



UNIVERSIDAD DE JAÉN

Trabajo Fin de Máster

Aproximación a la Computación Distribuida desde la Tecnología de Cadena de Bloques: Un Caso de Uso de Aplicación Descentralizada

ALUMNO:

Mohamed Issam Adnane

TUTOR/Co-TUTOR:

Dr. D. José Santamaría López / Dr. D. Francisco
de Paula Roca Rodríguez

(Departamento de Informática / Departamento de Matemáticas)

Junio, 2022





Universidad de Jaén

Departamento de Informática

El Dr. D. **José Santamaría López** y el Dr. D. **Francisco de Paula Roca Rodríguez** en calidad de, respectivamente, tutor y co-tutor del Proyecto Fin de Máster titulado: ***Aproximación a la Computación Distribuida desde la Tecnología de Cadena de Bloques: Un Caso de Uso de Aplicación Descentralizada***, que presenta **Mohamed Issam Adnane**, autorizan su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

En Jaén, a 22 de Junio de 2022

El alumno:

A handwritten signature in black ink, appearing to read "Mohamed Issam Adnane".

El tutor:

El co-tutor:

RESUMEN

En esta memoria se presenta el Trabajo de Fin de Máster (TFM) titulado “*Aproximación a la Computación Distribuida desde la Tecnología de Cadena de Bloques: Un Caso de Uso de Aplicación Descentralizada*”, cuyo objetivo principal es introducir las bases de la tecnología de la cadena de bloques para a continuación detallar el desarrollo concreto de un caso de uso concreto de aplicación descentralizada (del inglés, decentralised App ó DApp) que hace uso de la tecnología de Blockchain basada en una arquitectura de microservicios. En concreto, en este TFM se presenta el desarrollo de una DApp para la gestión de una plataforma de venta inmobiliaria. Además, se propone el uso de un token de utilidad (MIA) específico para la DApp desarrollada.

Dedicado a:

*mis padres, que me han apoyado en todas mis decisiones,
a Nouhaila, mi gemela que me da fuerzas,
a mis amigos, que siempre están ahí,
y a mis tutores José y Francisco.*

Índice

1. Introducción	7
1.2. Motivación	7
1.3. Objetivos del TFM	8
1.4. Objetivos del Proyecto	8
1.5. Planificación Temporal	8
1.5.1. Estimación de tiempos	9
1.5.2. Diagrama de Gantt	11
1.6. Estimación de recursos y costes	12
1.7. Estructura de la memoria	13
2. Blockchain	14
2.1. Lo que no es Blockchain	15
2.2. Definiciones del Blockchain	15
2.3. Historia de Blockchain	15
2.4. Arquitectura de la Blockchain	17
2.4.1. Aplicaciones	19
2.4.2. Ledger descentralizado	19
2.5. Token	21
2.5.1. ¿Qué es un token?	21
2.5.1. Los diferentes tipos de tokens	22
3. Casos de uso	23
3.1. Casos de uso de blockchain en banca y finanzas	23
3.1.1. Pagos Internacionales	23
3.1.2. Los mercados de capitales	24
3.1.3. Financiación del comercio	26
3.1.4. Cumplimiento normativo y auditoría	26
3.1.5. Protección contra el lavado de dinero	26
3.1.6. Transacciones entre pares	26
3.2. Aplicaciones de blockchain en los negocios	27
3.2.1. Gestión de la cadena de suministro	27
3.2.2. Cuidado de la salud	27
3.2.3. Inamovible	28
3.2.4. Medios de comunicación	28
3.3. Aplicaciones de blockchain en el gobierno	28
3.3.1. Gestión de registros	28
3.3.2. Gestión de identidad	28

3.3.3. Votar	28
3.3.4. Impuestos	29
3.3.5. Agencias sin fines de lucro	29
3.3.6. Cumplimiento normativo/supervisión	29
3.4. Aplicaciones de blockchain en otras industrias	29
3.4.1. Gestión Financiera y Contabilidad	29
3.4.2. Gestión de registros	30
3.4.3. La seguridad cibernetica	30
3.4.4. Big Data	30
4. Estado del arte	31
4.1. Tipos de blockchain	31
4.1.1. Clasificación basada en la red	31
4.1.1.1. Blockchain pública	32
4.1.1.2. Blockchain privada	32
4.1.1.3. Blockchain (« autorizada »)	32
4.1.2. Clasificación basada en las aplicaciones derivadas	33
4.1.2.1. Blockchain 1.0: Criptomonedas	33
4.1.2.2. Blockchain 2.0: contratos inteligentes	34
4.1.2.3. Blockchain 3.0: futuras aplicaciones	35
4.2. Estructura de la Blockchain	36
4.2.1. La transacción	37
4.2.2. El bloque	40
4.3. La seguridad	41
4.3.1. Vulnerabilidades, amenazas, riesgos y ataques en la cadena de bloques	42
4.3.2. Mecanismos de seguridad	45
4.3.2.1. Criptografía	45
4.3.2.2. Criptografía simétrica	45
4.3.2.3. Criptografía asimétrica	46
4.3.2.4 Firma digital	47
4.3.2.5. Hash	47
4.3.3. Integridad de los datos	49
4.4. Operaciones básicas en la Blockchain	52
4.4.1. Validación de una transacción	52
4.4.2. Creación de un nuevo bloque (minería)	53
5. La blockchain de Ethereum	55
5.1. Cuentas, mensajes y transacciones.	56
5.1.1. Las cuentas	56
5.1.1.1. Estado de la cuenta	57
5.1.1.2. El estado general	58

5.1.2. Mensaje y Transacción	59
5.1.3. Ejecución de una transacción	61
5.1.3.1. Ejecutar una transacción de creación de contrato	63
5.1.3.2. Ejecución de una transacción de invocación de mensaje	64
5.2. El modelo de incentivos mineros.	65
5.2.1. Operación Minería y Seguridad	66
5.2.2. Operación Minería y Riqueza	66
5.3. Contratos inteligentes	68
5.3.1. Estructura de un contrato inteligente	69
5.3.2. Ejecución e implementación de un contrato inteligente	71
5.4. Dapps: aplicaciones descentralizadas	72
6. Caso de USO: Una DApp para la gestión inmobiliaria	74
6.1. Análisis y diseño de sistemas	75
6.1.1. Especificación de necesidades funcionales.	75
6.1.1.1. Diagrama de casos de uso	75
6.1.1.2. Diagrama de clase	78
6.1.2. Arquitectura del sistema	78
6.2. Implementación del sistema	79
6.2.1. Herramientas de desarrollo y tecnologías utilizadas.	79
6.2.2. Requisitos	82
6.2.3. Codificación de servicios	83
6.2.4. Arquitectura de microservicios	84
6.3. Descripción general del DAPP	86
6.3.1. Interfaces del DAPP	86
6.4. Token de la DApp desarrollada	92
7. Conclusión y trabajo futuro	97
7.1. Conclusión	97
7.2. Trabajos futuros	98
Referencias	99
Índice de figuras	104
Índice de Tablas	106

Capítulo 1

1. Introducción

La tecnología de la cadena de bloques (del inglés, Blockchain) [1] comienza a estar cada vez más integrada en numerosos ámbitos de la sociedad. Su enorme potencial y tecnología revolucionaria está entrando en la vida cotidiana de las personas. Su modelo seguro, descentralizado y público garantiza la confidencialidad y veracidad de la información sin la necesidad de terceros. Son numerosos también los protocolos de Blockchain en desarrollo en la actualidad. Uno de los más prometedores que desarrollan esta tecnología es *Ethereum*, del cual ya trataremos en profundidad más adelante en esta memoria. En particular, el objetivo de Ethereum es crear una cadena de bloques programable que cambie el modelo de Internet, con programas almacenados que sirvan como columna vertebral de futuras aplicaciones descentralizadas, DApps. Estos programas se denominan contratos inteligentes (del inglés, smart contracts). Tomaremos Ethereum como plataforma de Blockchain para el desarrollo posterior de nuestro caso de uso de DApp propuesto en este TFM.

1.2. Motivación

La Blockchain y las tecnologías que dependen de ella, como las DApps, NFT y Web3, están tomando el mundo por sorpresa por las posibilidades que éstas ofrecen. La Blockchain es ahora de interés para nuevos sectores de la economía, finanzas, banca, servicios, académicos, gubernamentales, entre otros.

En concreto, el sector inmobiliario no es muy progresista en términos de tecnología. En efecto, debe facilitar la gestión de los activos, intercambiar el libre sin intermediarios, la normalización de los métodos de trabajo y facilitar el arreglo financiero de las operaciones. Hay que tener en cuenta que algunos actores pueden verse afectados: notarios, agentes inmobiliarios, urbanistas, promotores, expertos inmobiliarios e inversores, etc.

1.3. Objetivos del TFM

Los objetivos principales de este TFM se pueden desglosar de la siguiente manera:

- Entender la blockchain y el desarrollo con Solidity.
- Desarrollar una DApp y una arquitectura de microservicios, con un Aplicación SPA/WPA que consume estos microservicios. En concreto, se adoptará la Blockchain de la red Ethereum. Dicha DApp permitirá la venta y compra de casas, usando contrato inteligente y Blockchain.
- Desarrollar un token asociado al Dapp.
- Despliegue del token.
- Manejar una cadena de bloques.

1.4. Objetivos del Proyecto

Los objetivos del Proyecto:

- Reducción de los costes de transacción y del tiempo.
- Apertura de la inversión inmobiliaria a un público amplio, ya sea compra o alquiler.
- Facilitación de la gestión inmobiliaria para particulares y para profesionales.
- Digitalización completa del sector, vinculada a la inteligencia artificial y a la Internet de las cosas.
- Analizar la posibilidad de digitalización completa del sector inmobiliario.
- Facilitar al estado el acceso a la propiedad ofrecida ya alquilada o vendida.
- Disminuir la estafa.

1.5. Planificación Temporal

Para lograr una correcta comprensión de la planificación temporal de este proyecto, se ha realizado una tabla con la estimación de tiempos de cada una de las tareas, acompañada de un diagrama de Gantt que permite comprobar visualmente el desarrollo a lo largo del tiempo.

1.5.1. Estimación de tiempos

Gracias a la siguiente lista, acompañada de una tabla, es posible contemplar de un vistazo las tareas en las que está dividido este proyecto, acompañado de la estimación inicial de la duración (en días y horas) de las tareas a desarrollar.

- **A0.** Análisis del estado del arte y selección de las herramientas.
- **A1.** Organizar
 - **A1.1** Crear un token asociado a la DApp.
 - **A1.2** Crear Diagrama caso de uso.
 - **A1.3** Crear diagrama de clase para describir la estructura del sistema modelando clases, atributos, operaciones y relaciones entre sus objetos.
 - **A1.4** Crear la maqueta del DAPP.
- **A2.** Instalación de las herramientas y preparación del ambiente del trabajo.
- **A3.** Smart Contract
 - **A3.1** Escribir el contrato: la base de aplicaciones de nuestra plataforma de venta de propiedades, que se distribuirá en la cadena de bloques.
 - **A3.2** Implementar y probar el contrato en una "Blockchain local".
- **A4.** Crear la base de datos.
- **A5.** Desarrollo de la DApp y creación de token MIA.
 - **A5.1** Backend.
 - **A5.2** Frontend.
- **A6.** Pruebas del Dapp.
- **A7.** Desarrollo de memoria.
- **A8.** Revisión y confección de la versión final.

Tarea	Estimación de tiempo en días	Estimación de tiempo en horas (6 horas/día)
A0	10	60
A1	22	132

A1.1	7	42
A1.2	4	24
A1.3	3	18
A1.4	8	48
A2	2	12
A3	22	132
A3.1	15	90
A3.2	7	42
A4	3	18
A5	42	252
A5.1	27	162
A5.2	15	90
A6	5	30
A7	20	120
A8	5	30
TOTAL 131 días y (6h/dia) 786 horas		

Tabla 1. Estimación temporal

1.5.2. Diagrama de Gantt

Para plasmar la planificación y programación de las tareas a lo largo del tiempo, se ha confeccionado un diagrama de Gantt, reflejando la dedicación establecida para cada una de las tareas:

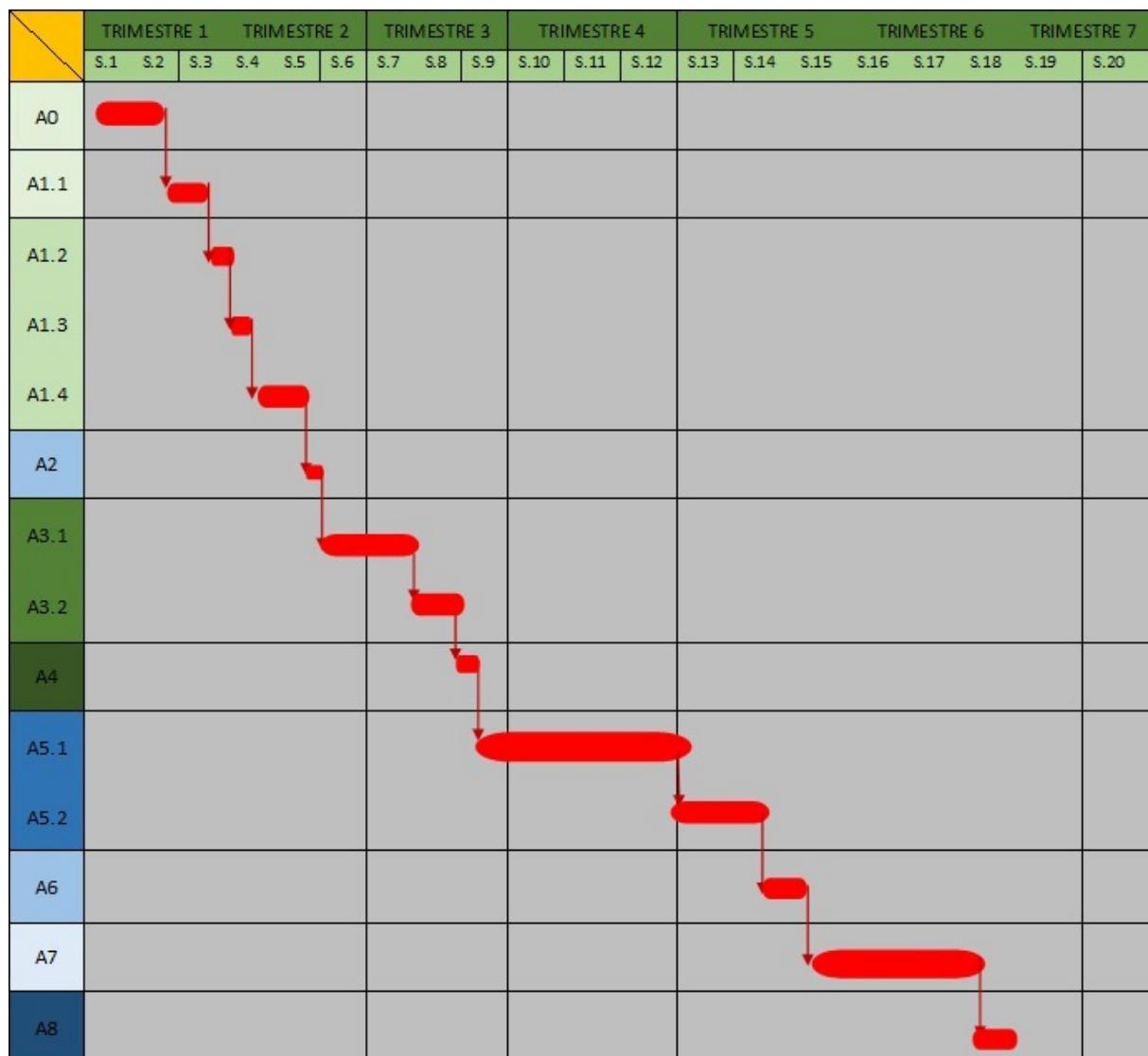


Figura 1. Diagrama de Gantt

1.6. Estimación de recursos y costes

En este apartado se detalla el presupuesto a tener en cuenta a la hora de la planificación del proyecto para su correcto desarrollo. Se presupuestan los gastos de todos los recursos usados en este trabajo para dar un presupuesto final.

