancia) y los métodos de contrato pueden ser invocados por defecto por cualquier transacción interna o externa. [56]

La sintaxis de Solidity es similar a la de JavaScript, con primitivas adicionales para acceder a la transacción y a la información de bloque, entre otras, que fueron añadidas debido a su funcionalidad en el entorno de Ethereum. [56]

2.4.9.5. Costos

Para la realización de transacciones en el Blockchain, Ethereum posee el concepto de “Gas”, el cual sirve para cuantificar la cantidad de potencia computacional necesaria para ejecutar contratos inteligentes a través de la EVM, esta cuantificación se realiza en Wei.

La plataforma Ethereum posee su propia moneda, el “Ether”, esta moneda se puede subdividir en cantidades más pequeñas de “Wei”, el Wei es la fracción más pequeña de Ether, un Ether es igual a 10^{18} Wei. [57]

Cuando se envía un contrato inteligente, el emisor especifica el valor en gas del Smart Contract. Cada operación realizada en el código del contrato inteligente requiere que se ejecute una cantidad predeterminada de Gas, en la figura 2.11 se muestra la cantidad de Gas necesaria para la realización de algunas transacciones en la plataforma Ethereum. [57]

APPENDIX G. FEE SCHEDULE

The fee schedule G is a tuple of 31 scalar values corresponding to the relative costs, in gas, of a number of abstract operations that a transaction may effect.

Name	Value	Description*
G_{zero}	0	Nothing paid for operations of the set W_{zero}.
G_{base}	2	Amount of gas to pay for operations of the set W_{base}.
$G_{verylow}$	3	Amount of gas to pay for operations of the set $W_{verylow}$.
G_{low}	5	Amount of gas to pay for operations of the set W_{low}.
G_{mid}	8	Amount of gas to pay for operations of the set W_{mid}.
G_{high}	10	Amount of gas to pay for operations of the set W_{high}.
$G_{extcode}$	700	Amount of gas to pay for operations of the set $W_{extcode}$.
$G_{balance}$	400	Amount of gas to pay for a BALANCE operation.
G_{sload}	200	Paid for a SLOAD operation.
$G_{jumpdest}$	1	Paid for a JUMPDEST operation.
G_{sset}	20000	Paid for an SSTORE operation when the storage value is set to non-zero from zero.
G_{sreset}	5000	Paid for an SSTORE operation when the storage value's zeroness remains unchanged or is set to zero.
R_{sclear}	15000	Refund given (added into refund counter) when the storage value is set to zero from non-zero.
$R_{suicide}$	24000	Refund given (added into refund counter) for suiciding an account.
$G_{suicide}$	5000	Amount of gas to pay for a SUICIDE operation.
G_{create}	32000	Paid for a CREATE operation.
$G_{codedeposit}$	200	Paid per byte for a CREATE operation to succeed in placing code into state.
G_{call}	700	Paid for a CALL operation.
$G_{callvalue}$	9000	Paid for a non-zero value transfer as part of the CALL operation.
$G_{callstipend}$	2300	A stipend for the called contract subtracted from $G_{callvalue}$ for a non-zero value transfer.
$G_{newaccount}$	25000	Paid for a CALL or SUICIDE operation which creates an account.
G_{exp}	10	Partial payment for an EXP operation.
$G_{expbyte}$	10	Partial payment when multiplied by $\lceil \log_{256}(exponent) \rceil$ for the EXP operation.
G_{memory}	3	Paid for every additional word when expanding memory.
$G_{txcreate}$	32000	Paid by all contract-creating transactions after the *Homestead transition*.
$G_{txdatazero}$	4	Paid for every zero byte of data or code for a transaction.
$G_{txdatanonzero}$	68	Paid for every non-zero byte of data or code for a transaction.
$G_{transaction}$	21000	Paid for every transaction.
G_{log}	375	Partial payment for a LOG operation.
$G_{logdata}$	8	Paid for each byte in a LOG operation's data.
$G_{logtopic}$	375	Paid for each topic of a LOG operation.
G_{sha3}	30	Paid for each SHA3 operation.
$G_{sha3word}$	6	Paid for each word (rounded up) for input data to a SHA3 operation.
G_{copy}	3	Partial payment for *COPY operations, multiplied by words copied, rounded up.
$G_{blockhash}$	20	Payment for BLOCKHASH operation.

Figura 2.11: Tasa de Gas para transacciones en Ethereum. [58]

Por lo tanto, para que un individuo pueda realizar una transacción en la red Ethereum, el emisor debe establecer el límite de gas y el precio del gas adjunto a la transacción. Si un remitente no posee el gas necesario para realizar una transacción, entonces se dice que "se queda sin gas" y el contrato es inválido. [57]

El concepto de “Gas” posee además la capacidad de reducir el uso innecesario de poder computacional, al limitar el número de cálculos que puede realizar la EVM en las siguientes maneras: [57]

- Los bloques que se extraen en el Blockchain de Ethereum, tienen un límite de gas adherido, por lo que la cantidad de gas utilizada en todas las transacciones dentro de un bloque no puede exceder una cierta cantidad.
- El costo del gas además limita que algunos Smart Contracts, pueden tener demasiadas transacciones que resulten demasiado complejas y por consiguiente económicamente poco prácticas.

2.4.9.6. Token

Como se ha mencionado, Ethereum ofrece la posibilidad de crear Propiedad Inteligente (Smart Property) en su Blockchain a través de la representación de activos mediante tokens. Estos tokens y las transacciones asociadas a ellos pueden ser rastreados en el Blockchain de Ethereum, son programados

con algunos de los estándares que ofrece Ethereum, los cuáles se encuentran en constante desarrollo.

- **ERC-20:** Fue el primer estándar en ganar popularidad, la idea tras la creación de ERC-20 es definir una lista de reglas para la creación y uso de los token, lamentablemente al ser el primer estándar posee varias fallas de diseño y de seguridad, que han provocado la pérdida de más de 3 millones de dólares. Existen dos formas que utiliza el estándar ERC-20 para realizar una transacción de token: [46] [59]
  1- **transfer():** esta función activa el envío de tokens a la dirección de un usuario específico.
  2- **approve() + transferFrom():** esta función enciende el depósito de tokens a un contrato inteligente predefinido.

  ERC-20 es en la actualidad el estándar *de facto*, algunos ejemplos de su uso son las implementaciones: EOS, TRON, VeChain, OmiseGO, ICON, entre otras.

- **ERC-223:** Se creó como resultado de la búsqueda de eliminación de los problemas que presenta el estándar ERC-20. Este estándar elimina la falla de la función transfer(), logrando que arroje un error en respuesta a transferencias inválidas y cancele la transacción de manera que no se pierdan fondos. Un ejemplo de esta falla, es una transferencia en la cual se utiliza un token ERC-20 en un Smart Contract que envía 5 ether a otro Smart Contract, el cual es incompatible con ERC-20, la transacción no será negada porque el contrato no reconocerá la transacción entrante, por lo cual los ether podrían quedarse atascados en el limbo y acabar perdiéndose. [46]
- **ERC-721:** Este estándar se hizo conocido por implementaciones como: CryptoKitties, CryptoPunks, CriptoCelebrities y EtherTulips.
  La innovación más importante que presenta esta versión de estándar de token, es permitir crear Tokens no fungibles, es decir, un token puede poseer un valor único y por tanto diferente que otro token con el cual se intercambia en un Smart Contract. [59]

Es necesario mencionar que existen otros estándares que aún se encuentran en revisión y no han sido declarados como oficiales, entre ellos están: ERC-777, ERC- 827, ERC-1155, entre otros.

Una de las grandes ventajas que plantea la estandarización de los tokens es la posibilidad de que otros sistemas, los cuales se basan en Ethereum como las Aplicaciones Descentralizadas (Đapps), las cuales han proliferado en los últimos años, utilicen estos estándares para el desarrollo de sus propios tokens y posean compatibilidad con los tokens de Ethereum. [46] [59]

2.5. Usos

Se podría decir en muchos casos, que la utilización de un objeto tiene como límite la imaginación de quien lo desea ocupar, por ello cuando se trata de contratos inteligentes, se podría decir que su uso es transversal a cualquier tipo de negocio y podríamos ocuparlo en la mayor parte de aspecto de nuestra vida diaria, definiendo diferentes tipos de transacciones en los cuales se centrará el Smart Contract, a continuación, se enuncian algunos de ellos:

2.5.1. Sistema de votación

En la actualidad existen tanto detractores como defensores de los sistemas de voto electrónico, y a pesar de que existen países tanto desarrollados como subdesarrollados que poseen implementado algún tipo de sistema de votación electrónica, se han suscitado registrados problemas asociados a este tipo de votación en países como: Argentina, Brasil, Estados Unidos, Finlandia, Holanda, India, entre otros.