Hardware				
Útil	Coste Bruto (€)	Tiempo de uso (meses)	Vida útil (meses)	Coste aplicado al uso (€)
Ordenador personal	1350	9	130	93.46 €
Disco SSD	120	6	130	5.53 €
Coste Total				98.99 €

Tabla 2. Presupuesto Hardware

Software				
Útil	Coste Bruto (€)	Tiempo de uso (meses)	Vida útil (meses)	Coste aplicado al uso (€)
web3j	0	7	-	0 €
truffle	0	4	-	0 €
ganache	0	4	-	0 €
MongoDB	0	8	-	0 €
Spring Tool Suite	0	9	-	0 €
Angular	0	8	-	0 €
Node	0	8	-	0 €
Coste Total				0 €

Tabla 3. Presupuesto Software

Costes Indirectos				
Útil	Coste Bruto (€)	Tiempo de uso (meses)	Vida útil (meses)	Coste aplicado al uso (€)
Electricidad	45	10	1	450 €
Internet	64,99	10	1	649,9 €
Coste Total				1099,9 €

Tabla 4. Costes Indirectos

1.7. Estructura de la memoria

La estructura de la presente memoria se ha dividido en las siguientes secciones:

- I. En el **capítulo 2** explicamos qué es la Blockchain, que no es, su historia y arquitectura.
- II. En el **capítulo 3** se muestran los diferentes casos de uso de la tecnología de blockchain en todas las industrias.
- III. En el **capítulo 4** se detalla el estado del arte de blockchain, sus tipos, estructura, seguridad y sus operaciones básicas.
- IV. En el **capítulo 5** se describen varias alternativas de blockchain, como Ethereum, así como las cuentas, mensajes, transacciones y los contratos inteligentes.
- V. En el **capítulo 6** se incluye el desarrollo particular de caso de uso de DApp propuesto en este TFM. Además, se describe el token de la DApp (MIA).
- VI. En el **capítulo 7** se incluyen las conclusiones y trabajos futuros derivados de este trabajo.
- VII. Por último, recogemos todas las referencias del presente documento.

Capítulo 2

2. Blockchain

Aunque la cadena de bloques tiene varias definiciones, la mejor manera de describirla es como una estructura de datos de bloques que se encadenan para formar una colección de registros, llamada libro de contabilidad, siendo la criptografía un ingrediente clave en el proceso. Una cadena de bloques no tiene un mecanismo de almacenamiento, sino un conjunto de protocolos que gobiernan cómo se falsifica la información. Así, una cadena de bloques puede almacenarse en archivos planos o en una base de datos.

La tecnología blockchain ha ganado popularidad porque su integridad no puede comprometerse fácilmente. Un blockchain comprometido puede ser reconocido por lo que es y rechazado con bastante facilidad por cualquier persona en una red. Esta integridad se consigue mediante la criptografía.

La promesa de Blockchain de proporcionar una integridad tan robusta es lo que finalmente allanó el camino para la idea de compartir cadenas de datos en redes peer-to-peer (P2P) no confiables. La validación de los bloques en una cadena de bloques es lo que garantiza que una cadena de bloques tenga un estado global válido que pueda ser aceptado por todos. Debido a la capacidad de una blockchain para compartir información en una red P2P abierta sin ninguna autoridad central que la gobierne, la tecnología puede tener muchas aplicaciones diferentes; sin embargo, la tecnología no podría simplemente desplegarse en estas aplicaciones inmediatamente sin ningún tipo de problema. Aunque la tecnología blockchain, desde el principio, tenía un enorme papel que desempeñar en la descentralización de las aplicaciones, todavía se enfrentaba a varios retos sobre su aplicación en entornos sin confianza. Uno de los mayores retos era mantener una cadena de bloques coherente entre todos los participantes de una red P2P. Esto se resolvió mediante la creación de un algoritmo de consenso, que acuerda cómo deben añadirse los bloques para hacer crecer la cadena en un entorno sin confianza.

El término blockchain conlleva varios conceptos, como la gestión de la red P2P, el mecanismo de consenso y otros, que contribuyen a la creación de una aplicación descentralizada.

2.1. Lo que no es Blockchain

Como acabamos de decir, a pesar de que la cadena de bloques es fascinante por su seguridad basada en la criptografía [20], su naturaleza descentralizada y su mecanismo de almacenamiento de datos casi inmutable, es muy importante entender sus limitaciones.

Aunque una red de cadenas de bloques es buena para manejar un estado global, no aportaría mucho valor cuando se trata de almacenar datos en masa, ya que habría problemas de escalabilidad. Es muy importante entender cuándo es mejor aplicar la tecnología blockchain para desarrollar una aplicación.

2.2. Definiciones del Blockchain

Una de las primeras cosas que hice en este capítulo fue señalar que existen múltiples definiciones de la palabra "blockchain". Antes de seguir en mi memoria, echemos un vistazo a varias definiciones de la palabra:

“La estructura de datos de la cadena de bloques es una lista de bloques ordenada y vinculada hacia atrás. “

- Andreas Antonopoulos, un popular evangelista de Bitcoin

“Una cadena de bloques es una forma específica o subconjunto de las tecnologías de libro mayor distribuido, que construye una cadena cronológica de bloques, de ahí el nombre de Blockchain “.

- Antony Lewis

2.3. Historia de Blockchain

En el año 1976, se publicó un documento sobre "New Directions in Cryptography" en el que se discutía el concepto de ledger distribuido. Con el avance en el campo de la criptografía, otro documento titulado "How to Time-Stamp a Digital Document" por Stuart Haber y Scott Stornetta, en el que se exponía el concepto de sellado de tiempo de los datos en lugar del medio. Otro concepto importante llamado "Dinero electrónico" o "moneda digital", que surgió a partir de un modelo propuesto por David Chaum también contribuyó al desarrollo del concepto de Blockchain que fue protocolos como los esquemas de e-cash que introdujeron la introdujeron la detección del doble gasto.

En 1997, Adam Back introdujo otro concepto llamado "hashcash" que ofrecía una solución para controlar los correos electrónicos basura. Esto condujo al concepto de crear dinero llamado "b-money" de Wei Dai, basado en la red peer to peer.

Satoshi Nakamoto es (o son, pues se especula con que es un equipo bajo un pseudónimo, aunque en el TFM nos referiremos como autor) considerado el inventor de la tecnología blockchain cuando publicó un artículo sobre bitcoin [1] en 2008 como "Bitcoin: Peer-to-Peer Electronic Cash System". El resumen del artículo trataba sobre el pago directo en línea de una fuente a otra sin depender de un tercero. El documento describe un sistema de pago electrónico basado en el concepto de criptografía. El documento de Nakamoto ofrecía una solución al doble donde una moneda digital no puede ser duplicada, y nadie puede gastar más de una vez. El documento estableció el concepto de libro de contabilidad público, donde el historial de transacciones de una moneda electrónica puede ser rastreado y confirmado si la moneda no ha sido y evitar el doble gasto.

Un programa de código abierto para implementar el sistema bitcoin fue meses después que la primera red de bitcoin se iniciara, a principios de 2009 cuando Satoshi Nakamoto creó los primeros bitcoins. Aunque el inventor de los bitcoins sigue siendo unánime, los bitcoins continuaron siendo creados y comercializados y una gran comunidad estaba allí para apoyar y abordar diversos problemas con el código.

Hay cientos de criptomonedas diferentes, como Litecoin, Dogecoin, etc., pero los bitcoins tienen la mayor parte del mercado se ha convertido en la criptodivisa más popular entre las demás. Fue capaz de llamar la atención de los usuarios debido a su capacidad de mantener la unanimidad de sus usuarios, pero se hizo realmente popular debido a su transparencia. Bitcoin comenzó a florecer desde entonces y en el año 2013, los inversores comenzaron a invertir en las empresas relacionadas con el Bitcoin. Los bitcoins pueden ser cambiarse por moneda corriente, por cualquier servicio o producto.

Con el uso de software de monedero, los usuarios pueden transferir electrónicamente transferir bitcoins electrónicamente mediante un ordenador, un móvil o una aplicación web. En 2015, se lanzó la plataforma Ethereum que permitió que la blockchain funcionara con préstamos y contactos. Se basaba en un algoritmo llamado contrato inteligente que aseguraba

la ejecución de una acción entre las dos partes. Debido a la capacidad de Ethereum de ofrecer un entorno más rápido, seguro y eficiente, la tecnología se hizo ampliamente popular.

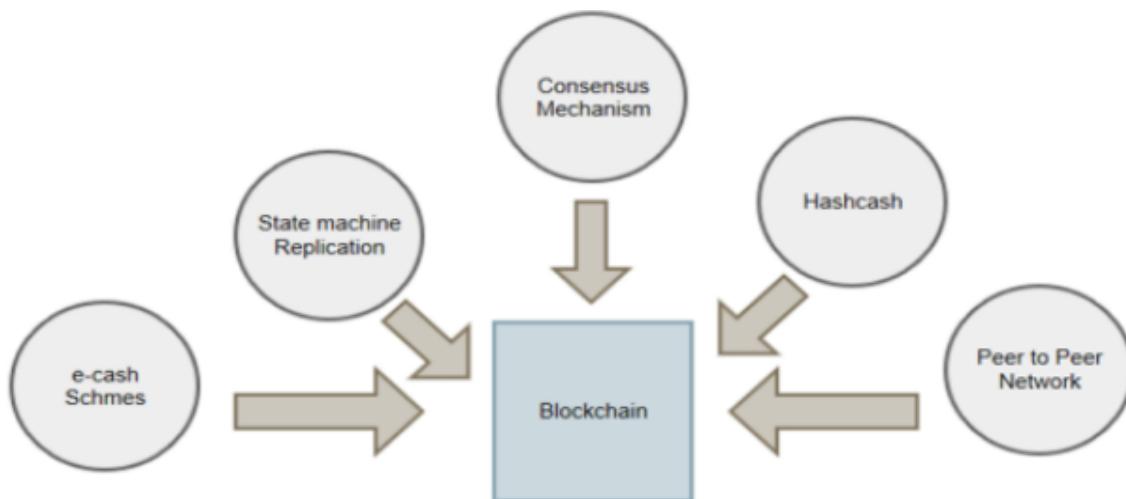


Figura 2. Blockchain

2.4. Arquitectura de la Blockchain

La tecnología Blockchain funciona con el concepto de base de datos descentralizada donde estas bases de datos existen en múltiples ordenadores y cada copia de estas bases de datos son idénticas.

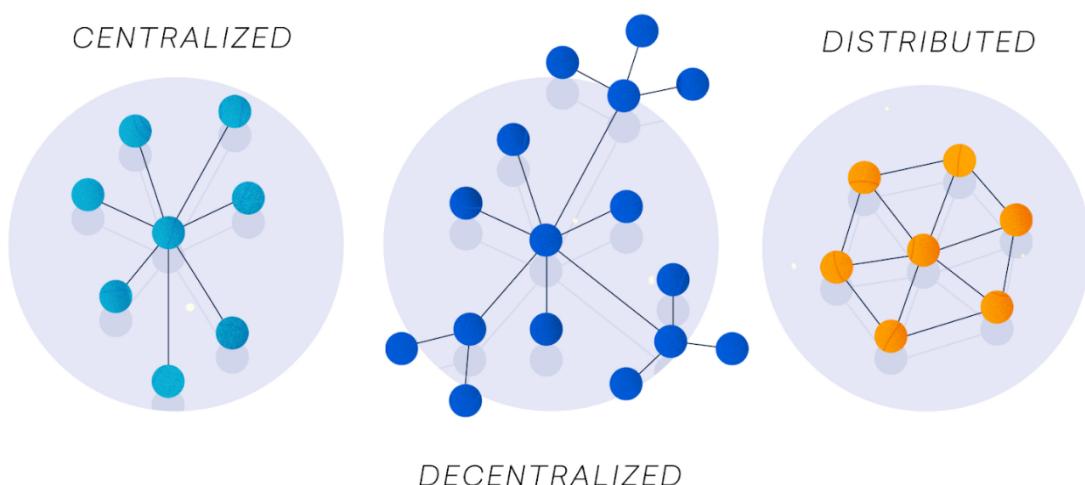


Figura 3. Concepto de base de datos

Las organizaciones mantienen sus datos en una base de datos centralizada lo que las convierte en un objetivo fácil para los hackers, mientras que debido a la estructura descentralizada de blockchain, ha hecho que el blockchain como una tecnología a prueba de mal genio. Blockchain puede ser considerada como una red peer to peer que se ejecuta en la parte superior de la Internet.

La arquitectura de Blockchain puede dividirse principalmente en tres capas que son Aplicaciones, ledger Descentralizado y Red de pares. Las aplicaciones son la capa superior de la red, seguida por el ledger Descentralizado y la capa inferior es la red de pares.

La capa de aplicación contiene el software de aplicación de la Blockchain. Por ejemplo, el software del monedero de Bitcoin crea y almacena claves privadas y públicas que permiten a los usuarios mantener control sobre los bitcoins no gastados. La capa de aplicación proporciona una interfaz legible para el ser humano en la que los usuarios pueden hacer un seguimiento de sus transacciones.

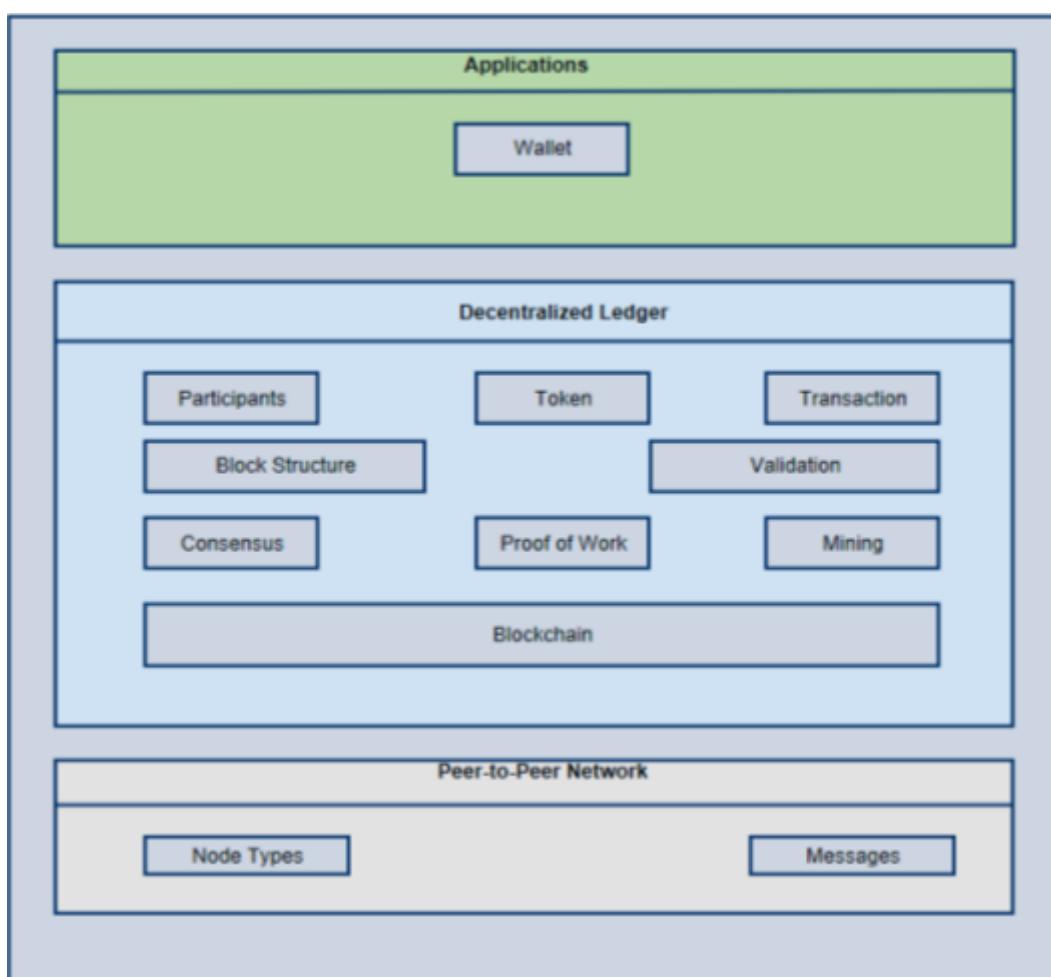


Figura 4. Arquitectura de la Blockchain [2]

El Ledger descentralizado es la capa intermedia de una arquitectura de blockchain que confirma un libro de contabilidad global consistente y ledger global. En esta capa, las transacciones pueden agruparse en bloques que están vinculados criptográficamente entre sí. Las transacciones pueden definirse como el intercambio de tokens (ver [apartado 2.5](#)) entre dos participantes y cada transacción pasa por un proceso de validación antes de ser considerada como una legítima. La minería es el proceso de agrupar las transacciones en un bloque que se añade al final de la cadena de bloques actual.

Blockchain utiliza un algoritmo de prueba de trabajo para decidir la cadena que ha requerido el mayor esfuerzo acumulado para su construcción y para asegurar el consenso entre todos los nodos para determinar la pierna de la cadena de bloques. La capa inferior de la arquitectura de la cadena de bloques es la red Peer-to-Peer, en la que los tipos de nodos juegan diferentes papeles y se intercambiaron varios mensajes para principal del ledger Descentralizado. Proporciona interfaces de aplicación en la parte superior de la cadena de bloques y se utiliza para mantener la seguridad de las criptomonedas. Este software puede instalarse en el ordenador o en los dispositivos móviles o también puede alojarse en una plataforma de terceros.