Por ello resulta más que interesante un sistema de Smart Contracts basado en Blockchain capaz de asegurar la transparencia del voto. Los votos protegidos por el libro mayor (cadena de bloques) tendrían que ser decodificados y requerirían una potencia de cálculo excesiva para poder acceder a modificarlos. Además, se podría garantizar la confidencialidad de cada voto, además de agilizar tanto el conteo como una auditoría del correcto conteo de los votos. [60]

2.5.2. Historial médico

Si bien esta aplicación se podría generalizar para cualquier tipo de información que deseemos almacenar, el historial médico es uno de los principales objetivos de los hackers de sombrero negro. A través de los contratos inteligentes se podría codificar y almacenar los informes médicos de los pacientes, utilizando una clave privada que sólo estaría disponible para personas autenticadas para su acceso. Tanto enfermedades, como recibos de medicamentos, entre otras cosas, pueden ser almacenados y enviados automáticamente a proveedores de seguros como prueba de entrega. El libro mayor también podría utilizarse para la gestión de la atención médica como: la supervisión de medicamentos, el cumplimiento de las normas, los resultados de las pruebas y gestión de los suministros de atención médica. [1]

2.5.3. Identidad en la red

Los contratos inteligentes pueden permitir que las personas posean y controlen su identidad digital, es decir, los datos personales, reputación y activos digitales que poseen. Además, se puede implementar una restricción destinada a permitir que los individuos decidan qué datos revelar a las contrapartes, proporcionando a las empresas la oportunidad de conocer a sus clientes a la perfección, esto sin poseer datos sensibles para verificar a

que persona corresponde la información contenida. Un ejemplo de esto se puede observar en la figura 2.12, en la cual un contrato inteligente evita a un banco entregar directamente información sensible de sus clientes a la entidad gubernamental asociada, la cual podría utilizarla para almacenarla y así obtener más información de los ciudadanos que entrega el banco.

Esto reduce la responsabilidad al tiempo que facilita los requisitos de conocer al cliente sin fricción. También aumenta el cumplimiento, la resistencia y la interoperabilidad. [61]

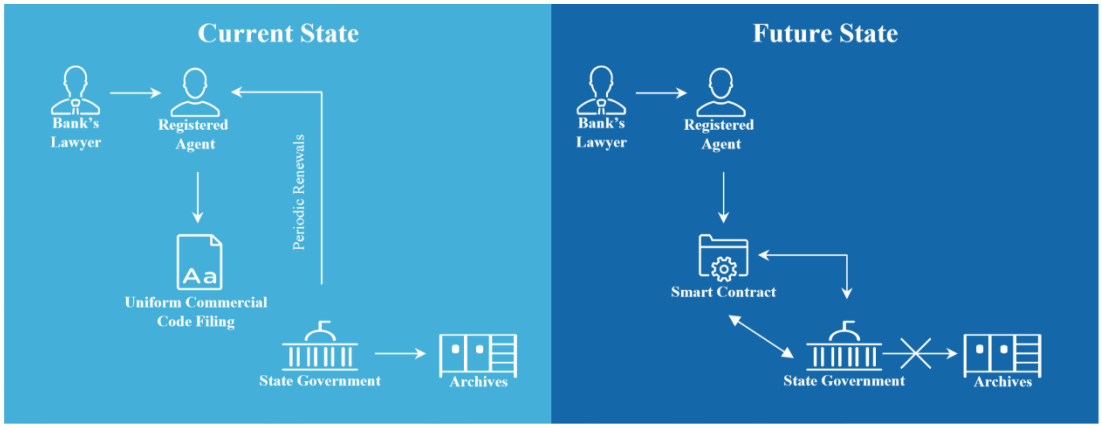

Figura 2.12: Ejemplo del funcionamiento de un Smart Contract. [61]

2.5.4. Industria musical

Existen varios problemas asociados con la industria de la música, entre los más importantes se puede mencionar: los derechos de autor, la propiedad, la transparencia y la distribución de las regalías. Los contratos inteligentes pueden resolver estos problemas creando una base de datos de derechos de música descentralizada. Proporcionará la transferencia de los derechos de autor y las distribuciones de los artistas en tiempo real. Los reproductores de música se pagarán utilizando la moneda digital según lo estipulado en el contrato. [62]

2.5.5. Comercio electrónico

Cuando se piensa en el uso de los Contratos Inteligentes, el comercio electrónico es posiblemente el principal caso de aplicación. La facilitación del comercio entre partes no confiables sin un tercero de confianza resultaría en una reducción de costes en la actividad comercial.

Como se ha mencionado, los contratos inteligentes sólo pueden liberar el pago al vendedor una vez que el comprador se encuentre satisfecho con el producto o servicio recibido, como se puede observar en la figura 2.13, donde tanto el pago como el envío del producto utiliza Smart Contracts para su aprobación y cumplimiento. Pero esto no se limitaría a productos que se podrían conseguir en una página de retail, por ejemplo: En general, cuando

se desea alquilar un apartamento, se paga a una empresa para hacer publicidad y una vez que alguien desea alquilar el lugar, se paga a un tercero (Banco, App, entre otros) para confirmar que la persona pagó el alquiler y siguió adelante. Mientras que utilizando contratos inteligentes, todo lo que se debe hacer es pagar a través de criptodivisas y codificar el contrato en el libro de contabilidad. Se logra transparencia, y cumplimiento automático, a esto también se lo conoce como “Propiedad Inteligente”. [1] [60]

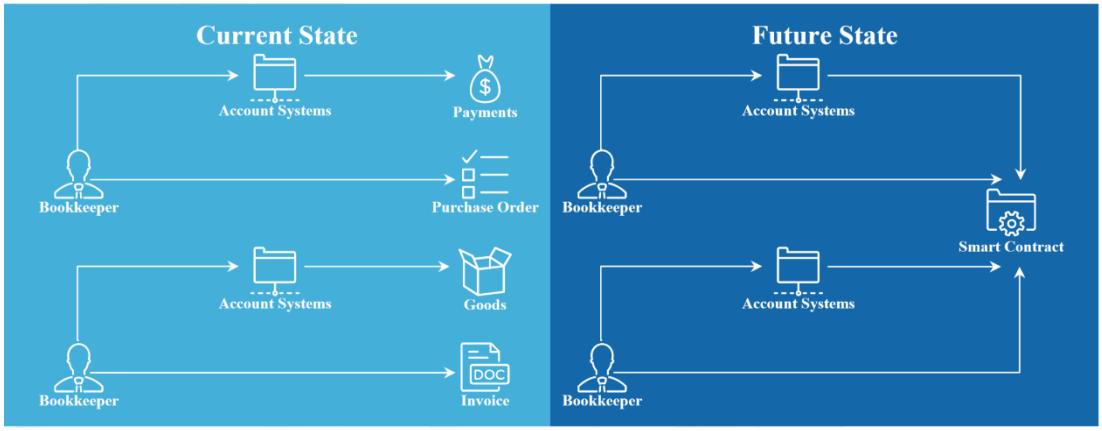

Figura 2.13: Ejemplo del funcionamento de un Smart Contract. [61]

2.6. Legalidad

La vigencia legal de cualquier tipo de contrato, depende del cumplimiento de las leyes locales de la jurisdicción en la que se desea aplicar, tratados internacionales, entre otras cosas. De momento los Contratos Inteligentes poseen ciertas dificultades para ser aceptados como legales.

A pesar de que los primeros desarrollos prácticos de Smart Contracts no poseen más de seis años de antigüedad, existen territorios en los que ya se ha comenzado el camino jurídico para convertirlos en una opción a la hora de realizar un contrato entre partes.

Un ejemplo de esto se da en EEUU, en el cual, los estados de Arizona, California, Nevada, Tennessee y Ohio, modificaron sus respectivas variantes de la Ley de Transacciones Electrónicas Uniformes (UETA) específicamente para que los registros mantenidos en un Blockchain sean considerados como "registros electrónicos" de acuerdo a lo expresado en UETA. Esta modificación permitiría que en un futuro un legislador pueda afirmar que toda firma digital registrada en una cadena de bloques sea ejecutable y, además, que un Smart Contract se considere un agente electrónico, término que se define en referencia a "iniciar una acción o responder a registros o interpretaciones electrónicas". [63]

A pesar que este cambio se pueda observar como un avance en el reconocimiento de los Smart Contracts, existen aún dificultades incluso en las leyes aprobadas, como es el caso específico de Arizona, donde existe un

problema con la definición de Blockchain, puesto que en este estado se lo define como un libro contable "inmutable"(“immutable” ledger), capaz de proporcionar una "verdad sin censura"(“uncensored truth”). Lamentablemente como se ha mencionado, las cadenas de bloques pueden ser modificadas y cambiadas, en casos limitados, y por lo tanto no encajan estrictamente en la definición de Arizona. [63]

Otra limitante para reconocer a un Smart Contract como un acuerdo legal válido, es la incertidumbre en cuanto a los términos que plantea, especialmente las partes del contrato.

Un contrato inteligente identifica a las partes de la transacción basándose únicamente en las direcciones públicas, y a sus respectivos propietarios. Sin embargo, existen por lo menos dos razones que limitan la certeza en cuanto a las partes intervinientes. [64]

- La dirección pública en una transacción de un Smart Contract puede apuntar a otro contrato inteligente en lugar de una billetera con un propietario determinado. Esta información no se puede distinguir sólo con la dirección pública, porque la dirección pública de una billetera es a menudo indistinguible de la de un Smart Contract. [64]
- Incluso si la dirección pública pertenece a una billetera, el propietario de esa billetera sigue siendo seudónimo y no se puede distinguir de la dirección pública sin información externa. [64]

Por lo tanto, actualmente parece imposible identificar a las partes de un Smart Contract basado en las direcciones públicas. Esta dificultad puede eventualmente ser superada por el desarrollo de identidades en los Blockchain en los cuales se ejecutan. Hasta entonces, un Smart Contract en sí mismo puede no equivaler a un acuerdo legal válido debido a la falta de certeza en cuanto a la identidad de las partes. [63] [64]

CAPÍTULO 3: Prueba de Concepto

En este último capítulo se desarrollará y documentará una PoC de la utilización de Smart Contracts en una transacción entre partes para mostrar la practicidad de los mismos y obtener las conclusiones correspondientes. La plataforma en la cual se desarrollará la prueba de concepto será elegida entre las vistas en el capítulo 2.

3.1. Caso

Para realizar la prueba de concepto de los Smart Contracts se tomará el caso de una subasta como se puede observar en la figura 3.1, la cual tendrá las siguientes características:

- Quien despliegue el contrato no necesariamente es el beneficiario.
- Se definirá un tiempo límite de puja.
- Los postores deben conocer la identidad del beneficiario de la subasta.
- Durante la subasta cualquier pujante podrá verificar la mayor oferta y quién lo realiza.
- Aquel postor o postores que no realicen la mayor oferta podrán recibir el valor de su apuesta de vuelta.

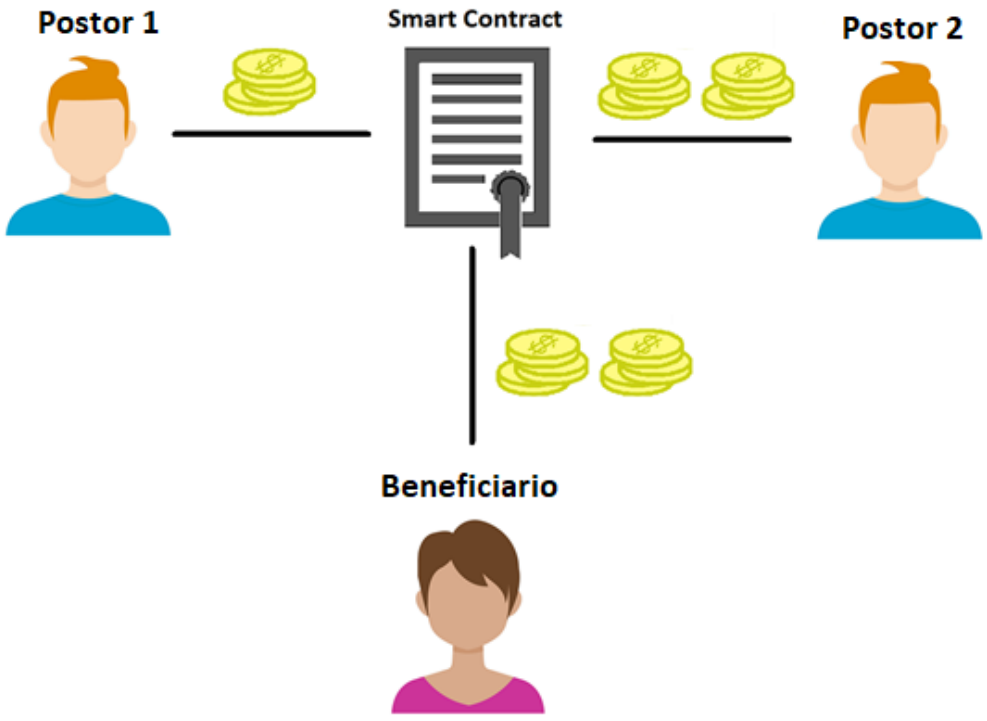

Figura 3.1: Caso utilizado como PoC.

3.2. Elección de plataforma

Como se mencionó en el párrafo anterior, esta elección se realizará a partir de las opciones planteadas en el capítulo anterior, las cuales representan las principales opciones actuales del mercado, estas son:

1. Bitcoin
2. NXT
3. Counterparty
4. Stellar
5. Monax
6. Lisk
7. Cardano
8. Neo
9. Ethereum

Debido a que la tecnología requerida para el desarrollo, uso y manejo de Smart Contracts posee menos de una década (2009), algunas de las opciones planteadas no se encuentran en una fase de maduración que al día de hoy permita su uso de forma comercial. Por esta razón se procederá a eliminar aquellas que aún se encuentran en una fase muy temprana de desarrollo.

Plataforma	Estado de Madurez	Ambiente de Prueba	Uso comercial
Bitcoin	Desarrollado	N/A	Sí
NXT	Desarrollado	N/A	Sí
Counterparty	Desarrollado	N/A	Sí
Stellar	En fase de prueba	Sí	No
Monax	Desarrollado	N/A	Sí
Lisk	En fase de prueba	Sí	No
Cardano	En fase de prueba	Sí	No
Neo	En fase de prueba	Sí	No
Ethereum	Desarrollado	N/A	Sí

Tabla 3.1: Comparación del estado de madurez de las plataformas vistas. [47] [65]

A partir del análisis realizado en la tabla 3.1, descartamos la opción de usar plataformas interesantes y promisorias como: Cardano, Neo o Stellar, debido a que estas aún no han podido ser desarrolladas al nivel de tener en producción una versión comercial, por tanto, no resultan tener el atractivo necesario para ser parte de la PoC.

Como resultado se obtiene que las siguientes plataformas son candidatas para el desarrollo de la PoC del uso de Smart Contract:

1. Bitcoin
2. NXT
3. Counterparty

4. Monax
5. Ethereum

A continuación, se muestra un análisis de las restantes plataformas, además de Stellar:

Platform	Blockchain			Contract Language	Total Tx	Volume (K USD)	Marketcap (M USD)
	Type	Size	Block int.				
Bitcoin	Public	96 GB	10 min.	Bitcoin scripts + signatures	184,045,240	83,178	15,482
Counterparty				EVM bytecode	12,170,386	33	4
Ethereum	Public	17-60 GB	12 sec.	EVM bytecode	14,754,984	10,354	723
Stellar	Public	?	3 sec.	Transaction chains + signatures	?	35	17
Monax	Private	?	Custom	EVM bytecode + permissions	?	n/a	n/a
Lisk	Private	?	Custom	JavaScript	?	45	15

Tabla 3.2: Comparación de plataformas restantes. [34]

Como resultado del estudio presentado en la tabla 3.2, podemos observar que entre los Blockchain de tipo público, existe una enorme diferencia (más de 10 veces) en la cantidad de transacciones en Bitcoin que Counterparty o Ethereum, esto se da debido a que Bitcoin posee la criptomoneda líder en el mercado y como se mencionó en el capítulo anterior no se encuentra focalizado en Smart Contracts y por ende no permite el desarrollo de contratos de complejos, además el tiempo de creación de bloque es sensiblemente mayor al resto de las opciones, lo cual presenta una clara desventaja para esta plataforma. Otra desventaja de Bitcoin y los sistemas basados en su Blockchain es que el mecanismo de consenso que utiliza es PoW, el cual permite que los mineros que poseen mayor poder computacional sean participes de la mayor parte del minado, lo cual desalienta a los pequeños mineros y otros nodos a mantener la red Blockchain, la cual terminaría por centralizarse en pocos nodos al no permitir una mayor competencia entre ellos.

Por tanto, se elimina a Bitcoin como opción para el desarrollo de Smart Contracts, así como los sistemas basados en su Blockchain como el caso de Counterparty.

En el caso de Monax y Lisk, como se muestra en la tabla 3.2, ambas plataformas son de tipo Blockchain privado, lo cual dificulta el requerimiento que cualquier persona o entidad pueda auditar el contrato. Como consecuencia se descarta estas opciones y se elige por tanto a la plataforma Ethereum para su uso.