2.4.1. Aplicaciones

Proporciona interfaces de aplicación en la parte superior de la cadena de bloques y se utiliza para mantener la seguridad de las criptomonedas. Este software puede instalarse en el ordenador o en los dispositivos móviles o también puede alojarse en una plataforma de terceros.

2.4.2. Ledger descentralizado

Un ledger descentralizado es una base de datos compartida y replicada que se sincroniza entre los miembros de la red. En mantiene los registros de las transacciones entre los participantes en la red. El ledger es responsable de mantener los registros de las transacciones entre los participantes. Blockchain tiene la propiedad de una base de datos, excepto el hecho de que almacena la información en la cabecera y los datos se almacenan en forma de un token o una criptomoneda.

Es necesario agrupar las transacciones recién validadas en bloques como primer paso para registrar las transacciones en el ledger. Cualquier participante en la cadena de bloques puede reunir nuevas transacciones y crear bloques que pueden ser añadidos a la cadena de bloques. Un bloque se compone principalmente de transacciones y tiene punteros, marcas de tiempo y el nombre.

Los nodos realizan varias funciones dependiendo de su papel en la red blockchain. Un nodo puede llamarse minero cuando propone y valida transacciones y realiza la minería para proporcionar consenso para asegurar la cadena de bloques. Puede realizar funciones como la simple verificación de pagos, etc., y funciones que dependen de la cadena de bloques utilizada.

La prueba de trabajo se define como un algoritmo de consenso que verifica la exactitud de los datos. Por ejemplo, Bitcoin utiliza hashcash como prueba de trabajo para las transacciones de bitcoin. Los mineros deben completar una prueba de trabajo para verificar las transacciones en el bloque para que pueda ser aceptado por la red.

La prueba de trabajo garantiza la seguridad y el consenso en la red blockchain. Durante la verificación, un bloque recibe un hash (id). Para verificar el siguiente bloque, este hash se añade al bloque actual de transacciones. En el siguiente paso, se añade un nonce -que se define como un número aleatorio que puede utilizarse sólo una vez, al final del siguiente bloque. La función Hash se utiliza para cambiar este número aleatorio para generar una cadena que contiene número de ceros delante de él.

La prueba de trabajo es costosa de mantener, y puede tener problemas de escalabilidad y seguridad, ya que siempre depende de los incentivos de los mineros. Existe una solución avanzada llamada "proof-of-stake", que es lucrativa de aplicar, y que identifica quién puede actualizar el consenso y evita la bifurcación no deseada de la cadena de bloques subyacente.

En una red blockchain no se transfiere información confidencial y todas las transacciones son visibles para todos los nodos de la red. Esta red peer to peer no requiere ninguna protección adicional y puede construirse en cualquier infraestructura física.

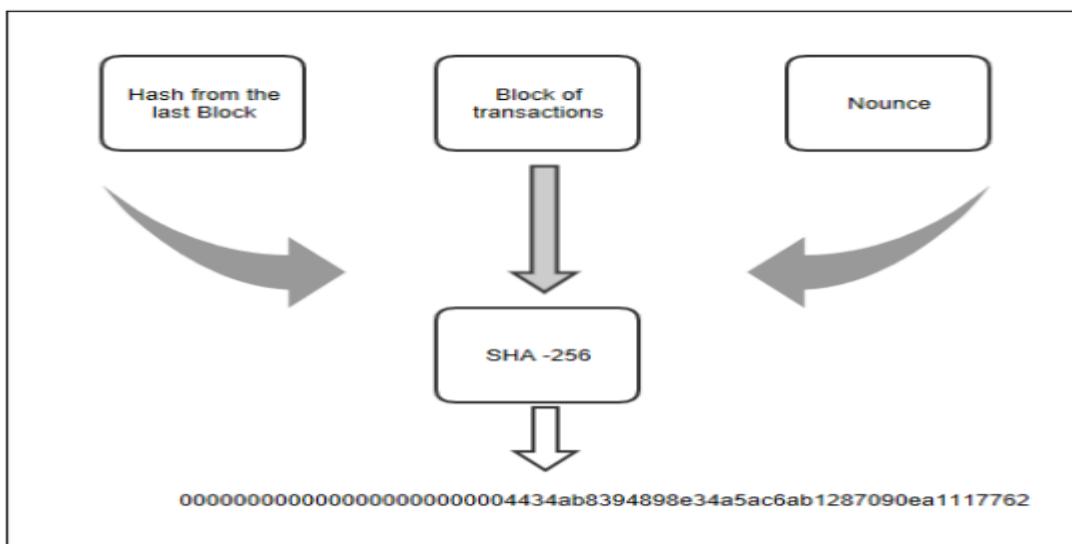


Figura 5. Ledger

2.5. Token

2.5.1. ¿Qué es un token?

Los tokens existen desde la creación de la primera criptomoneda, Bitcoin, en 2008. Por definición, un token es un activo digital que se puede emitir y comercializar en línea en una cadena de bloques, sin necesidad de duplicación o presencia de un intermediario. Así, a diferencia de una moneda tradicional, el emisor del token no necesita conservar un original de este ni obtener la validación de un tercero (banco, Estado, etc.) para realizar su transferencia. Un token también puede representar un derecho, un valor o incluso un poder dentro de una red. De hecho, cualquier usuario puede crear un token y darle una aplicación personalizada: medio de pago, derecho de uso, derechos de autor... Esta personalización se lleva a cabo cuando se crea el token, a través de un contrato inteligente. En consecuencia, es gracias a los tokens que los diferentes usuarios de una cadena de bloques interactúan entre sí.

Un token tiene todas las características de una criptomoneda, como la velocidad de intercambio (que se realiza en tiempo real y en unos minutos como máximo), la seguridad y la conservación de todas las transacciones dentro de una cadena de bloques, sirviendo como registro inalterable.

2.5.1. Los diferentes tipos de tokens

- **Token de utilidad**

Los utility tokens, o "tokens de utilidad" en español, permiten a su comprador acceder a un producto o servicio. No se consideran activos legalmente, pero aún es posible especular con este tipo de token. En efecto, su valor varía según el estimado de los productos y servicios a los que dan acceso.

- **Tokens de valor**

Los security tokens, o "tokens de valor" en español, permiten a los usuarios de Internet invertir en un proyecto de cadena de bloques, particularmente en el contexto de las ICO. Muy a menudo se comparan con activos financieros como las acciones. Sin embargo, esto es una tontería total, porque estos tokens de inversión en realidad no comparten ninguna de las características de estos valores.

De hecho, poseer estos tokens no ofrece a los inversores ninguna garantía en cuanto al logro de los objetivos de la empresa ni ningún peso en las decisiones estratégicas de esta última. Además, dichos tokens generalmente no tienen valor hasta que se logran los objetivos de la empresa.

- **Tokens de equidad**

Los equity tokens, o "Tokens de equidad" en español, representan el tipo de token que es verdaderamente comparable a los valores financieros de una empresa. Todavía en el estado de concepto actual, estos tokens deberían encarnar la forma lógica futura de los antiguos títulos utilizados en el origen de las bolsas de valores. Por lo tanto, los tokens de seguridad son acciones de una empresa que se han transpuesto dentro de una cadena de bloques. Por lo tanto, estas acciones son accesibles para todos los usuarios de la red, a cambio de su valor en criptomonedas.

Capítulo 3

3. Casos de uso

Blockchain es una tecnología de propósito general, lo que significa que es aplicable a todas las industrias. Por ejemplo, los servicios financieros pueden usarlo para redactar contratos inteligentes entre los consumidores y su institución bancaria. De manera similar, la atención médica puede usarlo para redactar contratos inteligentes entre aseguradoras y hospitales, así como entre pacientes y hospitales. Las posibilidades son infinitas.

Los casos de uso de blockchain continúan creciendo. Estas son algunas las aplicaciones en diferentes sectores

3.1. Casos de uso de blockchain en banca y finanzas

3.1.1. Pagos Internacionales

Blockchain ofrece una manera de crear de manera segura y eficiente un registro de actividad confidencial a prueba de manipulaciones. Esto lo hace excelente para pagos internacionales y transferencias de dinero.

Por ejemplo, en abril de 2018, Banco Santander lanzó el primer servicio de transferencia de dinero basado en blockchain del mundo. Conocido como "Santander One Pay FX", el servicio utiliza Current de Ripple para permitir a los clientes realizar transferencias de dinero internacionales el mismo día o al día siguiente.

Al automatizar todo el proceso en blockchain, Santander ha reducido la cantidad de intermediarios que normalmente se requieren en estas transacciones, lo que hace que el proceso sea más eficiente.

Como gran banco comercial, Santander tiene muchos clientes minoristas que se beneficiarían de pagos más eficientes y económicos, especialmente en el área de transferencias internacionales. La tecnología Blockchain se puede utilizar para reducir el

costo de estas transferencias al reducir la necesidad de que los bancos liquidan transacciones manualmente.

3.1.2. Los mercados de capitales

Los sistemas basados en blockchain también tienen el potencial de mejorar los mercados de capital.:

- **Compensación y liquidación más rápidas:** en teoría, la tecnología blockchain podría admitir la compresión de la compensación y la liquidación porque una vez que una transacción se confirma y se compromete con el libro mayor, el token asociado también se ha asentado en la billetera del beneficiario final. La liquidación más rápida conduciría a costos reducidos y menor riesgo de liquidación y fraude de la contraparte, aunque los pasos discretos del proceso asociados con la compensación y la liquidación aún facilitan la lógica comercial adicional, como la compensación y el margen. Las cadenas de bloques existentes están diseñadas con retrasos que van desde unos segundos hasta unos minutos para facilitar la verificación antes de que cada bloque se escriba en el libro mayor; Las cadenas de bloques alternativas que se están desarrollando actualmente pueden proporcionar tiempos de liquidación aún más rápidos. En cualquier caso, los retrasos son considerablemente más cortos que los actuales ciclos de liquidación de T+días en la negociación de valores.
- **Retención de documentación:** transferencias de activos más complejas en general comprenden términos personalizados y requieren procesos manuales a través del ciclo de vida de la transacción. En las cadenas de bloques, cualquier parte autorizada puede acceder y verificar los registros de propiedad, con información vital de transacciones y procesos integrados en "contratos inteligentes".
- **Consolidación de libros mayores:** la implementación de protocolos de cadena de bloques en una sola institución, con entidades legales o sucursales que actúan como nodos completos, podría abordar los requisitos reglamentarios para la consolidación de libros mayores patentados en un solo modelo de datos para fines de informes, si no más en un solo producto múltiple, registro multidivisa.
- **Pista de auditoría consolidada:** por diseño, las cadenas de bloques contienen historias detalladas y precisas de los movimientos de activos, lo que tiene el beneficio

adicional de ser atractivo para los reguladores. Con los libros de clientes y distribuidores en el mismo libro mayor, los requisitos relacionados con el lavado de dinero o el manejo de pedidos de clientes pueden abordarse de manera eficiente. Las cadenas de bloques incorporan únicamente marcas de tiempo relativas (o "altura del bloque").

- **Reducción del riesgo sistémico:** Está bien documentado que los mercados, incluidos los de financiación a corto plazo y repos, son vulnerables a una escasez de liquidez. Además, los largos ciclos de liquidación y los desajustes en la contabilización de operaciones hacen que las garantías sean un objetivo móvil. Los libros mayores distribuidos prácticamente eliminan el riesgo crediticio y de liquidez al requerir una financiación previa (presencia de efectivo y garantías en las cuentas) antes de la negociación. La capacidad de las cadenas de bloques para revertir/terminar transacciones y realizar otras nuevas en el día es prometedora para eliminar el llamado riesgo "a la luz del día" asociado con, por ejemplo, la sustitución de garantías en el repo.
- **Mejoras operativas en el middle y back-office:** Instrumento la estandarización y alineación de los términos antes del comercio de blockchain eliminaría varios procesos intermedios y administrativos, incluido el enriquecimiento comercial, la corrección de errores, las asignaciones y la comparación de contrapartes. La adopción generalizada de una sola cadena de bloques también eliminaría la necesidad de actividades como la conciliación de libros contables propietarios.
- **Eficiencia en el manejo de cuentas:** las cadenas de bloques están diseñadas para brindar escalabilidad y seguridad al nivel de la billetera o cuenta individual, y pueden usarse para crear cuentas ómnibus que están separadas legalmente y combinadas operativamente.
- **Redundancia de los sistemas de anotaciones en cuenta:** si bien muchos activos se negocian en forma digital, están efectivamente garantizados por intermediarios financieros que representan que están respaldados por activos físicos, al igual que los gobiernos en un momento garantizaron que las monedas estaban respaldadas por reservas de oro. Si los activos pueden representarse completamente en forma digital con la finalidad de la liquidación y pleno reconocimiento legal, el actual sistema de anotaciones en cuenta de valores entre los depositarios centrales de valores (CSD) y los custodios de "nombre de la calle" podría volverse innecesario en algunos casos. El

propio libro mayor se vuelve determinista, sin abstracción ni necesidad de que un tercero lo represente como verdadero.

3.1.3. Financiación del comercio

Los métodos históricos de financiación del comercio han sido un problema importante para las empresas, ya que los procesos lentos a menudo interrumpen el negocio y dificultan la gestión de la liquidez. El comercio transfronterizo implica una gran cantidad de variables al comunicar información, como el país de origen y los detalles del producto, y las transacciones generan grandes volúmenes de documentación.

Blockchain tiene la capacidad de optimizar los acuerdos de financiación comercial y simplificar el proceso a través de las fronteras. Facilita que las empresas realicen transacciones entre sí a través de fronteras regionales o geográficas.

3.1.4. Cumplimiento normativo y auditoría

La naturaleza extremadamente segura de la cadena de bloques la hace bastante útil para la contabilidad y la auditoría, ya que reduce en gran medida la posibilidad de errores humanos y garantiza la integridad de los registros. Además de eso, nadie puede modificar los registros de la cuenta una vez que están bloqueados mediante la tecnología blockchain, ni siquiera los propietarios de los registros. La compensación aquí es que la tecnología blockchain podría eliminar en última instancia la necesidad de auditores y eliminar puestos de trabajo.

3.1.5. Protección contra el lavado de dinero

Nuevamente, el cifrado que es tan integral en la cadena de bloques lo hace extremadamente útil en la lucha contra el lavado de dinero. La tecnología subyacente permite el mantenimiento de registros, que admite "Conozca a su cliente (KYC)", el proceso mediante el cual una empresa identifica y verifica la identidad de sus clientes.

3.1.6. Transacciones entre pares

Como vemos, los servicios de pago P2P son convenientes, pero tienen limitaciones. Algunos servicios restringen las transacciones en función de la geografía. Otros cobran una

tarifa por su uso. Y muchos son vulnerables a los piratas informáticos, lo que no es atractivo para los clientes que filtran su información financiera personal. La tecnología Blockchain, con todas sus ventajas antes mencionadas, podría resolver estos obstáculos.

3.2. Aplicaciones de blockchain en los negocios

3.2.1. Gestión de la cadena de suministro

El libro de contabilidad inmutable de Blockchain lo hace ideal para tareas como el seguimiento en tiempo real de bienes a medida que se mueven y cambian de manos a lo largo de la cadena de suministro. El uso de una cadena de bloques abre varias opciones para las empresas que transportan estos bienes. Las entradas en una cadena de bloques se pueden usar para poner en cola eventos con una cadena de suministro, por ejemplo, asignando productos recién llegados a un puerto a diferentes contenedores de envío. Blockchain proporciona una nueva forma dinámica de organizar y utilizar los datos de seguimiento.

3.2.2. Cuidado de la salud

Los datos de salud amigables con Blockchain incluyen información general como edad, género y datos de historial médico potencialmente básicos, como historial de vacunación o signos vitales. Por sí sola, ninguna de esta información podría identificar específicamente a un paciente en particular, lo que permitiría almacenarla en una cadena de bloques compartida accesible para muchas personas sin preocupaciones de privacidad indebidas.

A medida que los dispositivos médicos conectados especializados se vuelven más comunes y cada vez más vinculados al registro de salud de una persona, blockchain puede conectar estos dispositivos a ese registro. Los dispositivos podrán almacenar los datos generados en una cadena de bloques de atención médica y adjuntarlos a los registros médicos personales. Un problema clave al que se enfrentan actualmente los dispositivos médicos conectados es el almacenamiento en silos de los datos que generan, pero blockchain podría ser el vínculo que conecte estos silos.

3.2.3. Inamovible

El propietario promedio vende su casa cada cinco a siete años, y la persona promedio se mudará casi 12 veces en su vida. Con movimientos tan frecuentes, blockchain ciertamente podría ser útil en el mercado inmobiliario. Aceleraría las ventas de viviendas al verificar rápidamente las finanzas, reduciría el fraude con su encriptación y proporciona transparencia en todo el proceso de compra y venta.