3.3. Requerimientos

Para desarrollar un Smart Contract en la plataforma Ethereum, es necesario tener un monedero con una cuenta que posea ether, un compilador para verificar el código, y un medio para conectarse a la red y desplegar el contrato. Para cumplir estos requerimientos se utilizará las siguientes herramientas:

- MetaMask
- IDE Remix

3.3.1. MetaMask

Es una extensión o complemento para navegadores web que funciona con Google Chrome, Opera y Firefox.

Cumple la función de enlace, como se muestra en la figura 3.2, entre los navegadores y Ethereum, así como de billetera, la cual permite la interacción con otras billeteras como MyEtherWallet.

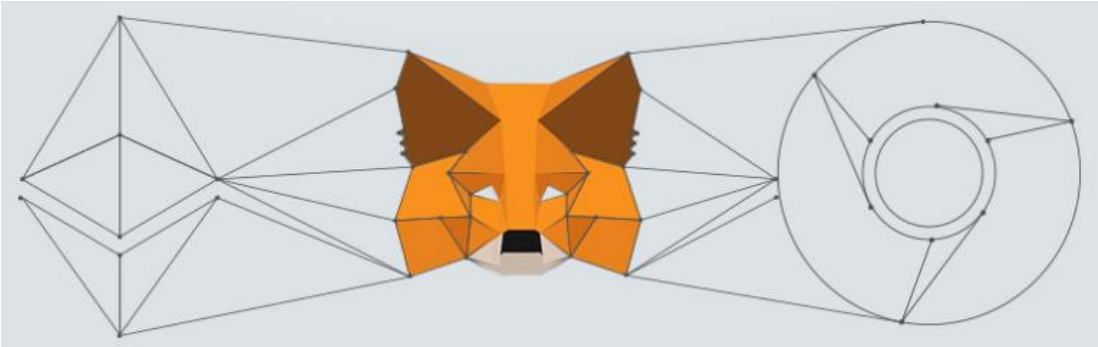

Figura 3.2: Logo de MetaMask y su interacción entre navegadores y Ethereum. [66]

MetaMask es capaz de interactuar con activos de distintos formatos como ERC-20, ya que permite añadir tokens personalizados en este formato. Entre las ventajas de MetaMask están:

- Práctico: En lugar de administrar varias claves privadas, los usuarios sólo necesitan recordar una lista de palabras y las transacciones se firman en su nombre. [67]
- Ahorra espacio: Los usuarios no tienen que descargar el bloque Ethereum completo, ya que envía peticiones a nodos fuera del ordenador del usuario. [67]
- Integrado: Las DApps (Aplicaciones Descentralizadas) están diseñados para trabajar con él, por lo que resulta mucho más fácil enviar y recibir Ether. [67]

Lamentablemente esta comodidad tiene su costoso, el cual es aceptado por los usuarios de esta extensión, entre las principales desventajas se puede mencionar:

- Terceros: MetaMask mantiene claves privadas en el navegador del usuario. Esto es menos seguro que una billetera digital tradicional, es decir, aquellas que no se instalan como extensión al navegador.

- Nodos externos: En lugar de ser un nodo completo, se basa en nodos externos que a veces tienen un tiempo de inactividad que puede hacer que MetaMask deje de funcionar o demore su accionar. [67]

Por último, una funcionalidad interesante de MetaMask, es permitir la interacción con las diferentes redes de pruebas o testnet de Ethereum, como se puede observar en la figura 3.3. Lo cual permite que MetaMask sea una herramienta útil para los desarrolladores.

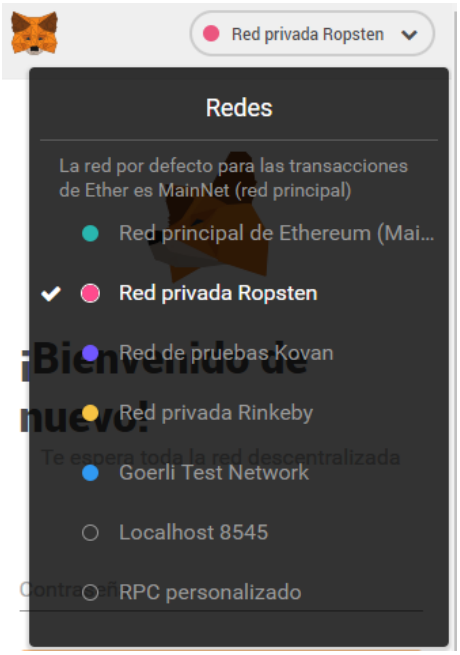

Figura 3.3: Redes con las cuales interactúa MetaMask. [66]

3.3.2. IDE Remix

Es una herramienta de código abierto que funciona como IDE (Entorno de Desarrollo Integrado), permite la escritura y compilación de contratos en lenguaje Solidity directamente desde el navegador o localmente, descargando la herramienta desde GitHub. Remix también soporta pruebas, depuración y despliegue de contratos inteligentes.

Se puede acceder tanto a proyectos en uso como anteriores, además posee varias versiones de compilación de Solidity por lo que es posible intentar compilar un contrato en una versión antigua o reciente.

En la PoC se utilizará la Testnet Ropsten de Ethereum, la cual es una de las Testnet más populares, además posee la gran ventaja de poder usarse de manera gratuita y soporta las mismas funcionalidades que Ethereum, Ropsten es empleada por los desarrolladores para probar la validez de los Smart Contracts en la plataforma.

3.4. Desarrollo

Utilizando el código enunciado en el Anexo, se logró llevar a cabo la PoC planteada en el punto 3.1.

Para iniciar se compiló el código utilizando la versión de 0.5.11 como se muestra en la figura 3.4, la cual no posee vulnerabilidades conocidas. [68]

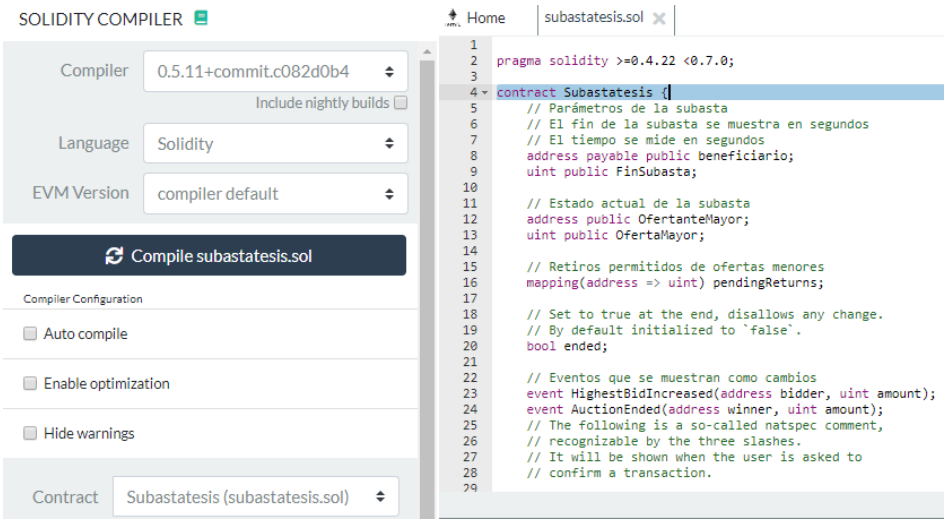

Figura 3.4: Código compilado.

Para el despliegue del contrato compilado en el párrafo anterior, se debe pagar una tasa por esta transacción según lo ilustrado en la figura 3.5.

Debido a las condiciones solicitadas por el contrato, este solicita el tiempo que va a durar la subasta y la dirección del beneficiario, cabe destacar que, de acuerdo a lo programado en el código, el contrato puede ser creado por el mismo beneficiario o por un tercero.

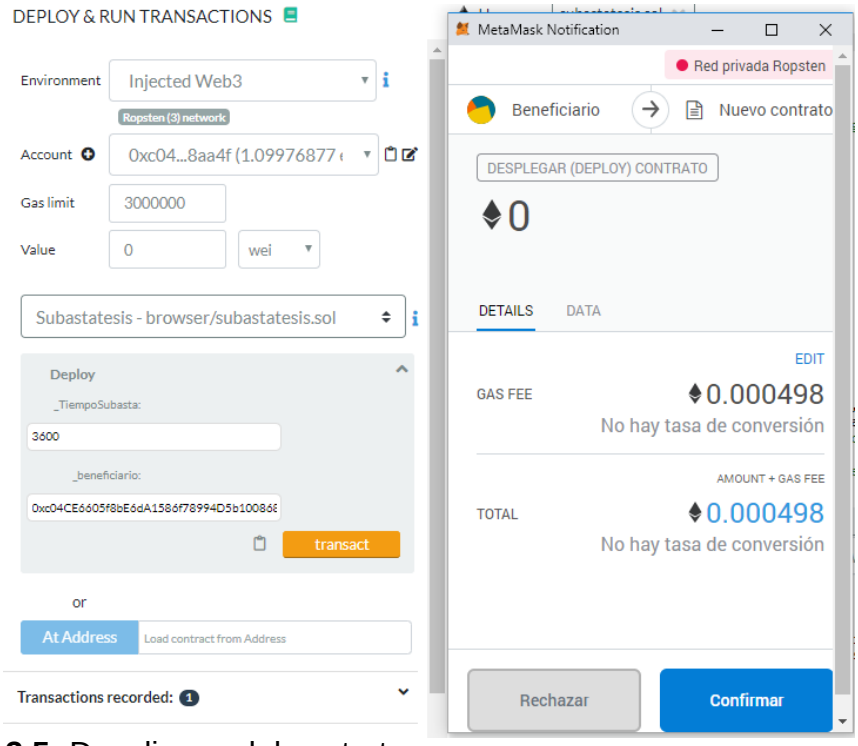

Figura 3.5: Despliegue del contrato.

Una vez que se encuentra creado el contrato, los ofertantes comienzan a interactuar con el mismo.

Antes de ofertar en la subasta, el primer postor verifica cuál es la mayor puja hasta el momento, la cual es de 0 de acuerdo a lo demostrado en la figura 3.6 y luego procede a ofertar 0.1 ether.

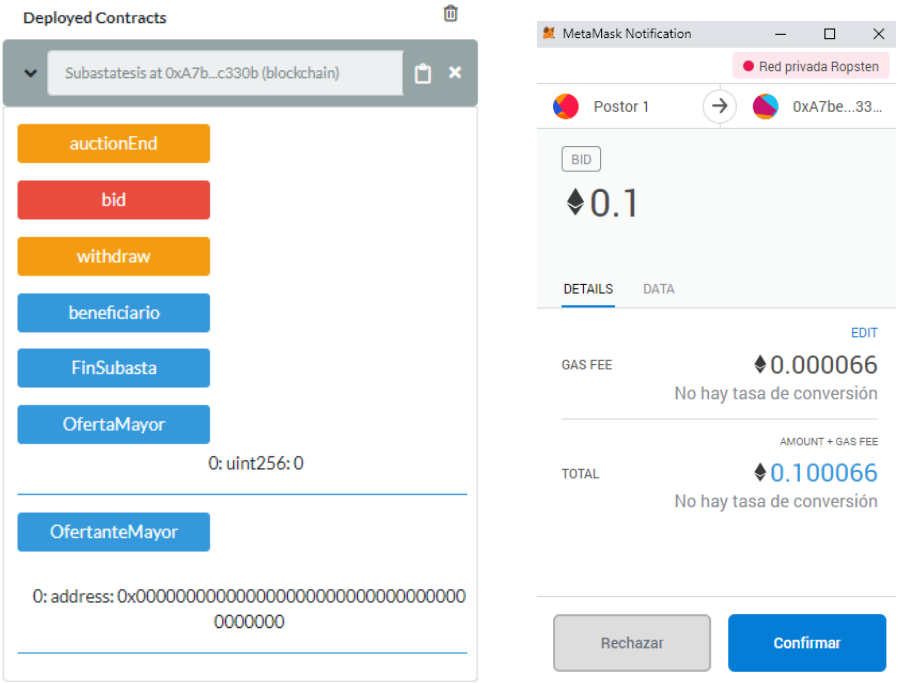

Figura 3.6: Verificación de oferta Mayor y primera puja en la subasta transacción “bid”.

A continuación, un segundo postor decide hacer una puja, para lo cual al igual que el primero verifica cuál es la mayor oferta hasta el momento y luego decide proceder a ofertar, esta vez según lo señalado en la figura 3.7 aumenta el valor de la máxima oferta a 1 ether.

Debido a las exigencias del protocolo, cada transacción en la red tiene un costo lo cual se refleja en todas las ofertas, este costo de transacción no aplica a consultas dentro del contrato como la realizada por ambos ofertantes para conocer el valor de máxima puja.

A pesar que en la figura 3.7, se realiza una transacción de mayor valor que en la figura anterior, los costos de transacción disminuyen, esto se explica debido a que el protocolo Ethereum, verifica varios factores para la definición de costos de transacción, como el nivel de ocupación de la red, entre otros. [57]

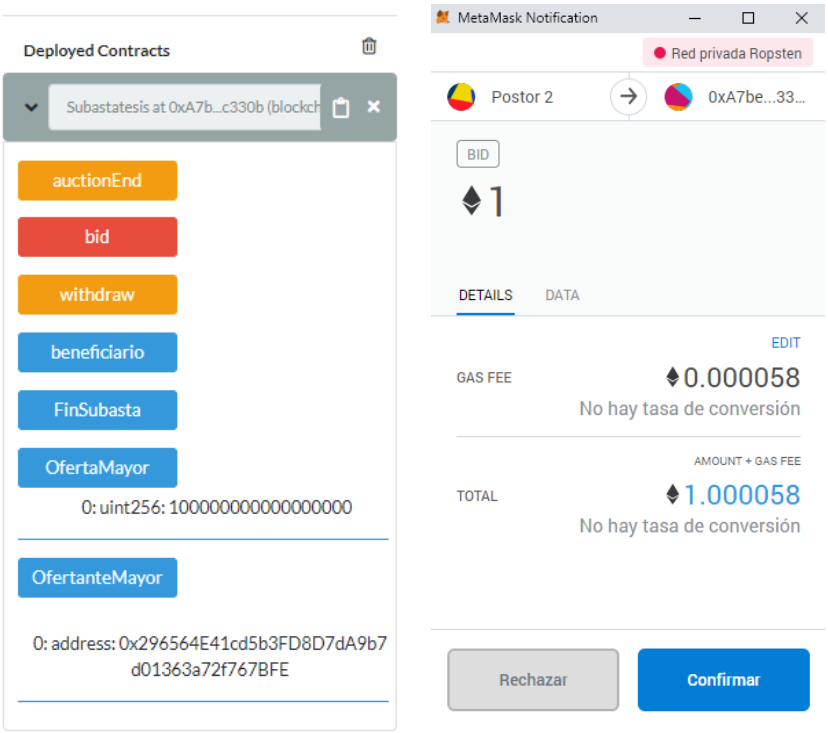

Figura 3.7: Verificación de oferta Mayor y segunda puja en la subasta.

El beneficiario verifica las mayores apuestas, y decide cerrar la subasta, antes de que la misma se cierre de acuerdo al tiempo configurado en el despliegue. Debido a que el proceso de cierre no requiere una oferta, no es necesario pagar más del costo de la transacción, como se muestra en la figura 3.8.

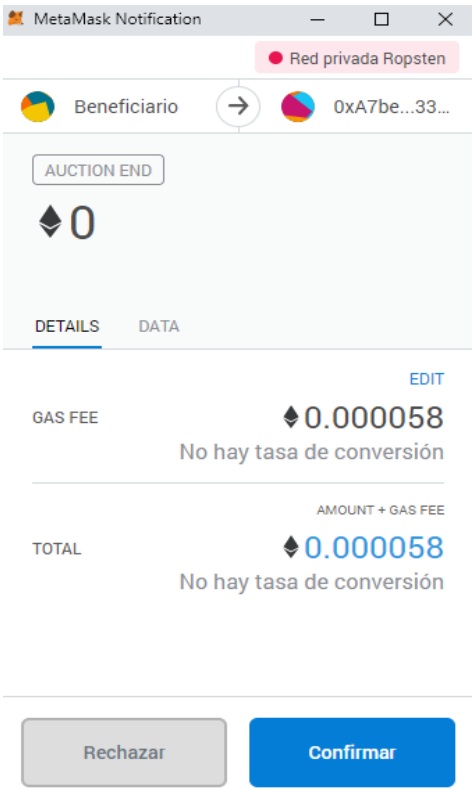

Figura 3.8: Cierre de subasta “Auction End”.

Para corroborar que la subasta se realizó de forma adecuada y que el beneficiario pudo obtener la máxima oferta, se puede realizar una comparativa entre el saldo del beneficiario previo a la subasta y posterior a ella, a continuación, en las figuras 3.9 y 3.10 se muestra como la mayor puja se acredita al beneficio.

Figura 3.9: Saldo del beneficiario previo a la subasta.

Figura 3.10: Saldo del beneficiario posterior a la subasta.

Finalmente, el Postor 1 verifica si ganó la subasta, al ver que no fue así retira el ether de su apuesta de acuerdo a lo ilustrado en la figura 3.11.