3.2.4. Medios de comunicación

Las empresas de medios ya han comenzado a adoptar la tecnología blockchain para eliminar el fraude, reducir costos e incluso proteger los derechos de propiedad intelectual (IP) del contenido, como los discos de música.

3.3. Aplicaciones de blockchain en el gobierno

3.3.1. Gestión de registros

Los gobiernos nacionales, estatales y locales son responsables de mantener registros de las personas, como las fechas de nacimiento y defunción, el estado civil o las transferencias de propiedad. Sin embargo, la gestión de estos datos puede ser difícil y, hasta la fecha, algunos de estos registros sólo existen en papel. Y a veces los ciudadanos tienen que ir físicamente a las oficinas de su gobierno local para hacer cambios, lo que lleva mucho tiempo, es innecesario y frustrante. La tecnología Blockchain podría simplificar este mantenimiento de registros y hacer que los datos sean mucho más seguros.

3.3.2. Gestión de identidad

Los defensores de la tecnología de cadena de bloques para la gestión de identidad afirman que con suficiente información en la cadena de bloques, las personas solo tendrían que proporcionar la información mínima (fecha de nacimiento, por ejemplo) para probar su identidad.

3.3.3. Votar

La tecnología Blockchain tiene la capacidad de hacer que el proceso de votación sea más accesible al tiempo que mejora la seguridad. Los piratas informáticos no serían rival para la tecnología blockchain porque incluso si alguien obtuviera acceso a la terminal, no podría

afectar a otros nodos. Cada voto se atribuiría a una identificación, y dado que la posibilidad de crear una identificación falsa es imposible, los funcionarios del gobierno podrían contar los votos de manera más efectiva y eficiente.

3.3.4. Impuestos

La tecnología de cadena de bloques podría hacer que el tedioso proceso de presentación de impuestos, que es propenso a errores humanos, sea mucho más eficiente con suficiente información almacenada en la cadena de bloques.

3.3.5. Agencias sin fines de lucro

Blockchain podría resolver los problemas antimonopolio que enfrentan cada vez más las organizaciones benéficas a través de una mayor transparencia; la tecnología tiene la capacidad de mostrar a los donantes que las OSFL están de hecho usando su dinero según lo previsto. Además, la tecnología blockchain podría ayudar a estas NPO a hacer que estos fondos sean más eficientes, administrar mejor sus recursos y mejorar sus capacidades de seguimiento.

3.3.6. Cumplimiento normativo/supervisión

La mayor parte de la supervisión regulatoria proviene del mantenimiento de registros, pero las consecuencias de no mantener registros son posiblemente mucho peores. Por lo tanto, el cumplimiento no es negociable para las empresas. Blockchain puede hacer que las actualizaciones de registros estén disponibles para los reguladores y las empresas en tiempo real, lo que reduce los retrasos y detecta señales de alerta e inconsistencias antes.

3.4. Aplicaciones de blockchain en otras industrias

3.4.1. Gestión Financiera y Contabilidad

Si la cadena de bloques es realmente tan segura como se ha demostrado en los últimos años, esa seguridad impenetrable sería atractiva para los clientes preocupados por el fraude financiero.

3.4.2. Gestión de registros

Como se dijo anteriormente, el cifrado que está en el corazón de la cadena de bloques lo hace muy útil para administrar registros, ya que evita duplicados, entradas fraudulentas, etc.

3.4.3. La seguridad cibernética

El mayor beneficio de blockchain en ciberseguridad es que elimina el riesgo de un único punto de falla. La tecnología Blockchain también proporciona encriptación y privacidad de extremo a extremo.

3.4.4. Big Data

La naturaleza inmutable de la cadena de bloques y el hecho de que cada computadora en la red verifica constantemente la información almacenada en ella hace que la cadena de bloques sea una excelente herramienta para almacenar grandes cantidades de datos.

Capítulo 4

4. Estado del arte

El 31 de octubre de 2008, un desconocido que utilizaba el seudónimo "Satoshi Nakamoto" escribió en una lista de correo electrónico reservada a los cypherpunks (un movimiento de personas que utilizan la criptografía para proteger la privacidad). “ Estoy trabajando en un nuevo sistema de dinero electrónico que es totalmente peer-to-peer, sin terceros de confianza ”. Este texto es acompañado de un enlace a Bitcoin.org, donde está alojado el libro blanco de Bitcoin [32].

libro blanco, redactado en un inglés impecable, que resume el funcionamiento del nuevo protocolo. El primer concepto de Blockchain se aplicó el 03 de enero de 2009 en el contexto de Bitcoin.

La tecnología detrás de Bitcoin y otras criptomonedas es una base de datos de libro mayor distribuido para registrar las transacciones, permitiendo a los usuarios compartir su libro de transacciones.

En este capítulo se presenta y explica las características más importantes del Blockchain y sus conceptos asociados.

4.1. Tipos de blockchain

La continua innovación y evolución de la tecnología blockchain está dando lugar a varios tipos de blockchain. Por otro lado, el blockchain que fue inicialmente diseñado para aplicaciones de criptomonedas se está extendiendo rápidamente a los contratos inteligentes y muchas nuevas aplicaciones son ya se están estudiando muchas aplicaciones nuevas.

4.1.1. Clasificación basada en la red

La blockchain es un protocolo para la transmisión segura de información en una red descentralizada. Existen tres tipos de blockchain, según los derechos de acceso a la red. derechos de acceso a la red.

4.1.1.1. Blockchain pública

Una blockchain pública, como su nombre indica, es accesible a cualquiera: cualquiera puede unirse a la red y salir de ella cuando lo deseé. Así, los bloques de transacciones y la cadena de bloques pueden observarse públicamente, aunque los participantes sean anónimos. Además, cualquiera puede enviar transacciones y esperar que se incluyan en el registro; obviamente, si estas transacciones respetan las reglas de esta blockchain. En las cadenas de bloques públicas, cualquiera puede participar libremente en el proceso de aprobación, es decir, el proceso que decide qué bloque se añade a la cadena y define el estado del sistema. La cadena de bloques de Bitcoin y la cadena de bloques de Ethereum son dos ejemplos principales de cadena de bloques pública.

4.1.1.2. Blockchain privada

En una cadena de bloques privada, el acceso a la cadena de bloques está restringido a determinados participantes, por ejemplo, los de una organización. Esta restricción ayuda a simplificar las operaciones normales, como la creación de bloques y el modelo de contingencia. Así, el proceso de aprobación es controlado por un número reducido y seleccionado de nodos.

Esto conlleva una doble modificación del sistema original, ya que no sólo se limitan y seleccionan los participantes en el proceso de aprobación, sino que además ya no se aplica la regla de la mayoría. En cuanto a la autorización para leer la blockchain, es decir, el acceso al registro, puede ser pública o estar reservada a los participantes de la red. Puede haber casos de Blockchains privadas en las que el proceso de aprobación se limita a un solo actor, aunque público, aunque los permisos de lectura, por ejemplo, pueden ser públicos.

4.1.1.3. Blockchain (« autorizada »)

La tercera categoría de blockchain es la blockchain autorizada, también llamada blockchain de consorcio. Está destinado a un consorcio de partes que colaboran para realizar transacciones en una cadena de bloques para facilitar blockchain para facilitar la gobernanza, la procedencia y la responsabilidad, por ejemplo, un consorcio de todas las empresas de automoción u organizaciones sanitarias. La blockchain autorizada tiene las ventajas de una

blockchain pública al permitir que sólo los usuarios autorizados colaboren y para colaborar y realizar transacciones. La existencia de una criptomoneda no es necesario para este tipo de blockchain: no necesitan remunerar a sus miembros por validar sus miembros para validar las transacciones.

Las cadenas de bloques privadas tienen ciertas ventajas, como la simplificación de la gobernanza, actores conocidos, reducción de costes, rapidez, confidencialidad, cumplimiento facilitado por la posibilidades de auditoría, incluso por parte del regulador... Sin embargo, reintroducen a los actores humanos actores humanos en la gestión de la red (gestionar el acceso y el funcionamiento), mientras que el concepto central de una blockchain de una cadena de bloques (pública) es eliminar al tercero de confianza. Además de la clasificación de las blockchains según la red (pública, privada o autorizada) ; Pueden clasificarse en tres generaciones según el tipo de aplicación.

4.1.2. Clasificación basada en las aplicaciones derivadas

La infraestructura de la cadena de bloques ha evolucionado rápidamente. Inicialmente diseñada para aplicaciones de criptomonedas, la cadena de bloques se está extendiendo rápidamente a los contratos inteligentes y ya se están considerando muchas aplicaciones nuevas. Para ello, no hablamos de una blockchain, sino de varias Blockchains que existen, coexisten e incluso interactúan. Así, una cadena de bloques puede tener especificidades técnicas para usos o aplicaciones particulares que dan lugar a tres generaciones de Blockchain [37], a saber, Blockchain 1.0, Blockchain 2.0 y Blockchain 3.0.

4.1.2.1. Blockchain 1.0: Criptomonedas

Blockchain 1.0 se refiere a la primera cadena de bloques y su enfoque inicial en las criptomonedas. De hecho, la cadena de bloques fue, ante todo, inventada y desarrollada en el contexto de la implantación de Bitcoin, que permite realizar transacciones en línea sin intermediarios [1]. La gestión de las transacciones y la creación del dinero electrónico Bitcoin (BTC) son gestionadas colectivamente por la red. La cadena de bloques de Bitcoin es de código abierto y todo el código está disponible en GitHub [23]. En los primeros años, este código abierto se extendió al lanzamiento de varias criptomonedas. Se han introducido unas 300 altcoins cada una tiene sus propias especificidades pero todas funcionan pero todos

funcionan según el mismo principio. Por ejemplo, se pueden enumerar las siguientes monedas:

- **Litecoin [6] [44]** : creada en 2011, Litecoin es una aplicación que permite realizar pagos instantáneos a coste casi nulo a cualquier persona del mundo. Se trata de una red global de pagos de código abierto totalmente descentralizada y sin autoridad central. La blockchain de Litecoin es capaz de manejar un mayor volumen de transacciones que su contraparte - Bitcoin. Gracias a la mayor frecuencia de generación de bloques, la red admite más transacciones sin necesidad de realizar cambios de software en el futuro.
- **Dash [3]** : creado en 2014, se basa en el código fuente de Bitcoin para hacerlo más seguro y completamente segura y completamente anónima. Dash permite realizar pagos rápidos haciendo que la información financiera sea privada.
- **Conscioin [35]** : se trata de una criptomoneda ética, que incluye una forma de inteligencia artificial para vigilar el entorno y garantizar que las transacciones no sean inteligencia artificial para vigilar el entorno y garantizar la ética de las transacciones. transacciones.

Además del lanzamiento de otras criptomonedas, la funcionalidad de scripting de bitcoin se está ampliando a un marco completo de ejecución de código llamado contratos inteligentes. Esto da a la cadena de bloques su segunda generación.

4.1.2.2. Blockchain 2.0: contratos inteligentes

La segunda generación, Blockchain 2.0, está marcada por contratos inteligentes y representada por el creciente éxito de la cadena de bloques Ethereum [4]. A diferencia de Bitcoin, Ethereum no se limita a las transacciones financieras sino que permite la creación de cualquier aplicación descentralizada gracias a los contratos inteligentes, que son pequeñas piezas de software capaces, de forma autónoma, de ejecutar automáticamente determinadas acciones según condiciones predefinidas. Un contrato inteligente proporciona la capacidad de ejecución de código muy poderosa para integrar la lógica comercial de la cadena de bloques. Esta arquitectura permite eliminar cualquier intermediario entre dos actores de esta blockchain, siendo uno el usuario y el otro el proveedor de un servicio. Registrados en la cadena de bloques, estos contratos no pueden verse afectados por ningún cambio. Reservamos un capítulo para ethereum porque nuestra solución se basa en esta cadena de bloques.

Entre las aplicaciones ya existentes sobre Ethereum, podemos mencionar las siguientes contribuciones:

- **Slock.it** [11] pretende convertirse en la futura infraestructura de la economía colaborativa. La startup Slock.it, bajo el lema “ Rent, sell or share anything – without middleman ”, desarrolla su oferta combinando las ventajas del Blockchain (ausencia de intermediario) y los del Internet de las Cosas. Así, ofrece, por ejemplo, cerraduras especiales para las puertas que puede abrir una persona que ha alquilado un apartamento, a través de su teléfono móvil.
- **Arcade City** [24] planea derrocar a Uber y convertirse así en la primera "killer app" en la cadena de bloques. Su objetivo es ofrecer un servicio de coche (« car sharing »), sin la restricción del intermediario. Todas las carreras realizadas están contenidas en la blockchain de Ethereum. La ventaja para el consumidor está a nivel del costo ya que al reducirse el número de intermediarios, los precios también bajan.
- **La DAO** quiere convertirse en la primera organización descentralizada en Blockchain. Los miembros de “The DAO” tendrán por ejemplo la posibilidad de votar para determinar a qué proyectos se les repartirán éteres para incentivar el desarrollo de determinadas startups.

Los conceptos y características de la Blockchain pueden extenderse a muchos ámbitos de actividad. áreas de actividad. Así, estamos convergiendo en la tercera generación de blockchain que no implica ninguna moneda (dinero), sino que admite la ejecución de software para la lógica empresarial.

4.1.2.3. Blockchain 3.0: futuras aplicaciones

Blockchain 3.0 se asimila a futuras aplicaciones de blockchain con un punto de inflexión totalmente alejado del aspecto financiero. Por lo tanto, ya existen muchas experiencias que prueban el enorme potencial de disruptión de Blockchain. A continuación enumeramos brevemente algunos ejemplos:

- **Namecoin** [9] es una de las primeras aplicaciones Blockchain no financieras. Es una tecnología experimental de código abierto que mejora la descentralización, la seguridad, la resistencia a la censura, la privacidad y la velocidad de ciertos componentes de la infraestructura de Internet, como el DNS y las identidades. Namecoin [18] se inspiró después de las discusiones sobre un protocolo BitDNS que

usa blockchain para ejecutar un servicio de búsqueda de nombres de dominio. La motivación fue que una autoridad central que administra los nombres de dominio, como la ICANN, depositó demasiada confianza en una sola entidad y representó un único punto de falla.

- **Hyperledger Fabric [26]**: Es un sistema de código abierto modular y extensible para implementar y usar cadenas de bloques autorizadas y uno de los proyectos Hyperledger alojados por Linux Foundation [5]. Admite protocolos de consenso modulares, lo que permite que el sistema se adapte a casos de uso particulares y modelos de confianza. Fabric es también el primer sistema de cadena de bloques que ejecuta aplicaciones distribuidas escritas en lenguajes de programación estándar de propósito general sin una dependencia sistemática de una criptomoneda nativa. Esto contrasta marcadamente con las plataformas de cadena de bloques existentes que requieren que los "contratos inteligentes" se escriben en lenguajes específicos de dominio o que se basen en criptomonedas.
- También están en marcha iniciativas destinadas a introducir Blockchain en el contexto de los MOOC, pero también en la publicación académica. Por extensión, Blockchain también podría desarrollarse dentro de los gobiernos. De hecho, permitiría ofrecer ciertos servicios estatales de forma descentralizada, como el registro de matrimonios, la emisión de pasaportes, etc. Al final, la organización de elecciones utilizando Blockchain también podría garantizar una mayor transparencia a los Estados.

Las cadenas de bloques privadas y autorizadas permiten el acceso controlado a la cadena de bloques. Además, importantes innovaciones, como los contratos inteligentes, han abierto nuevas aplicaciones para la tecnología blockchain. Para todas estas aplicaciones, la técnica utilizada para garantizar la seguridad de las transacciones sigue siendo casi la misma.

4.2. Estructura de la Blockchain

Bitcoin es la madre de todas las cadenas de bloques. En este sentido, la estructura general de otras cadenas de bloques es muy similar a la de bitcoin. Entonces, para comprender la estructura de Blockchains, presentamos la estructura de Bitcoin. La transacción es el elemento básico de la cadena de bloques de Bitcoin. Las transacciones son validadas y transmitidas.

Muchas transacciones forman un bloque. Muchos bloques forman una cadena a través de un enlace de datos digitales [2].

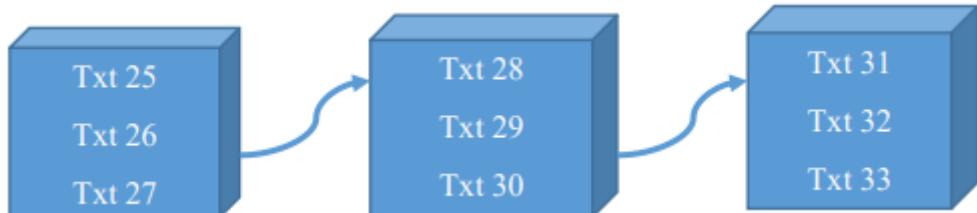


Figura 6. cadena de bloques

Los bloques pasan por un proceso de consenso para seleccionar el siguiente bloque para agregar a la cadena. El bloque elegido se verifica y se agrega a la cadena actual. El proceso de validación y consenso lo realizan nodos especiales llamados mineros. Se trata de ordenadores con grandes capacidades de procesamiento, directamente útiles para el funcionamiento de la Blockchain y capaces de emitir transacciones de la misma forma que un nodo normal.