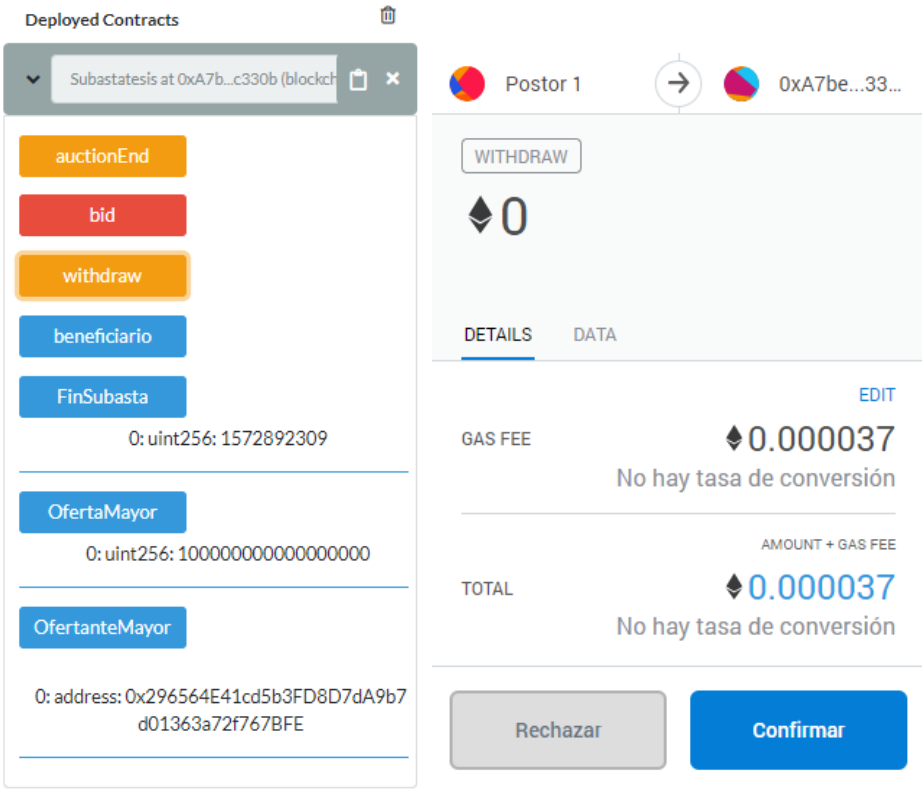

Figura 3.11: Transacción de retiro “withdraw” de la oferta menor.

Para verificar que el Postor 1 pudo recuperar el valor de su oferta, se puede realizar una comparativa de igual forma que en el caso del beneficiario, entre el saldo previo a la subasta (figura 3,12) y posterior a ella (figura 3.13).

Figura 3.12: Saldo del Postor 1 previo a la subasta.

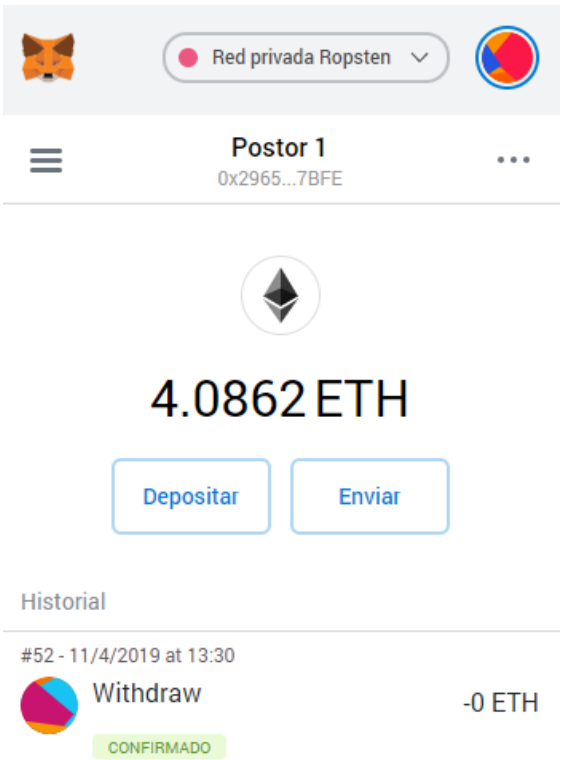

Figura 3.13: Saldo del Postor 1 posterior a la subasta.

3.5. Resultados

Como resultado de la subasta se puede observar que los participantes de la misma lograron desempeñar su rol de manera exitosa, los postores lograron ofertar, el beneficiario se quedó con el valor de la puja mayor, el Postor 1 logró recuperar el valor de su apuesta.

Para el beneficiario el costo del despliegue del Smart Contract y la obtención del dinero del mismo rondó el 0.05% del valor de la máxima oferta, puesto que las figuras 3.5 y 3.8 muestran que el costo total fue 0.000556 Ether.

En el caso de los ofertantes, la PoC demostró que el sistema permite mejorar las ofertas sin necesidad de grandes aumentos de costos por transacción inclusive en las figuras 3.6 y 3.7 se muestra como a pesar de incrementar diez veces el valor de la oferta anterior, el costo de transacción fue menor.

Para finalizar este trabajo como comentario final, se recomienda que en el caso de querer realizar una subasta tomar en consideración la opción de ejecutarla mediante Smart Contracts. La PoC mostró que los costos de transacción son despreciables con relación a las ofertas, lo cual motiva a los postores a realizar mayores pujas. En la primera puja se ofertó cerca $16 USD ó $959 ARS[1] a un costo cercano a $0.01 USD ó $0.58 ARS, mientras que en la segunda oferta el valor fue cerca de $160 USD ó $9595 ARS, con aproximadamente el mismo coste de la puja anterior.

[1] Valores de cotización obtenidos de la página https://es.ratesviewer.com/ al 25 de ene. de 2020

Conclusiones

Una vez concluida la tesis de maestría, se pueden obtener las siguientes conclusiones:

- Existen varias plataformas que prometen desbancar a Ethereum como estándar de facto en el desarrollo de Contratos Inteligentes, una de ellas es Cardano que promete mucho por algunas de sus innovadoras características, así como el gran equipo de expertos que se encuentra conformado para su desarrollo, sin embargo, existe la posibilidad que el letargo de su puesta en producción, produzca un acontecimiento similar a la historia de la distribución de teclado Dvorak, el cual a pesar de las evidentes ventajas que posee frente al teclado Qwerty, nunca pudo desplazarlo como el estándar *de facto* en las distribuciones de teclado.
- A medida que se populariza la tecnología Blockchain, mejoran los procesos en las plataformas desarrolladas en torno a esta tecnología. Como es el caso del reemplazo de la Prueba de Trabajo con la Prueba de Participación, la cual reduce el impacto energético de la generación de un nuevo bloque en una red Blockchain.
- Las plataformas con base en Blockchain pueden ser desarrolladas no sólo con fines comerciales, se pueden acoplar en ámbitos empresariales los cuáles tienen mayores restricciones en la privacidad de los datos utilizados.
- Los Smart Contracts son en la actualidad uno de los desarrollos más importantes basados en Blockchain, una prueba simple pero efectiva de ello, es que el Ether la moneda de la plataforma de Contratos Inteligentes Ethereum es una de las criptodivisas más importantes del mercado. [69]
- Los Contratos inteligentes sin duda jugarán un papel preponderante dentro del mercado de software para soluciones comerciales, debido a factores como: bajos costos por transacción (aproximadamente de $0.01 USD utilizando Ether[2]), auditabilidad, escalabilidad y adaptación a distintas necesidades de privacidad.
- Debido a la versatilidad de aplicaciones prácticas de los Smart Contracts, las plataformas que podrán competir de mejor forma con Ethereum serán posiblemente aquellas que busquen una especialización de sus objetivos, así como mejoren la facilidad de su uso con los usuarios. Un ejemplo de esto es el caso de Monax, que promete ser una solución “Plug and Play” para contratos legales.

[2] Valor de cotización obtenido de la página https://es.ratesviewer.com/ al 25 de ene. de 2020

Glosario

Hash: Función matemática de una sola vía la cual independientemente del valor ingresado entrega una salida con la misma de longitud, las cuales idealmente no poseen relación con la entrada.

Lenguaje de alto nivel: Es aquel cuyas instrucciones son más cercanas al lenguaje humano que al de las computadoras (binarios), por ello necesitan un interpretador que puede ser un compilador, algunos ejemplos son: C++, Python, Java, etc.

Lenguaje Turing completo: Es aquel lenguaje de programación capaz de realizar cualquier cálculo computacional con los recursos adecuados (memoria, capacidad de procesamiento, etc.).

Libro Mayor: En una red Blockchain el libro mayor es el registro de todas las transacciones realizadas en la misma, el cual generalmente se encuentra fraccionado entre los nodos de la red.

Opcode: Porción de la instrucción de lenguaje de máquina que define la operación a realizar.

On Premise: En informática un sistema o parte de él (software, hardware o una combinación de ambos) se califica de esta forma cuando se aloja y se gestiona de manera local por la entidad o empresa a la cual pertenece.