4.2.1. La transacción

Una transacción [46] representa un pago en bitcoins, en otras palabras, una transferencia de activos (de bitcoins). Contiene datos técnicos y se caracteriza por entradas y salidas.



Figura 7. Forma simplificada de una transacción de Bitcoin

Los **datos técnicos** son:

- El **número de versión**: se utiliza para especificar a qué conjunto de reglas de protocolo se refiere esta transacción,
- **Término de transacción** (nLockTime): la marca de tiempo que hace que una transacción sea válida sólo a partir de una fecha futura, similar a un cheque posfechado,
- El **recuento de entradas**: el número de entradas contenidas en esta transacción,
- Y el **recuento de salida**: la cantidad de salida que crea esta transacción.

Las entradas son las "monedas" gastadas. Técnicamente, se caracterizan por la longitud del script de entrada (la cantidad de datos de entrada) y el número de secuencia. Además las entradas de transacción contienen, por un lado, el hash de la transacción anterior y el índice que permite identificar de dónde provienen las monedas especificando un valor de salida de una transacción anterior y, por otro lado, los datos del script donde demostramos que somos dueños de las monedas y que estamos autorizados a gastarlas, firmando con la clave privada de la dirección que contiene los bitcoins.

Las salidas son las monedas recibidas por el destinatario de la transacción. Contienen tres propiedades: la **longitud del script** de salida (número de datos de salida), la cantidad, es decir, el número de bitcoins enviados y el **script de salida** que determina a quién (a qué dirección/es) se envían y qué firmas se requieren para volver gastar esos bitcoins.

Las entradas y salidas revelan un concepto fundamental de la red bitcoin: las salidas de transacciones no gastadas ("Salida de transacciones no gastadas") denominadas **UTXO** [22]. El conjunto de todos los UTXO en una red de Bitcoin define colectivamente el estado de la cadena de bloques de Bitcoin.

Se hace referencia a los UTXO como entradas de una transacción y también son salidas generadas por una transacción. Todos estos UTXO residen en un sistema y los nodos participantes los almacenan en una base de datos. La transacción toma la cantidad especificada por uno o más UTXO y la pasa a uno o más UTXO de salida recién creados,

según la solicitud iniciada por el remitente. Expresamos en paradigma el mecanismo de validación de una transacción por un nodo:

- Para cada entrada en la transacción:
 - Si el UTXO al que se hace referencia no está en la base de datos de UTXO, devuelve un error.
 - Si la firma proporcionada no coincide con la del titular de la UTXO (es decir, no es la misma dirección), devolver un error.
- Si la suma de las cantidades de las UTXO a las que se hace referencia en la entrada no es igual a la suma de las cantidades de las UTXO en la salida, devolverá un error.
- Actualizar la base de datos de UTXO.

En la Figura 8 siguiente se muestra un ejemplo de transacción de bitcoin (disponible en blockchain.com).

Inputs ⓘ

Index	0	Details	Output
Address	12cgPFdJVixbwHbhrA3TuW1EGnL25Zqc3P ⓘ		Value 20.09195266 BTC
Pkscript	OP_DUP OP_HASH160 11b7eb8a3c1cc8a2a076c8ce916a4f0da3a18ab6 OP_EQUALVERIFY OP_CHECKSIG		
Sigscript	3044022053ad3c35d28f6df25eee232b7e28cf73eac39d130cff322f18ce3b5684599a102202a485d3bbe3186cbad0f5c2ce5e1cde93e43af97459bca9116907b149a243ea101 03a0c53fcc4704ba78331a896c3bd684328b44890b25f91cfb853ab0bb301c7875		
Witness			

Outputs ⓘ

Index	0	Details	Spent
Address	3BMExrAmyBb8683C9BmP6CoQ3rLADDLJAP ⓘ		Value 0.00950000 BTC
Pkscript	OP_HASH160 69f376bcbe63bd2ce5980ca8c455c848c52dcf15 OP_EQUAL		
Index	1	Details	Spent
Address	1FFxAGNwSmwwgtGLQf45Hes5Emrx9zXJRK ⓘ		Value 0.16750880 BTC

Figura 8. Ejemplo de transacción de Bitcoin

4.2.2. El bloque

Los bloques son unidades de Blockchain comparables a páginas de transacciones en un libro de cuentas. Un bloque se compone de una cabecera y un contenido en el que se agrupan las transacciones.

El encabezado del bloque se procesa dos veces para crear el hash al que se hace referencia en el siguiente bloque. Contiene:

- Datos técnicos que incluyen un ID mágico, número de versión (especifique a qué conjunto de reglas de protocolo se ajusta este bloque) y tamaño de bloque.
- Hash del **bloque anterior**: el hash del encabezado del bloque anterior (excluyendo el ID mágico y el tamaño del bloque). Este es el eslabón que crea la cadena de bloques, y esto a partir del bloque génesis (“bloque Génesis”). Por lo tanto, en la Figura 9, el bloque 2 está efectivamente ubicado entre los bloques 1 y 3, lo que se puede verificar asegurándose de que el hash del bloque 1 se ingrese correctamente en el encabezado del bloque 2 y lo mismo para el hash del bloque 2 en el encabezado del bloque 3.
- La **raíz de Merkle**: todas las transacciones en el bloque se destilan en un solo hash. De hecho, se calcula para cada transacción un identificador (TxID) que es el hash del contenido de la transacción. El árbol Merkle entonces permite solidificar todas las transacciones calculando hashes sucesivos, hasta encontrar la raíz del árbol.
- Marca de tiempo de creación del bloque.
- La **dificultad específica**: relacionada con la minería y la dificultad de extraer con éxito el bloque.
- Y el **nonce**: que es un número aleatorio. Esta es una de las cosas que puede modificar mientras extrae, para crear diferentes hashes y encontrar el hash correcto.

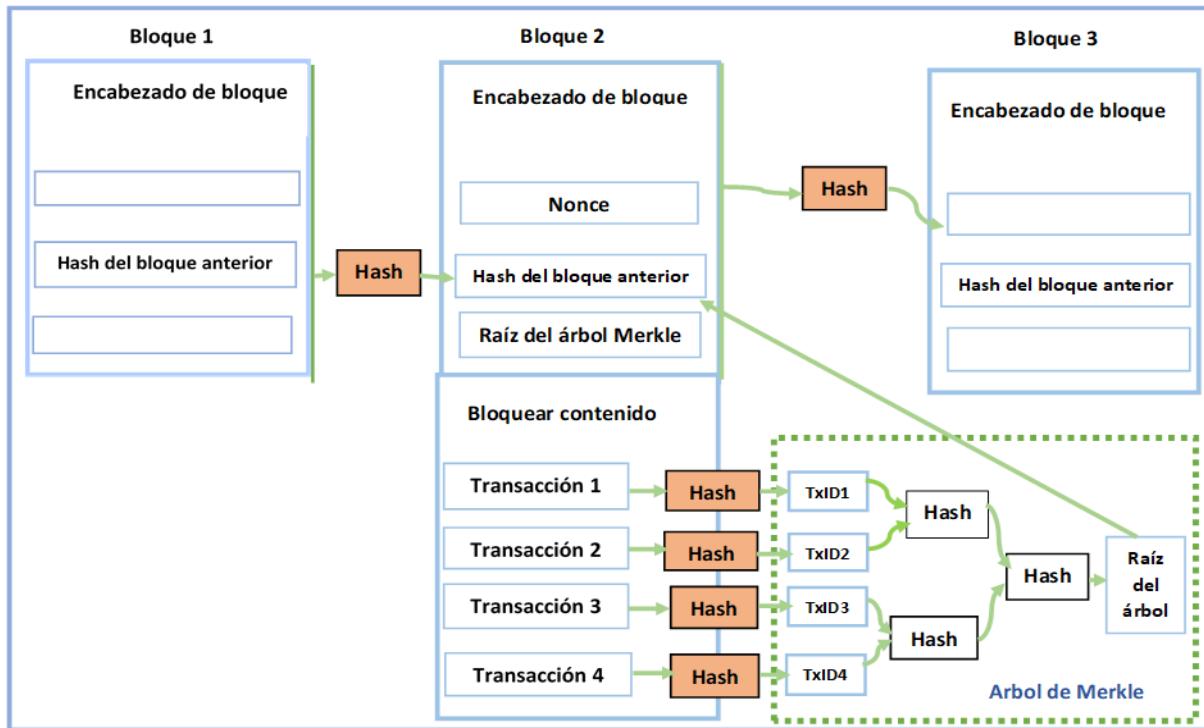


Figura 9. Forma simplificada de un bloque de Bitcoin [36]

El contenido del bloque consiste en la transacción coinbase que es una transacción especial donde no hay entradas ni dirección del remitente y el conjunto de transacciones en el bloque que representa la carga principal.

4.3. La seguridad

La seguridad en la informática se trata de garantizar que los recursos de hardware o software de una organización se utilicen sólo según lo previsto. Corresponde a cualquier organización proteger su patrimonio que está representado, esencialmente, por su sistema de información.

Un sistema informático es seguro si incorpora mecanismos para hacer frente a vulnerabilidades, amenazas, riesgos y ataques. De hecho, una vulnerabilidad o falla en un sistema es una debilidad o software que permite a un atacante comprometer la seguridad de la información o un sistema de información. Las amenazas, por otro lado, son acciones que son potencialmente dañinas para el sistema. Un riesgo se refiere a la probabilidad de un evento dañino y los costos resultantes. El riesgo también depende de la cantidad de valores a proteger. Un ataque es la explotación de una falla en un sistema informático para propósitos

desconocidos para el operador del sistema y generalmente dañinos. Para asegurar un sistema, debemos asegurarnos de:

- Autenticación: la identificación del usuario es esencial para administrar el acceso a los espacios de trabajo relevantes y mantener la confianza en las relaciones de intercambio.
- Integridad: Asegurar que los datos no sean alterados accidentalmente, los elementos considerados deben ser exactos y completos.
- Confidencialidad: Limita el acceso a la información solo a las personas autorizadas.
- Disponibilidad: Garantizar el perfecto funcionamiento del sistema, y el acceso a los servicios y recursos instalados con el tiempo de respuesta deseado.
- No repudio: Es la propiedad que da prueba de la autenticidad de un acto, es decir que ningún usuario puede entonces impugnar las operaciones que ha realizado, y que ningún tercero puede reclamar las acciones de otro usuario.

El almacenamiento de información en cualquier sistema distribuido, y por tanto también en una cadena de bloques, está sujeto a limitaciones fundamentales, conocidas como el "teorema CAP": es imposible garantizar tanto la consistencia (C) como la disponibilidad (A) de los datos a lo largo del tiempo en una red de particiones (P). Las propiedades de seguridad que las cadenas de bloques quieren garantizar en una red de la que desconfiamos porque no es segura (por lo tanto, implica coherencia y resistencia a las particiones) se obtienen a expensas de la disponibilidad de información reciente que ha sido consensuada o "consensuada" por todo el sistema.

En una cadena de bloques, en ausencia de una autoridad central que determine el estado correcto, el consenso se determina de forma distribuida aplicando técnicas probabilísticas (como prueba trabajo) para la elección de un nodo, que será responsable por un tiempo limitado de la evolución del estado del sistema (típicamente, por la duración de un solo bloque).

4.3.1. Vulnerabilidades, amenazas, riesgos y ataques en la cadena de bloques

La tecnología Blockchain es un componente "simple" de un sistema, aplicación o servicio, al igual que una base de datos en una aplicación comercial. Proporciona propiedades

de seguridad, pero también se deben evaluar las vulnerabilidades y los riesgos. Sin embargo, en la práctica, existen varios ataques que manipulan directa o indirectamente el mecanismo de recompensa, otorgando ventajas injustas a los grandes mineros a expensas de los pequeños mineros. Si inicialmente se suponía que la seguridad de Bitcoin se vería comprometida por un ataque de tipo "50% + 1", ahora reconocemos que el tamaño crítico (en términos de potencia de cómputo) de un actor malicioso que quisiera manipular el sistema es menor: el umbral crítico para lanzar un ataque conocido como "selfish mining" se estima en un 30% o menos [19]. Otros trabajos concluyen que cualquier arbitraje del que disfrutan los mineros, como elegir qué transacciones encajan en un bloque de la cadena de bloques, permite manipular el comportamiento de otros mineros [34]. Y dado que el valor intrínseco de una unidad de criptomoneda es difícil de estimar y está sujeto a una fuerte especulación, cualquier fenómeno de una 'burbuja' de precios de mercado corre el riesgo de intensificar el compromiso de los mineros, al aumentar el consumo total de energía, y de desalentar el uso de la criptomoneda. Para ejecutar transacciones legítimas, ya sea a través de tarifas altas [28] [40] o una mayor volatilidad de los precios. Esto al tiempo que alimenta el vigor de los ataques contra el sistema, que en cualquier momento podrían pulverizar la confianza de los usuarios y mineros si de repente se explotara una falla grave en los algoritmos o protocolos.

Tal como lo describe el grupo de seguridad del comité ISO TC307, existen varias propiedades de seguridad proporcionadas por los sistemas blockchain, denominadas en el contexto de ISO TC307 como DLT (Distributed Ledger Technologies). La siguiente es una lista de estas propiedades de seguridad. Algunas de ellas son propiedades de seguridad comunes a todas las aplicaciones basadas en DLT, mientras que otras son opcionales y dependen de la naturaleza de la aplicación.

Esta lista es provisional y está sujeta a cambios:

- Integridad: los registros en el libro mayor están protegidos contra cualquier modificación posterior a su creación. También conocida como inmutabilidad o resistencia a la alteración.
- Autenticidad: cualquier persona (o un conjunto certificado de entidades) puede verificar la entidad creando una transacción registrada en el libro mayor.
- Confidencialidad: los registros en el registro sólo pueden ser vistos por una entidad autorizada.

- El orden de los eventos: el orden de los registros en el libro mayor no se puede cambiar.
- Disponibilidad: las transacciones registradas en el libro mayor y la funcionalidad para guardar y recuperar los datos están siempre disponibles para los usuarios.
- Resiliencia: el sistema DLT continúa funcionando según lo previsto frente a fallas y otros incidentes. Es una especie de requisito de disponibilidad.
- "Servidor de confianza menos": incluso si no hay una entidad (servidor de confianza), Blockchain (DLT) continúa funcionando como se esperaba.
- Garantía a largo plazo: algunos casos de uso (por ejemplo, registros de tierras) implican el mantenimiento a largo plazo del libro mayor y la transferencia segura de sus registros a otro sistema en caso de desmantelamiento.

Los riesgos asociados a componentes que interactúan con una Blockchain como personas físicas o jurídicas que utilizan interfaces, aplicaciones, objetos inteligentes como medidores de electricidad o sensores, son múltiples. Por ejemplo, todos estos componentes deben tener un identificador único y seguro en la cadena de bloques para evitar cualquier robo de identidad y poder asociar una responsabilidad con una acción en la cadena de bloques.

Los riesgos de datos también deben ser analizados. ¿Cómo puede estar seguro, por ejemplo, de que una transacción enviada al sistema será validada y que no ha habido alteración de los datos por parte de un contacto inteligente o del sistema mismo? La integridad de los datos injectados y manipulados es un punto clave de la cadena de bloques. ¿Cómo cumple con los casos de uso o las reglamentaciones vigentes que exigen la confidencialidad de los datos, como los datos personales?

Desde la perspectiva de la norma ISO, los riesgos y vulnerabilidades del sistema DLT incluyen riesgos y vulnerabilidades comunes del sistema de información, tales como:

- ➔ Mal manejo de la información (alteración, borrado, destrucción no autorizada, divulgación, etc.).
- ➔ Vulnerabilidades de implementación (incluidos mecanismos criptográficos, vulnerabilidades de implementación, fugas de información en tiempo de ejecución, etc.).
- ➔ Mala gestión de los mecanismos criptográficos (incluido el uso de algoritmos débiles, divulgación de claves).

- ➔ Mala gestión de los privilegios de los usuarios.

Identificar vulnerabilidades y/o amenazas de blockchain ayuda a comprender mejor los mecanismos de seguridad incorporados en sus protocolos.

4.3.2. Mecanismos de seguridad

Para hacer frente a las amenazas y vulnerabilidades de un sistema, se deben implementar varios mecanismos de seguridad. Cada mecanismo es específico e incluso puede tener debilidades o ventajas sobre otro.