Anexo

En este punto se encuentra el código empleado para la solución de la prueba de concepto planteada en el punto 3.1, se basa en los casos planteados en el repositorio: https://github.com/magonicolas/Ethereum-Solidity

```
pragma solidity >=0.4.22 <0.7.0;

contract Subastatesis {
    // Parámetros de la subasta
    // El fin de la subasta se muestra en segundos
    // El tiempo se mide en segundos
    address payable public beneficiario;
    uint public FinSubasta;

    // Variables para mostrar el estado actual de la subasta
    address public OfertanteMayor;
    uint public OfertaMayor;

    // Retiros de ofertas menores
    mapping(address => uint) pendingReturns;

    // Bandera utilizada para terminar la subasta
    bool ended;

    // Incremento de apuesta y final de subasta
    event HighestBidIncreased(address bidder, uint amount);
    event AuctionEnded(address winner, uint amount);

    // Creación de la subasta
    constructor(
        uint _TiempoSubasta,
```

```
        address payable _beneficiario
    ) public {
        beneficiario = _beneficiario;
        FinSubasta = now + _TiempoSubasta;
    }


    modifier onlyOwner() {
        require (msg.sender == beneficiario);
        _;
    }

    // Función de apuesta
    function bid() public payable {

        // Verifica que la subasta tenga vigencia
        require(
            now <= FinSubasta,
            "Auction already ended."
        );

        // Verifica que la oferta realizada sea mayor que la mayor apuesta
actual
        require(
            msg.value > OfertaMayor,
            "There already is a higher bid."
        );

        // Registra la mayor oferta
        if (OfertaMayor != 0) {
```

```
            pendingReturns[OfertanteMayor] += OfertaMayor;
        }
        OfertanteMayor = msg.sender;
        OfertaMayor = msg.value;
        emit HighestBidIncreased(msg.sender, msg.value);
    }

    // Retiro de dinero
    function withdraw() public {
        uint amount = pendingReturns[msg.sender];
        pendingReturns[msg.sender] = 0;
        msg.sender.transfer(amount);
    }

    // Función de fin de subasta
    function auctionEnd() public {

        // Verifica que no se cumpla el tiempo límite
        require(now >= FinSubasta, "Auction not yet ended.");
        require(!ended, "auctionEnd has already been called.");

        ended = true;
        emit AuctionEnded(OfertanteMayor, OfertaMayor);

        // Se transfiere al beneficiario la mayor oferta
        beneficiario.transfer(OfertaMayor);
    }
}
```

Bibliografía específica

[1] M. A. a. A. v. Moorsel, BLOCKCHAIN-BASED SMART CONTRACTS: A SYSTEMATIC MAPPING STUDY, Newcastle: Newcastle University, 2017.

[2] Pro-universitarios, «Pro-universitarios,» 7 Mayo 2018. [En línea]. Available: http://pro-universitarios.com/como-funciona-blockchain/. [Último acceso: 4 Febrero 2019].

[3] T. Geek, «6 Major Features Of Blockchain,» 22 Noviembre 2018. [En línea]. Available: https://medium.com/@techgeek628/6-major-features-of-blockchain-why-blockchain-is-popular-59f7a65b6698. [Último acceso: 5 Febrero 2019].

[4] W. i. t. m. i. f. o. b. technology?, «Quora,» 14 Noviembre 2018. [En línea]. Available: https://www.quora.com/What-is-the-most-important-feature-of-blockchain-technology. [Último acceso: 5 Febrero 2019].

[5] H. Anwar, «101 Blockchains,» 24 Mayo 2018. [En línea]. Available: https://101blockchains.com/introduction-to-blockchain-features/. [Último acceso: 5 Febrero 2019].

[6] T. K. Sharma, «Blockchain Council,» 3 Noviembre 2018. [En línea]. Available: https://www.blockchain-council.org/blockchain/how-blockchain-is-solving-the-problem-of-double-spending-in-the-finance-sector/. [Último acceso: 5 Febrero 2019].

[7] Lisk, [En línea]. Available: https://lisk.io/academy/blockchain-basics/how-does-blockchain-work/nodes. [Último acceso: 6 Febrero 2019].

[8] «PluralSight,» Blockchain Architecture, 10 Enero 2019. [En línea]. Available: https://www.pluralsight.com/guides/blockchain-architecture. [Último acceso: 6 Febrero 2019].

[9] M. A. K. y. K. Salah, «IoT Security: Review, Blockchain Solutions, and Open Challenge,» Khalifa University, 19 Noviembre 2017. [En línea]. Available: https://www.researchgate.net/figure/Blockchain-design-structure-showing-chained-blocks-with-header-and-body-fields_fig2_321017113. [Último acceso: 6 Febrero 2019].

[10] Y. C. E. B. y. G. C. Ronghua Xua, «An Exploration of Blockchain Enabled Decentralized Capability based Access Control Strategy for Space Situation Awareness,» Binghamton University, Binghamton, 2018.

[11] Blockgeeks, «Proof of Work vs Proof of Stake: Basic Mining Guide».

[12] «P2PF Wiki,» 28 Diciembre 2018. [En línea]. Available: http://wiki.p2pfoundation.net/Proof_of_Work. [Último acceso: 10 Febrero 2019].

[13] F. Z. y. M. A. Peter Kovary, «Blockchain - Blueprint for a new economy,» [En línea]. Available: https://www.doc.ic.ac.uk/~ma7614/topics_website/tech.html. [Último acceso: 5 Febrero 2019].

[14] D. Cosset, «DEV,» 5 Enero 2018. [En línea]. Available: https://dev.to/damcosset/blockchain-what-is-mining-2eod. [Último acceso: 10 Febrero 2019].

[15] L. Shiff, «Bmc Blogs,» The Business of IT Blog, 1 Octubre 2018. [En línea]. Available: https://www.bmc.com/blogs/public-vs-private-blockchain/. [Último acceso: 11 Febrero 2019].

[16] K.-L. B. A. D. E. T. y. E. B. H. Omar Dib, «Consortium Blockchains: Overview, Applications and Challenges,» *International Journal on Advances in Telecommunications,* vol. 11, pp. 51-64, 2018.

[17] «Blockchain,» Blockchains & Distributed Ledger Technologies, [En línea]. Available: https://blockchainhub.net/blockchains-and-distributed-ledger-technologies-in-general/. [Último acceso: 11 Febrero 2019].

[18] L. B. News, «Hackernoon,» 1 Agosto 2018. [En línea]. Available: https://hackernoon.com/blockchain-architecture-analysis-private-vs-public-vs-consortium-65eb061b907b. [Último acceso: 12 Febrero 2019].

[19] Applicature, «Applicature,» 19 Abril 2018. [En línea]. Available: https://medium.com/applicature/the-top-5-mistakes-in-blockchain-project-implementation-770b17064d11. [Último acceso: 12 Febrero 2019].

[20] J. Frankenfield, «Investopedia,» 17 Mayo 2018. [En línea]. Available: https://www.investopedia.com/terms/b/blockchainasaservice-baas.asp. [Último acceso: 13 Febrero 2019].

[21] A. Patrizio, «Datamation,» 27 Marzo 2018. [En línea]. Available: https://www.datamation.com/data-center/top-10-blockchain-as-a-service-providers.html. [Último acceso: 2 Febrero 2019].

[22] «R3,» The Corda Platform, [En línea]. Available: https://www.r3.com/corda-platform/. [Último acceso: 13 Febrero 2019].

[23] A. Kumar, «SAP,» 28 Junio 2018. [En línea]. Available: https://blogs.sap.com/2018/06/28/introduction-to-blockchain-and-sap-cloud-platform-blockchain-service/. [Último acceso: 13 Febrero 2019].

[24] «Betalist,» Rubix Core, [En línea]. Available: https://betalist.com/startups/rubixcore. [Último acceso: 13 Febrero 2019].

[25] P. Altimore, «Microsoft Azure,» Azure Blockchain Workbench architecture, 13 Enero 2019. [En línea]. Available: https://docs.microsoft.com/en-us/azure/blockchain/workbench/architecture. [Último acceso: 13 Febrero 2019].

[26] M. Ament, «Hackernoon,» 8 Septiembre 2017. [En línea]. Available: https://hackernoon.com/blockchain-first-1bdbe39d7778. [Último acceso: 13 Febrero 2019].

[27] S. Patterson, «Openbazaar.zendesk,» 2018. [En línea]. Available: https://openbazaar.zendesk.com/hc/en-us/articles/208020193-What-is-OpenBazaar-. [Último acceso: 13 Febrero 2019].

[28] A. b. Bit2me. [En línea]. Available: https://academy.bit2me.com/que-es-openbazaar/. [Último acceso: 13 Febrero 2019].

[29] «Antiguas Civilizaciones,» 2013. [En línea]. Available: http://anarueda4.wixsite.com/a-civilizaciones/sociedad-y-economa-en-babilonia. [Último acceso: 27 Enero 2019].