4.3.2.1. Criptografía

La criptografía es la ciencia que utiliza las matemáticas para cifrar y descifrar datos. Por lo tanto, nos permite almacenar información confidencial o transmitirla a través de redes no seguras (como Internet), para que nadie más que el destinatario pueda leerla. Hay dos tipos de criptografía: simétrica (o clave secreta) y asimétrica (o clave pública).

4.3.2.2. Criptografía simétrica

En la criptografía convencional, también denominada clave secreta o cifrado de clave simétrica, una sola clave es suficiente para el cifrado y descifrado. Toda la seguridad de esta criptografía está directamente relacionada con el hecho de que la clave de cifrado sólo la conocen el remitente y el destinatario. La criptografía simétrica es muy utilizada y se caracteriza por una gran velocidad (cifrado sobre la marcha), implementaciones de software (firewalls de software tipo Krypto Zone, Checkpoint Firewall-1 y VPN-1) que de hardware (tarjetas dedicadas, criptografía de 8 a 64 bits). procesadores, algoritmos cableados, etc.), lo que acelera significativamente las velocidades y autoriza su uso masivo. Sin embargo, tiene inconvenientes. De hecho, la distribución de claves es el principal problema de este método de cifrado. Si la misma clave es utilizada por más de dos personas, debe desecharse cuando se intercepta una copia. No se puede autenticar porque es conocido por más de una persona. Por otro lado, se destaca la incertidumbre en la seguridad durante la transferencia de la clave y la necesidad de generar tantas claves como pares de correspondientes.

4.3.2.3. Criptografía asimétrica

La criptografía de clave pública [28], también denominada criptografía asimétrica, es un método de cifrado que utiliza dos claves que son matemáticamente similares pero no idénticas: una clave pública y una clave privada. A diferencia de los algoritmos de criptografía simétrica que dependen de una sola clave para el cifrado y el descifrado, cada una de las claves de la criptografía asimétrica tiene una función muy específica: la clave pública se usa para cifrar y la clave privada se usa para descifrar. El hecho de que sea imposible adivinar la clave privada de la clave pública le otorga un alto nivel de seguridad.

En este sentido, las claves públicas se pueden compartir de manera segura, lo que permite a los usuarios beneficiarse de un método fácil y conveniente de encriptación de contenido y verificación de firma digital. En cuanto a las claves privadas, se mantienen en secreto. Por lo tanto, solo su propietario puede descifrar el contenido y crear firmas digitales. Por ejemplo, considere una comunicación entre dos entidades A y B en la Figura 10.

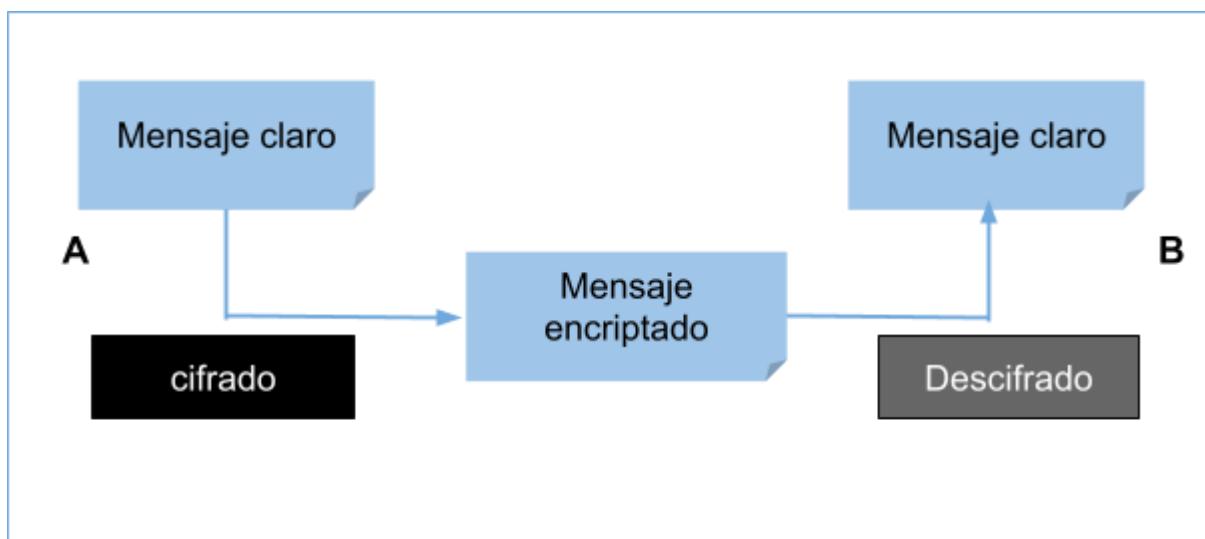


Figura 10. Principio de criptografía asimétrica

A y B generan sus pares de claves. A cifra su mensaje con la clave pública de B para garantizar que solo B pueda descifrar el mensaje. Entonces, para leer el mensaje de A, B descifra el mensaje encriptado usando su clave privada.

Hay varios criptosistemas que implementan la criptografía asimétrica. Dado que el par de claves pública-privada se usa con frecuencia en muchas operaciones. A diferencia del protocolo blockchain, es importante elegir un algoritmo eficiente y potente. Es en este sentido que se eligió el criptosistema ECC (Elliptic Curve Cryptography) en la blockchain de bitcoin

y en la blockchain de Ethereum para generar el par de claves, con el algoritmo de firma **ECDSA** [14] (Elliptic Curve Digital Signature Algorithm). El algoritmo ECDSA se basa en curvas elípticas que tienen la ventaja de garantizar, para el mismo nivel de seguridad, tamaños de clave razonables en comparación con otros criptosistemas de clave pública más tradicionales como RSA (Rivest Shamir Adelman). De hecho, para un nivel de seguridad de 1124, donde RSA requiere claves de 3072 bits, ECC solo requiere 256 bits.

4.3.2.4 Firma digital

Las firmas digitales permiten al destinatario verificar la autenticidad de los datos, su origen, pero también asegurarse de que estén intactos. Así, las firmas digitales garantizan la autenticación y la integridad de los datos. También proporcionan funcionalidad de no repudio. Estas funciones juegan un papel tan importante para la criptografía como la privacidad, si no más. Una firma digital tiene el mismo propósito que una firma manuscrita. Sin embargo, una firma manuscrita se puede imitar fácilmente, mientras que una firma digital es prácticamente infalsificable. Además, certifica el contenido de la información, así como la identificación del firmante.

4.3.2.5. Hash

Las funciones hash son funciones matemáticas que transforman una cadena de caracteres de cualquier longitud en otra cadena de longitud fija. Se caracterizan por:

- Irreversibilidad: Conociendo el resultado de la función, es muy difícil encontrar el mensaje proporcionado como entrada.
- Resistencia a colisiones: Es muy difícil encontrar dos mensajes que den como resultado el mismo hash.
- Efecto avalancha: La modificación de un bit del mensaje de entrada provoca que al menos la mitad de los bits se modifiquen en la salida. Esta propiedad es interesante para garantizar la propiedad de integridad.

Se hace una distinción entre el hash simple y el hash en forma de árbol llamado árbol Merkle. En el enfoque de hash simple, todos los datos se organizan y se hash de una manera

lineal. El hashing simple se utiliza para un número fijo de elementos a hash, como elementos en un encabezado de bloque, pero también para verificar la integridad del bloque compuesto y no la integridad de cada elemento. La función hash SHA-256 se usa en bitcoin.

En un enfoque de árbol, un árbol de Merkle es un árbol binario, que consta de un conjunto de nodos con una gran cantidad de nodos de hoja en la parte inferior del árbol que contiene los datos subyacentes (si el número es impar, el último elemento se duplicará) , a conjunto de nodos intermedios donde cada nodo es el hash de sus dos hijos. Y, finalmente, un solo nodo raíz, también formado a partir del hash de sus dos hijos, que representa la "raíz" del árbol. En la Figura 11 se muestra un ejemplo de un árbol de Merkle.

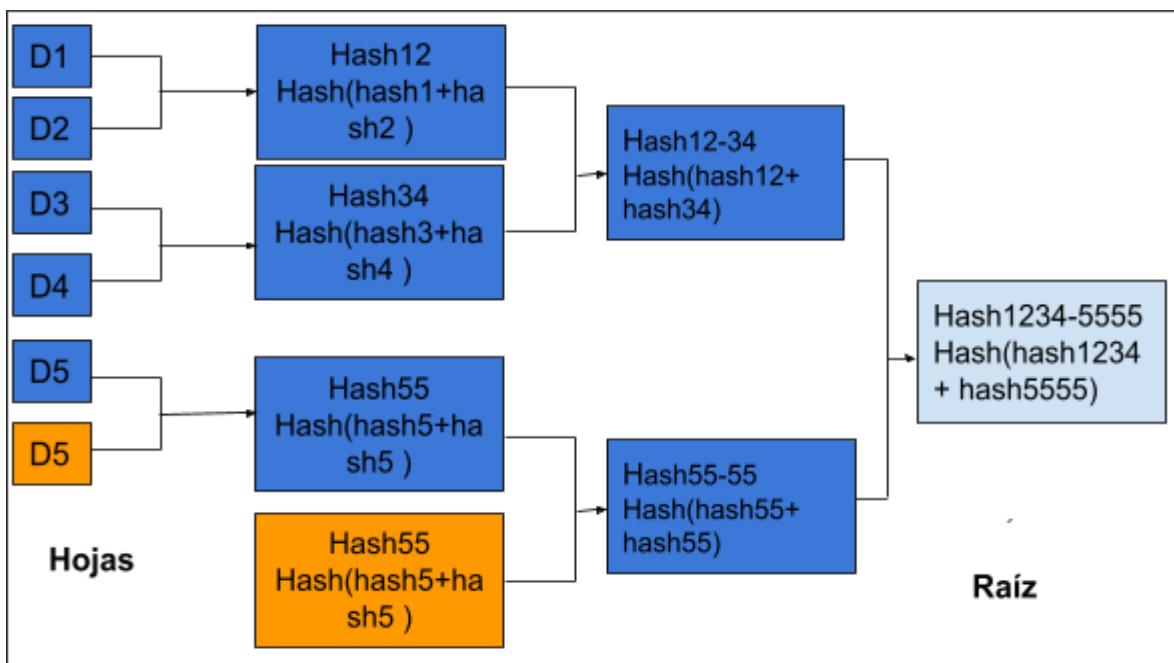


Figura 11. árbol de Merkle

Tenemos cinco datos para agrupar en forma de árbol de Merkle. Los datos D5 se duplican para formar los nodos intermedios. La raíz del árbol es, por tanto, el hash de todos los datos combinados en parejas.

El hash de árbol se utiliza para un número variable de elementos de un bloque a otro, por ejemplo, el número de transacciones. De hecho, cada bloque contiene miles y miles de transacciones. Lo que dificulta almacenar todos los datos en su interior. cada bloque en serie. Por lo tanto, la búsqueda de una transacción en particular es extremadamente tediosa y requiere mucho tiempo. El uso del árbol Merkle reduce en gran medida el tiempo necesario para determinar si una transacción en particular pertenece a un bloque o no.

Las funciones hash se utilizan para generar direcciones de cuentas, firmas digitales, hash de transacciones, hash de estado y hash de encabezado de bloque. El sistema descrito anteriormente tiene ciertos problemas. Es lento y produce un gran volumen de datos (al menos el doble del tamaño de la información original). Agregar una función hash unidireccional al proceso mejora este sistema. Esta función procesa una entrada de longitud variable para generar un elemento de longitud fija, es decir, 160 bits. Si cambia incluso un solo bit de datos, la función hash garantiza la producción de un valor de salida completamente diferente. Se utilizan principalmente dos técnicas para garantizar la seguridad de la cadena de bloques, así como la eficiencia de la validación y verificación de las transacciones: la criptografía de clave pública y las funciones hash.

4.3.3. Integridad de los datos

Un sistema distribuido como una cadena de bloques pública requiere la participación de una gran cantidad de actores para proporcionar propiedades de seguridad adecuadas para salvaguardar la integridad de los datos guardados por el libro mayor. El uso de la prueba de trabajo (Proof of Work – PoW) como mecanismo de seguridad para una cadena de bloques como Bitcoin conduce a un consumo de energía significativo, cuyo costo es absorbido por completo por menores de edad.

La combinación de funciones hash y criptografía de clave pública ayuda a administrar la integridad de la transacción al asegurar direcciones de cuenta únicas, autorizar la transacción por parte del remitente a través de una firma digital y verificar que el contenido de esa transacción no se modifique.

Las direcciones de cuenta se generan utilizando pares de claves públicas y claves privadas. Para ello, se genera un número aleatorio de 256 bits y se designa como clave privada. Se mantiene seguro y bloqueado mediante una frase de contraseña. Luego se aplica un algoritmo ECC a la clave privada para obtener una clave pública única. Finalmente se aplica una función hash a la clave pública para obtener la dirección de la cuenta. La dirección tiene un tamaño más corto, solo 20 bytes o 160 bits.

La clave privada se usa para firmar transacciones y la clave pública correspondiente se usa para verificar que el remitente coincide con el firmante de las transacciones. La firma digital incluye dos fases:

- Fase de firma: el remitente procesa los datos y los firma con la firma digital generada con su clave privada. Luego, el hash firmado se envía junto con los datos originales al destinatario.
- Fase de verificación: los datos firmados se descifran con la clave pública del remitente y se comparan con el valor hash de los datos originales.

Tenga en cuenta que, en ambas fases, la función hash utilizada debe ser la misma (por ejemplo, SHA256 para la cadena de bloques de Bitcoin).

La figura 12 ilustra estas dos fases. Para firmar digitalmente una transacción, calculamos el hash de los campos de datos de esta transacción. Este hash luego se cifra utilizando la clave privada del remitente de la transacción para autorizarla y hacerla no repudiable. La transacción es verificada por otras personas descifrando utilizando la clave pública de su remitente y recalculando el hash de la transacción: si el hash calculado y el hash recibido en la firma digital coinciden, se acepta la transacción; de lo contrario, se rechaza. Así, la firma y la verificación permiten autorizar la transacción y hacerla no repudiable y no modificable.

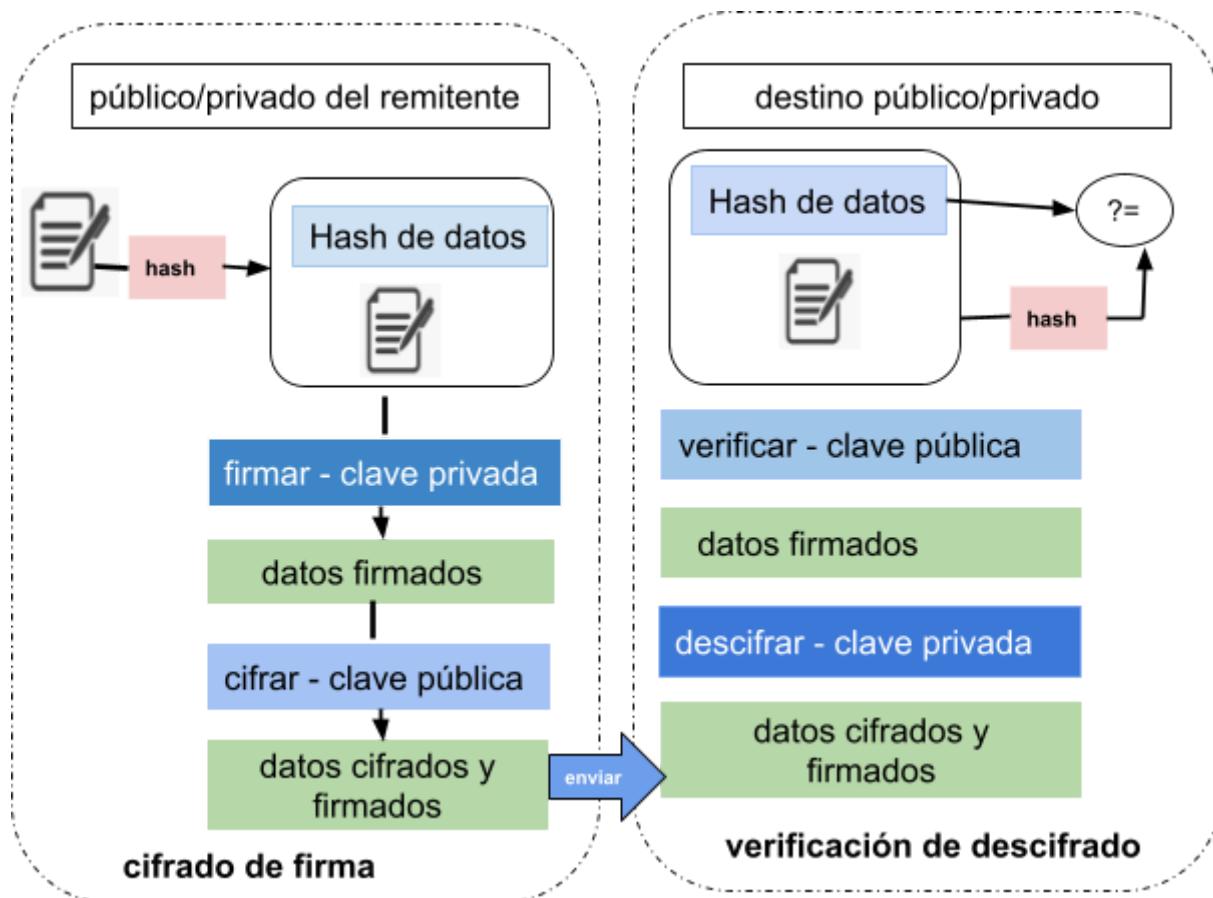


Figura 12. Criptografía asimétrica: fases de firma-cifrado y verificación-descifrado

La integridad de los bloques se gestiona asegurándose de que el contenido del encabezado del bloque no se altere, las transacciones no se alteren y las transiciones de estado se calculen, reduzcan y verifiquen de manera eficiente. Por lo tanto, las transacciones en un bloque son codificadas por el árbol Merkle. Cada cambio de estado requiere un nuevo cálculo del hash raíz del estado. En lugar de calcular el hash para todos los estados, solo se debe volver a calcular la ruta afectada en el árbol de Merkle. Por otro lado, el hash de bloque se obtiene calculando primero el hash de estado raíz, el hash de transacción raíz y luego el hash de recepción raíz. Estas raíces y todos los demás elementos del encabezado son hashes con los nodos variables para resolver el rompecabezas de la prueba de trabajo.

El hash de bloque tiene dos propósitos principales: verificar la integridad del bloque y de la transacción y formar el enlace incorporando el hash del bloque anterior en el encabezado del bloque actual. Si un nodo participante manipula el bloque, los valores de hash cambian, lo que provoca una discrepancia de valor de hash y hace que la cadena local del nodo quede en

un estado no válido. Cualquier bloqueo futuro iniciado por el nodo sería rechazado por otros mineros debido a la falta de coincidencia de hash. Hablamos de la inmutabilidad de la cadena de bloques.

4.4. Operaciones básicas en la Blockchain

La cadena de bloques es una red peer-to-peer descentralizada con dos tipos principales de nodos participantes. Por un lado, los participantes que inician la transferencia de valor creando una transacción y por otro lado, los participantes denominados mineros. Los mineros realizan trabajos o cálculos adicionales para verificar transacciones, transmitir una transacción. Compiten para reclamar el derecho a crear un nuevo bloque y lograr un consenso para validar un bloque. Los mineros también aseguran la difusión del bloque recién creado y la confirmación de las transacciones. Los bitcoins recompensados los alientan a ejecutar la cadena de bloques.

4.4.1. Validación de una transacción

La validación de transacciones se realiza de forma independiente por todos los mineros. El proceso en bitcoin implica validar más de 20 criterios que incluyen:

- la talla,
- la sintaxis,
- las referencias de la entrada y salida UTXO,
- la validez de las UTXO,
- las referencias de la cantidad de entrada y la cantidad de salida están suficientemente emparejadas.

Las transacciones no válidas se rechazan y no se transmitirán. Todas las transacciones válidas se agregan a un grupo de transacciones. Los mineros seleccionan un conjunto de transacciones de este grupo para crear un bloque. La transacción cero (el índice cero del bloque confirmado) es creada por el minero del bloque. Tiene un UTXO especial y no tiene entrada UTXO. Esto se llama la transacción "coinbase" que genera una tarifa de minero por crear el bloque.

4.4.2. Creación de un nuevo bloque (minería)

Blockchain es una cadena de flujos vinculada única y consistente. Sin embargo, si cada minero agrega el bloque a la cadena, la cadena tendrá muchas ramificaciones, lo que dará como resultado un estado inconsistente. Por lo tanto, un sistema para superar este desafío sería esencial. Los mineros compiten para crear un nuevo bloque y obtener la recompensa. De hecho, la minería es una operación que permite al minero encontrar un bloque válido resolviendo un problema matemático complejo y obtener ganancias.

Una vez que un minero resuelve el problema, el anuncio se transmite en la red y el bloque también se transmite en la red. Luego, otro minero verifica el nuevo bloque. Llegan a un consenso para agregar un nuevo bloque a la cadena. Este nuevo bloque se agrega a su copia local de la cadena de bloques. Por lo tanto, se registra y confirma un nuevo conjunto de transacciones. El algoritmo de consenso se llama protocolo de prueba de trabajo, ya que requiere muchos cálculos por parte del minero y consiste en encontrar el valor del campo Nonce que se ingresará en el encabezado del bloque para que el hash del encabezado del bloque da como resultado un resultado inferior a un cierto valor. Cuanto menor sea este valor, más difícil será resolver el problema. Para mantener la misma complejidad computacional en el tiempo, es interesante que Blockchain ajuste el nivel de dificultad. Este es el caso del proyecto Bitcoin que se basa en un promedio de minería de un bloque de 10 minutos.

El carácter probabilístico de esta selección también da lugar a la creación de divergencias (o “forks”) en la historia del Estado, que deben ser resueltas de manera unívoca para lograr rápidamente un consenso estable. Por ejemplo, los sistemas que usan PoW consideran la longitud de la cadena para decidir: la cadena más larga, que corresponde a transacciones validadas (con alta probabilidad) por la mayoría de la capacidad de cómputo del sistema, define el estado de consenso en cualquier momento. La convergencia de este proceso requiere un tiempo de latencia importante: este último no puede reducirse más allá de un cierto umbral acelerando la frecuencia de generación de bloques, ya que esto conduciría a un aumento en la ocurrencia de bifurcaciones. Esta limitación fundamental se deriva del teorema CAP y, por lo tanto, constituye un obstáculo para la escalabilidad de las cadenas de bloques.

Así, las operaciones principales en una cadena de bloques son la validación de transacciones y la creación de bloques con el consenso de los nodos participantes. Además, también hay muchas operaciones implícitas subyacentes en la cadena de bloques de bitcoin.

En resumen de este capítulo, el proyecto Bitcoin creado en 2008 con el objetivo de intercambiar criptomonedas (BTC) en una red descentralizada sin utilizar un tercero de

confianza está en el origen de la tecnología Blockchain. Blockchain a menudo se compara con un gran libro de cuentas que es accesible públicamente y verificable. Sus miembros (los nodos) pueden agregarle escrituras, pero esta operación requiere la validación de varios miembros del grupo, o incluso de la mayoría del grupo.

La apertura del código fuente de Bitcoin ha permitido la generación de varias “Altcoins” ofreciendo una diversidad de aplicaciones de la tecnología Blockchain. La evolución de la arquitectura y la adición, la función también hace posible converger en varios tipos de Blockchain, incluidas las cadenas de bloques públicas, privadas y autorizadas.

Los mecanismos de seguridad como la criptografía asimétrica y las funciones hash se utilizan en varios procesos de Blockchain para garantizar la integridad de las transacciones y hacer que los datos sean inviolables.

En 2014, la fundación sin ánimo de lucro Ethereum se embarcó en el proyecto de extender el principio Blockchain a una Blockchain programable, abriendo así el campo a todo tipo de transacciones (smart contracts) y nuevos servicios. En el próximo capítulo, veremos esta cadena de bloques en detalle.

Capítulo 5

5. La blockchain de Ethereum

Bitcoin es la madre de todas las cadenas de bloques. Estaba destinado a la transferencia de valor entre pares sin ningún intermediario. Alrededor de 2013, se introdujo un marco para la ejecución de código para dar a luz a una nueva cadena de bloques, llamada Ethereum, que permite una gran cantidad de aplicaciones que no se limitan solo a la transferencia de dinero. Como resultado, abre muchas perspectivas en el campo descentralizado.

De hecho, en la cadena de bloques de Bitcoin, la base de datos distribuida está diseñada como una tabla de saldos de cuentas considerada como un libro mayor y las transacciones son transferencias de divisas que facilitan la gestión financiera sin ningún tipo de confianza entre las partes. Sin embargo, a medida que los desarrolladores y tecnólogos dieron importancia a bitcoin, nuevos proyectos comenzaron a usar la red de bitcoins para fines distintos de la transferencia de valor. Si bien un buen número de ellos se ha inclinado por las "alt-coins" (cadenas de bloques distintas con sus propias criptomonedas, que mejoran la protocolo original de Bitcoin para agregar nuevas características o capacidades), el inventor de Ethereum, Vitalik Buterin, está impulsado por el deseo de ofrecer una cadena de bloques reprogramable para realizar cálculos arbitrariamente complejos que pueden abarcar muchos otros proyectos. Así, publicó el libro blanco de Ethereum [31] en noviembre de 2013, en el que describe en detalle el diseño técnico y la razón de ser del protocolo Ethereum, pero también la arquitectura de los contratos inteligentes. Según Buterin, Ethereum se fusiona y mejora conceptos de secuencias de comandos (como en bitcoin) y "alt-coins" para permitir a los desarrolladores crear aplicaciones basadas en un consenso arbitrario que proporciona la escalabilidad, estandarización, facilidad de desarrollo e interoperabilidad que ofrecen estos diferentes paradigmas. Y para hacer esto, Ethereum integra la última capa fundamental abstracta, una Blockchain con un lenguaje de programación completo, que permite a todos escribir contratos inteligentes y aplicaciones descentralizadas en las que pueden crear sus propias reglas de propiedad, sus propios formatos de función y transición de estado. Más tarde, en enero de 2014, Etherik anunció oficialmente Ethereum [27] durante una

conferencia de bitcoin en Miami, Florida, EE. UU. Además, Vitalik también comenzó a trabajar con el Dr. Gavin Wood, quien publicó, en abril de 2014, el libro Ethereum yellow, el libro de especificaciones técnicas de la máquina virtual Ethereum (EVM). Los detalles de estas especificaciones muestran varias implementaciones del cliente. Ethereum, en unos siete lenguajes de programación (C++, Go, Python, Java, JavaScript, Haskell, Rust).

Inspirándose en Bitcoin, Ethereum opera en una red descentralizada de igual a igual que permite comunicaciones basadas en el consenso y elimina a terceros confiables.

Pero algunos elementos técnicos los diferencian, en particular las nociones de cuenta, mensaje, un nuevo marco de ejecución basado en contratos inteligentes en lugar de scripts y un modelo particular de incentivos mineros. Así, veremos en detalle estos diferentes conceptos en este capítulo.

5.1. Cuentas, mensajes y transacciones.

A diferencia de Bitcoin, Ethereum es como una máquina de estado donde una nueva transacción se mueve de un estado a otro. El estado global de Ethereum está compuesto por muchas cuentas capaces de interactuar entre sí a través de transacciones o mensajes.

5.1.1. Las cuentas

Ethereum introduce formalmente el concepto de cuenta como parte de su protocolo. La cuenta es el iniciador y el destino de una transacción. Una transacción actualiza directamente los saldos de las cuentas, en lugar de mantener el estado, como en los UTXO de bitcoin, y permite la transmisión de valor, mensajes y datos entre cuentas que pueden resultar en transiciones de estado. Cada cuenta tiene asociado un estado y una dirección de 20 bytes (es decir, 160 bits) que permite identificarla. Hay dos tipos de cuentas:

- **Cuentas de propiedad externa o EOA ("Externally Owned Accounts")** que están controladas por claves privadas y no están asociadas con ningún código. Se requiere una cuenta externa para participar en la red Ethereum. Interactúa con la cadena de bloques al crear y firmar una transacción utilizando su clave privada. Una transacción entre dos cuentas externas es simplemente una transferencia de valor. Pero una transacción entre una cuenta externa y una cuenta de contrato activa el código de la cuenta de contrato, permitiéndole

realizar varias acciones (por ejemplo, transferir tokens, escribir en la memoria interna, crear nuevos tokens, realizar cálculos, crear nuevos contratos, etc.).

- Cuentas de contrato o CA (“Contract Account”), las cuales están controladas por su código de contrato y están asociadas a un código. Representan un contrato inteligente (consulte la sección de contratos inteligentes a continuación) y solo pueden ser activados por un EOA. Las cuentas de contrato no pueden iniciar nuevas transacciones por sí mismas. Sin embargo, solo pueden desencadenar transacciones para responder a otras transacciones que hayan recibido (de una cuenta de propiedad externa o de otra cuenta de contrato).

5.1.1.1. Estado de la cuenta

El estado de una cuenta de Ethereum se compone (independientemente de su tipo) de cuatro campos (ver Figura 12):

- ➔ El **nonce**: si la cuenta es una cuenta externa, el nonce representa la cantidad de transacciones enviadas desde la dirección de la cuenta. Si se trata de una cuenta de contrato, el nonce es el número de contratos creados por la cuenta.
- ➔ El **saldo de la cuenta** corriente expresado en Wei (1 Ether = 1018 Wei),
- ➔ La **storageRoot**: este es el hash de la raíz del árbol Merkle. Esta raíz se obtiene del hash de los contenidos de almacenamiento de la cuenta. StorageRoot está vacío de forma predeterminada.
- ➔ Y finalmente el **codeHash** que representa el hash del código de cuenta en la máquina virtual Ethereum (EVM). Para una cuenta de contrato, el código de contrato se codifica y almacena como codeHash y para una cuenta externa, codeHash es el hash de la cadena vacía.

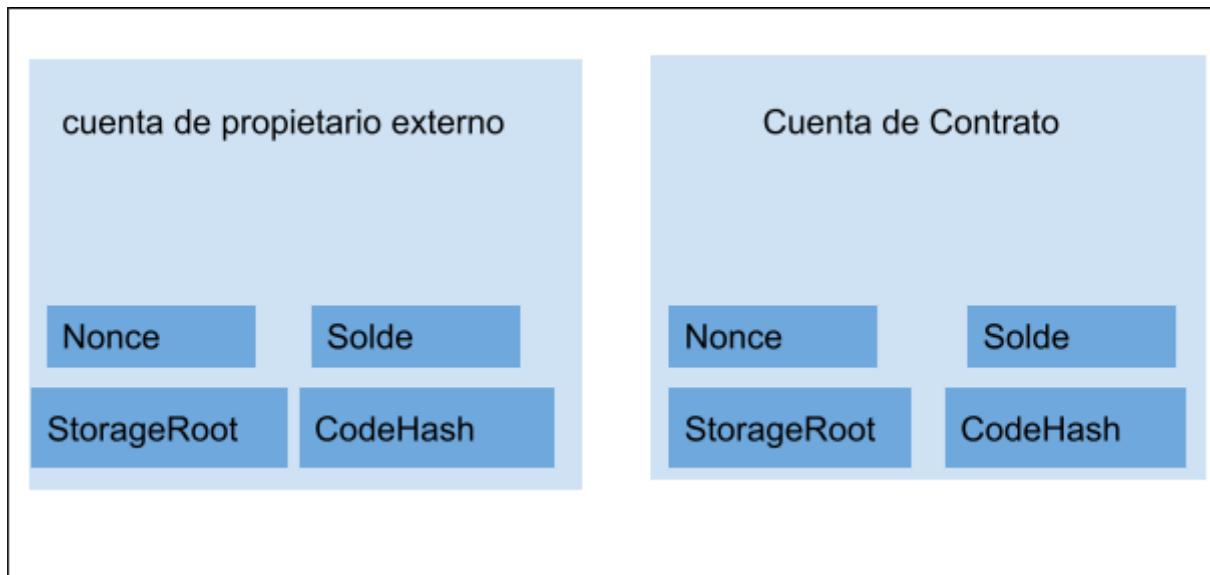


Figura 13. El estado de una cuenta en Ethereum

5.1.1.2. El estado general

El estado general de la cadena de bloques de Ethereum se considera un mapeo entre direcciones de cuenta y estados de cuenta. Este mapeo se almacena en una estructura de datos llamada árbol de Merkle. Este árbol debe tener una clave para cada valor almacenado en él. Comenzando desde el nodo raíz del árbol, la clave debe decirle qué nodo secundario debe seguir para obtener el valor correspondiente, almacenado en los nodos hoja. En el caso de Ethereum, el mapeo de clave/valor del árbol de estado es entre direcciones y sus cuentas asociadas, incluido el saldo, nonce, codeHash y storageRoot para cada cuenta (donde storageRoot en sí es la raíz de otro árbol).