[30] A. O. C. d. Albornoz, Derecho privado romano, Malaga: Promotor cultural Malagueña, 1999.

[31] *Ley 25.506,* Buenos Aires, 2001.

[32] Tendencias 21, 2016 Octubre 7. [En línea]. Available: https://www.tendencias21.net/Un-novedoso-sistema-permite-firmar-contratos-con-la-voz_a43288.html. [Último acceso: 27 Enero 2019].

[33] E. W. y. G. Malm, «Using Blockchain Technology and Smart Contracts to Create a Distributed Securities Depository,» Lund University, Lund, 2016.

[34] M. B. y. L. Pompianu, «An empirical analysis of smart contracts: platforms, applications, and design patterns,» Universidad de Cagliari, Cagliari, 2017.

[35] K. Ray, «Coin Central,» 29 Agosto 2018. [En línea]. Available: https://coincentral.com/what-is-a-smart-contract/. [Último acceso: 19 Febrero 2019].

[36] «BlockchainHub,» [En línea]. Available: https://blockchainhub.net/smart-contracts/. [Último acceso: 19 Febrero 2019].

[37] M. Contreas, «Clipset,» Augur, la casa de apuestas 'blockchain' sobre muertes de famosos, 2018. [En línea]. Available: https://clipset.20minutos.es/augur-mercado-blockchain-asesinatos/. [Último acceso: 26 Febrero 2019].

[38] T. Burger, «Crypthor.net,» WHAT ARE SMART CONTRACTS? HOW DO THEY WORK?, 3 Octubre 2017. [En línea]. Available: http://www.crypthor.net/smart-contracts-work/. [Último acceso: 26 Febrero 2019].

[39] S. Kishor, «DZone,» 11 Enero 2017. [En línea]. Available: https://dzone.com/articles/the-bitcoin-protocol-how-it-works. [Último acceso: 26 Marzo 2019].

[40] BTC Studios, «Bitcoin Magazine,» [En línea]. Available: https://bitcoinmagazine.com/articles/yes-bitcoin-can-do-smart-contracts-and-particl-demonstrates-how/. [Último acceso: 25 Marzo 2019].

[41] Bitcoin Magazine, «Bitcoin Magazine,» 20 Mayo 2015. [En línea]. Available: https://bitcoinmagazine.com/articles/nxt-original-bitcoin-2-0-platform-smart-contracts-decentralized-crowdfunding-open-source-18-months-development-1432169550/. [Último acceso: 4 Marzo 2019].

[42] «nxter magazine,» [En línea]. Available: https://www.nxter.org/developers/#code_examples. [Último acceso: 4 Marzo 2019].

[43] Counterparty, «Counterparty,» [En línea]. Available: https://counterparty.io/docs/faq-smartcontracts/. [Último acceso: 22 Marzo 2019].

[44] BigChain DB, «BigChain DB,» [En línea]. Available: https://www.bigchaindb.com/partners/monax/. [Último acceso: 21 Marzo 2019].

[45] R. Davidson, «Monax,» 4 Diciembre 2018. [En línea]. Available: https://monax.io/blog/2018/12/04/introducing-the-monax-platform---contract-lifecycle-management-for-the-digital-age/. [Último acceso: 21 Marzo 2019].

[46] M. Mulders, «Hackernoon,» 5 Marzo 2018. [En línea]. Available: https://hackernoon.com/comparison-of-smart-contract-platforms-2796e34673b7. [Último acceso: 20 Marzo 2019].

[47] «Icorating,» [En línea]. Available: https://icorating.com/pdf/56/1//Bd40ljAOmjaCFAXmkCj9NAKZAIUj1Dwb9v75AAZe.pdf. [Último acceso: 20 Marzo 2019].

[48] Lisk, «LIsk,» [En línea]. Available: https://lisk.io/academy/blockchain-basics/how-does-blockchain-work/delegated-proof-of-stake. [Último acceso: 21 Marzo 2019].

[49] S. Bowden, «All Crypto,» 8 Marxo 2018. [En línea]. Available: https://www.allcrypto.com/analysis/cardano-future-smart-contracts/. [Último acceso: 3 Marzo 2019].

[50] J. Wall, «Invest in Blockchain,» 12 Diciembre 2018. [En línea]. Available: https://www.investinblockchain.com/smart-contract-development-cardano-blockchain/. [Último acceso: 12 Marzo 2019].

[51] Emurgo, «Good Audience,» 3 Septiembre 2018. [En línea]. Available: https://blog.goodaudience.com/marlowe-financial-smart-contracts-in-cardano-4ebb8fd94e24. [Último acceso: 12 Marzo 2019].

[52] Ethereum Community, «Ethereum Homestead,» 2016. [En línea]. Available: http://ethdocs.org/en/latest/introduction/what-is-ethereum.html#how-does-ethereum-work. [Último acceso: 3 Abril 2019].

[53] M. B. y. T. C. Nicola Atzei, «A survey of attacks on Ethereum smart contracts,» Universit`a degli Studi di Cagliari, Cagliari, 2016.

[54] K. Halunen, «Evaluating the Efficiency of Blockchains in IoT with Simulations,» Research Gate, 2017.

[55] N. Monkey, «Fast dot,» 16 Agosto 2018. [En línea]. Available: https://news.fastdot.com/getting-deep-into-evm-how-ethereum-works-backstage/. [Último acceso: 3 Abril 2019].

[56] M. M. y. C. S. Ilya Grishchenko, «A Semantic Framework for the Security Analysis of Ethereum Smart Contracts,» Springer, 2018.

[57] B. Asolo, «My Cryptopedia,» 1 Noviembre 2018. [En línea]. Available: https://www.mycryptopedia.com/ethereum-virtual-machine-explained/. [Último acceso: 3 Abril 2019].

[58] L. Hollander, «My Crypto,» 29 Enero 2019. [En línea]. Available: https://medium.com/mycrypto/the-ethereum-virtual-machine-how-does-it-work-9abac2b7c9e. [Último acceso: 3 Abril 2019].

[59] J. Rodríguez, «IHODL,» 2 Septiembre 2018. [En línea]. Available: https://es.ihodl.com/tutorials/2018-09-02/tokens-erc-20-erc-223-erc-721-y-erc-777-en-que-se-diferencian/. [Último acceso: 7 Abril 2019].

[60] «Blockgeeks,» Smart Contracts: The Blockchain Technology That Will Replace Lawyers, [En línea]. Available: https://blockgeeks.com/guides/smart-contracts/. [Último acceso: 25 Febrero 2019].

[61] L. Coleman, «CCN,» Smart Contracts: 12 Use Cases For Business And Beyond, 10 Febrero 2016. [En línea]. Available: https://www.ccn.com/smart-contracts-12-use-cases-for-business-and-beyond. [Último acceso: 26 Febrero 2019].

[62] A. Bhattacharya, «Hackenoon,» Smart Contracts—A Time Saving Primer, 29 Agosto 2018. [En línea]. Available: https://hackernoon.com/smart-contracts-a-time-saving-primer-b3060e3e5667. [Último acceso: 25 Febrero 2019].

[63] Cardozo Blockchain Project, «Cardozo Law,» 16 Octubre 2018. [En línea]. Available: https://cardozo.yu.edu/sites/default/files/Smart%20Contracts%20Report%20%232_0.pdf. [Último acceso: 14 Abril 2019].

[64] G. Tse, «Lexology,» Taylor Vinters Via LLC, 24 Septiembre 2018. [En línea]. Available: https://www.lexology.com/library/detail.aspx?g=f16551a7-e974-41d2-ba45-37e2dc7f6e41. [Último acceso: 14 Abril 2019].

[65] vasa, «Hackernoon,» 18 Julio 2018. [En línea]. Available: https://hackernoon.com/contractpedia-an-encyclopedia-of-40-smart-contract-platforms-4867f66da1e5. [Último acceso: 18 Mayo 2019].

[66] «Metamask,» [En línea]. Available: https://metamask.io/. [Último acceso: 5 Mayo 2019].

[67] M. Hussey, «Decrypt,» 16 Enero 2019. [En línea]. Available: https://decryptmedia.com/resources/metamask. [Último acceso: 18 Mayo 2019].

[68] 28 05 2019. [En línea]. Available: https://github.com/ethereum/solidity/blob/develop/docs/bugs_by_version.json. [Último acceso: 3 06 2019].

[69] N. Reiff, «Investopedia,» 8 Enero 2020. [En línea]. Available: https://www.investopedia.com/tech/most-important-cryptocurrencies-other-than-bitcoin/. [Último acceso: 25 Enero 2020].

Bibliografía General

- Smart Contracts o Contratos Inteligentes – miethereum.com
- https://docs.neo.org/en-us/sc/introduction.html
- https://www.stellar.org/