Al igual que en Bitcoin, la Blockchain de Ethereum es administrada por un grupo de nodos. En términos generales, hay dos tipos de nudos: nudos completos (también llamados menores) y nudos ligeros (nudos simples). Un nodo completo sincroniza la cadena de bloques descargando la cadena completa, desde el bloque de génesis hasta el bloque actual, ejecutando todas las transacciones contenidas en él. Por lo general, los mineros almacenan todo el archivo completo, ya que son necesarios para el proceso de minería. Es posible que un nodo completo no ejecute todas las transacciones. Pero a menos que un nodo necesite ejecutar cada transacción o consultar fácilmente datos históricos, realmente no hay necesidad para almacenar toda la cadena. Aquí es donde entra el concepto de nodo ligero. En lugar de

descargar y almacenar la cadena completa y ejecutar todas las transacciones, los nodos ligeros solo descargan los encabezados de la cadena, desde el bloque de génesis hasta el estado actual, sin ejecutar ninguna transacción u operación. Dado que estos nodos tienen acceso a los encabezados de bloque, siempre pueden generar y obtener fácilmente respuestas verificables sobre transacciones, eventos, saldos, etc.

Esto es posible porque en el árbol de Merkle los datos se propagan hacia arriba. De hecho, si un usuario malicioso intenta intercambiar una transacción falsa en la parte inferior del árbol, esta modificación provocará una modificación del hash del nodo que está arriba, lo que cambiará el hash del nodo superior, y así sucesivamente, hasta que finalmente cambie la raíz del árbol. Por lo tanto, usar un árbol de Merkle tiene la ventaja de que el nodo raíz de esta estructura depende criptográficamente de los datos almacenados en el árbol. Por lo tanto, el hash del nodo raíz se puede utilizar como una identidad segura para estos datos.

Las cuentas gobiernan el estado de Ethereum Blockchain. Se comunican a través de transacciones. La naturaleza del iniciador de la transacción afecta el tipo de transacción.

5.1.2. Mensaje y Transacción

La intervención de una transacción permite cambiar el estado global de la cadena de bloques de Ethereum. En este sentido, una transacción es una instrucción firmada con funciones criptográficas que es generada por una cuenta externa, serializada y luego enviada a la cadena de bloques. Existen dos tipos de transacciones: llamadas de mensajes y creaciones de contratos (es decir, transacciones que crean nuevos contratos de Ethereum). Todas las transacciones contienen los siguientes componentes, independientemente del tipo (Figura 13):

- **El nonce:** número de transacciones enviadas por el remitente.
- **El precio del “gas”** (gasPrice): El número de Wei que el remitente está dispuesto a pagar por unidad de “gas” necesaria para ejecutar la transacción.
- **El límite de “gas”** (gasLimit): cantidad máxima de “gas” que el remitente está dispuesto a pagar por la ejecución de esta transacción. Esta cantidad se define y paga desde el principio, antes de cualquier cálculo.
- **La dirección del destinatario.** En una transacción de creación de contrato, la dirección de la cuenta del contrato aún no existe, por lo que se utiliza un valor vacío.

- **Un campo de valor (Valor)**: la cantidad (en Wei) que se transferirá del remitente al destinatario. En una transacción de creación de contrato, este valor sirve como saldo inicial en la cuenta de contrato recién creada.
- **v, r, s**: tres variables utilizadas para generar la firma que identifica al remitente de la transacción.
- **Un campo de inicio** (solo para transacciones que crean un contrato) que es un fragmento de código utilizado para inicializar la nueva cuenta de contrato. Se ejecuta una vez y luego se ignora. Cuando 'init' se ejecuta por primera vez, devuelve el cuerpo del código de cuenta, que es la parte del código asociada permanentemente con la cuenta del contrato.
- Datos (campo opcional que solo existe para llamadas de mensajes): Estos son los datos de entrada (es decir, parámetros) de la llamada de mensajes. Por ejemplo, si un contrato inteligente sirve como un servicio de registro de estudiantes, una llamada a ese contrato puede esperar campos de entrada como apellido, nombre, etc.



Figura 14. Componentes de una transacción de Ethereum

Generalmente una transacción proviene de una cuenta externa de la red. Pero eso no significa que los contratos no puedan comunicarse entre sí. Un contrato envía "mensajes" (o transacciones internas) a otros contratos. Entonces un mensaje puede ser considerado una transacción, excepto que no sea generada por una cuenta perteneciente a un tercero. En cambio, los mensajes son generados por un contrato. Son objetos virtuales que, a diferencia de las transacciones, no están serializados y solo existen dentro del entorno de tiempo de ejecución de Ethereum. Cuando un contrato envía una transacción interna a otro contrato, se ejecuta el código asociado a la cuenta de contrato del destinatario. Por lo tanto, debe tenerse en cuenta que las transacciones internas no contienen un gasLimit. De hecho, el límite de "gas" lo determina el originador externo de la transacción original (es decir, una cuenta externa). El límite de gas de todas las cuentas externas debe ser lo suficientemente alto para ejecutar la transacción.

5.1.3. Ejecución de una transacción

Ejecutar una transacción en Ethereum pasa por un proceso muy complejo antes de incluir la transacción en la cadena. Por lo tanto, se deben proporcionar una serie de requisitos para que se ejecute una transacción. Estos requisitos son, entre otros:

- La transacción debe tener un prefijo de longitud recursivo (PLR o RLP - Recursive Length Prefix – en inglés) con el formato correcto. Un PLR es un formato de datos que se utiliza para codificar matrices anidadas de datos binarios. Se utiliza en Ethereum para serializar objetos.
- La transacción debe tener una firma válida.
- El nonce de la transacción también debe ser válido. Como recordatorio, el nonce de una cuenta es el número de transacciones enviadas desde esta cuenta. El nonce de una transacción es válido si es igual al nonce de la cuenta del remitente.
- El límite de gas de la transacción debe ser igual o mayor que el gas intrínseco utilizado por la transacción. El gas intrínseco se calcula de la siguiente manera:

```
Gas intrínseco = X + 4 Y + 68 Z + U
Con :
X : El precio del "gas" predefinido = 21000
Y : El número de bytes de datos o código igual a cero
Z : El número de datos distintos de cero o bytes de
código

U : Cuotas adicionales:
    32000 USD si es una transacción
    de creación de contrato
    0 si no
```

- El saldo de la cuenta del remitente debe tener suficiente Ether para cubrir los costos iniciales de gas que debe pagar el remitente. Calcular el costo inicial del gas es simple: primero, el límite de gas (GasLimit) de la transacción se multiplica por el precio del gas (GasPrice) de la transacción para determinar el costo máximo del gas. Luego, este costo máximo se agrega al valor total transferido del remitente al receptor.

Una vez verificadas y validadas todas estas condiciones, se descuenta del saldo del remitente el costo de ejecución inicial y se incrementa en 1 su nonce para tener en cuenta la transacción actual. Así, el "gas restante" puede obtenerse disminuyendo el "gas" intrínseco del GasLimit total de la transacción. Aquí es donde la transacción comienza a ejecutarse.

Por un lado, Ethereum realiza un seguimiento del subestado a lo largo de la ejecución de la transacción. Este subestado se caracteriza por:

- **Un conjunto de autodestrucción:** el conjunto de cuentas (si las hay) que se eliminarán una vez que se complete la transacción.
- **Una serie de registros:** puntos de control archivados de ejecución de código en la máquina virtual.
- **Saldo de reembolso:** el monto que se reembolsará a la cuenta del remitente después de la transacción. De hecho, el almacenamiento en Ethereum es costoso y, por lo tanto, se remunera su limpieza (se reembolsa al remitente la

limpieza del almacenamiento). Por lo tanto, se crea un contador de reembolso, se inicializa a cero y se incrementa cada vez que el contrato elimina algo almacenado.

Por otro lado, se procesan las diferentes instrucciones de la transacción. Una vez que se han procesado todos los pasos requeridos por la transacción y suponiendo que no hay estados inválidos, el estado se finaliza determinando la cantidad de gas no utilizado que se devolverá al remitente. Además del gas no utilizado, al remitente también se le reembolsa una parte de la asignación del "saldo de reembolso". Después de hacer el reembolso:

- Ethereum para gas se le da al minero,
- El gas utilizado por la transacción se agrega al contador de gas del bloque (que realiza un seguimiento del gas total utilizado por todas las transacciones en el bloque y es útil al confirmar un bloque),
- Y todas las cuentas en el conjunto de autodestrucción (si las hay) se eliminan.

Estas son las diferentes etapas de la ejecución de las transacciones. Sin embargo, este proceso difiere dependiendo del tipo de transacción (transacciones que crean contratos y llamadas de mensajes).

5.1.3.1. Ejecutar una transacción de creación de contrato

La transacción de creación de contrato crea una nueva cuenta de contrato. Para ello, la dirección de la cuenta se declara mediante una fórmula especial. La inicialización de la cuenta se realiza siguiendo estos pasos:

- Establecer el nonce a cero,
- Si el remitente envió una cierta cantidad de Ether como valor con la transacción, establezca el saldo de la nueva cuenta con este valor.
- Deducir el valor agregado al saldo de esta nueva cuenta del saldo del remitente,
- Establecer el almacenamiento como vacío,
- Establezca el codeHash del contrato como el hash de una cadena vacía,

Una vez que se inicializa la cuenta, la cuenta se crea ejecutando el código de inicio enviado con la transacción. Lo que sucede durante la ejecución de este código de inicio es

variado. Según el creador de contratos, puede actualizar el almacenamiento de la cuenta, crear otras cuentas de contrato, realizar otras llamadas de mensajes, etc.

Cuando se ejecuta el código para inicializar un contrato, utiliza gas. No se permite que la transacción use más gas que el gas restante. Si es así, la ejecución encuentra una excepción de falta de combustible y finaliza.

Si la transacción finaliza debido a una excepción de no consumo, el estado se revierte al punto inmediatamente anterior a la transacción. El remitente no es reembolsado por el gas que se gastó antes de que se agotara. Sin embargo, si el remitente envió valor de Ether con la transacción, el valor se reembolsará incluso si falla la creación del contrato.

Si el código de inicialización se ejecuta con éxito, se paga un costo de creación de contrato final. Este es un costo de almacenamiento, proporcional al tamaño del código de contrato creado. Si no queda suficiente gas para pagar este costo final, la transacción declara nuevamente una excepción de falta de gas y finaliza.

Si todo va bien y llegamos tan lejos sin excepción, cualquier gas restante no utilizado se devuelve al remitente original de la transacción y ahora se permite que persista el estado modificado.

5.1.3.2. Ejecución de una transacción de invocación de mensaje

Realizar una llamada de mensaje es similar a realizar la creación de un contrato, con algunas diferencias. Esta ejecución no incluye ningún código de inicialización porque no se crea ninguna cuenta nueva. Sin embargo, puede contener datos de entrada, si estos datos fueron proporcionados por el remitente de la transacción. Una vez ejecutadas, las llamadas de mensajes también tienen un componente adicional que contiene los datos de salida, que se utiliza si una ejecución posterior necesita estos datos.

Como es el caso con la creación de un contrato, si la ejecución de una llamada de mensaje finaliza porque se queda sin "gas" o la transacción no es válida (por ejemplo, desbordamiento de pila, destino de salto no válido o instrucción no válida), el "gas" utilizado no se devuelve al remitente. En su lugar, se consume cualquier gas restante sin usar y el estado se restablece al punto inmediatamente anterior a la transferencia de saldo.

La tarifa de "gas" involucrada en la ejecución de transacciones es un incentivo para que los mineros mantengan la consistencia del estado global de Ethereum.

5.2. El modelo de incentivos mineros.

La cadena de bloques de Ethereum es similar en muchos aspectos a la cadena de bloques de Bitcoin, aunque tiene algunas diferencias. La principal diferencia entre Ethereum y Bitcoin en lo que respecta a la arquitectura de cadena de bloques es que, a diferencia de Bitcoin, los bloques de Ethereum contienen una copia de la lista y el estado de transacciones más recientes. Además de esto, el bloque contiene otros dos valores, el número de bloque y la dificultad. El algoritmo que da sentido al concepto de dificultad de un bloque se denomina Prueba de Trabajo (PoW – Prof Of Work). El algoritmo de prueba de trabajo utilizado se llama Ethash y consiste en buscar una entrada nonce en el algoritmo para que el resultado esté por debajo de un determinado umbral de dificultad. Este algoritmo hace posible, en cierto modo, calcular el mixHash y el bloque nonce. De hecho, estos dos componentes que se encuentran en un bloque Ethereum están técnicamente vinculados:

- El mixHash de un bloque es un hash que, combinado con el nonce, demuestra que este bloque ha realizado suficientes cálculos.
- El nonce de un bloque es un hash que, combinado con el mixHash, demuestra que este bloque ha realizado suficientes cálculos.

El cálculo de estos dos componentes por el algoritmo PoW es bastante complejo pero se puede resumir, a un alto nivel, de la siguiente manera:

Se calcula una “semilla” para cada bloque. Esta semilla es diferente para cada “época”, donde cada época tiene 30.000 bloques. Para la primera época, la semilla es el hash de una serie de ceros de 32 bytes. Para cada época posterior, este es el hash del hash anterior. Usando esta semilla, un nodo puede calcular un "caché" pseudoaleatorio.

Este caché es utilizado por nodos livianos para la verificación eficiente de una transacción sin la sobrecarga de almacenar todo el conjunto de datos de la cadena de bloques. Por lo tanto, un nodo ligero puede comprobar la validez de una transacción basándose únicamente en este caché, ya que este puede regenerar el bloque específico que se va a comprobar.

Luego, los mineros pueden tomar porciones aleatorias del conjunto de datos y pasárselas por una función matemática para combinarlas en un "mixHash". Un minero generará repetidamente un mixHash hasta que la salida esté por debajo del objetivo deseado. Cuando la salida cumple con este requisito, este nonce se considera válido y el bloque se puede agregar a la cadena.

La prueba de trabajo que realizan los mineros garantiza la seguridad del protocolo Ethereum pero también les permite enriquecerse.

5.2.1. Operación Minería y Seguridad

En general, el objetivo de PoW es probar, de forma criptográfica y segura, que se utilizó una cantidad particular de cómputo para generar una salida (es decir, el nonce). De hecho, no hay mejor manera de encontrar un nonce por debajo del umbral requerido que enumerar todas las posibilidades. Los resultados de la aplicación repetida de la función hash tienen una distribución uniforme, por lo que podemos estar seguros de que, en promedio, el tiempo necesario para encontrar dicho elemento depende del umbral de dificultad. Cuanto mayor sea la dificultad, más tiempo lleva resolver el problema. De esta forma, el algoritmo PoW da sentido a la noción de dificultad, que se utiliza para hacer cumplir la seguridad de la Blockchain.

De hecho, la seguridad de blockchain se basa en la confianza de los usuarios (miembros de la red). Si existiera más de una cadena, los usuarios perderían la confianza porque no podrían determinar razonablemente qué cadena era la cadena "válida". En este sentido, para que un grupo de usuarios acepte el estado subyacente que se almacena en una cadena de bloques, es necesario tener una única cadena canónica en la que todos tengan confianza. Esto es exactamente lo que hace el algoritmo PoW: garantiza que una determinada cadena de bloques seguirá siendo canónica en el futuro, lo que dificulta enormemente que un atacante cree nuevos bloques que sobrescriben parte del historial (por ejemplo).

ejemplo, eliminando transacciones o creando transacciones falsas) o manteniendo una "bifurcación". Para que su bloque sea validado primero, un atacante debe resolver el problema de manera constante más rápido que cualquier otra persona en la red, por lo que la red cree que su cadena es la cadena más pesada. Esto sería imposible a menos que el atacante tenga más de la mitad del poder de minería [17] de la red, un escenario conocido como ataque del 51%.

5.2.2. Operación Minería y Riqueza

Además de proporcionar una cadena de bloques segura, la prueba de trabajo también es un medio para distribuir la riqueza entre quienes usan sus cálculos para brindar esta seguridad. Recuerda que un minero recibe una recompensa por minar un bloque, que incluye:

- una recompensa de bloque estático de 5 Ether por el bloque "ganador" (que se reduce a 3 Ether con las actualizaciones),
- un costo del gas gastado en el bloque por las transacciones incluidas en el bloque,
- una recompensa adicional por incluir miembros en el bloque.

Para garantizar el uso a prueba de futuro del mecanismo de consenso de prueba de trabajo para la seguridad y la distribución de la riqueza, Ethereum [25] se esfuerza por inculcar estas dos propiedades:

- Hazlo accesible a tantas personas como sea posible. En otras palabras, los usuarios no deberían necesitar hardware especializado o inusual para ejecutar el algoritmo. El objetivo es hacer que el modelo de distribución de la riqueza sea lo más abierto posible, de modo que cualquiera pueda contribuir con cualquier cantidad de poder de cómputo a cambio de Ether.
- Reducir la posibilidad de que un solo nodo (o pequeño grupo) genere una ganancia desproporcionada. Cualquier nodo que pueda generar una ganancia desproporcionada significa que el nodo tiene una gran influencia en la determinación de la cadena de bloques canónica. Esto es un inconveniente ya que reduce la seguridad de la red.

En la red Bitcoin Blockchain, uno de los problemas con las dos propiedades anteriores es que el algoritmo PoW es una función hash SHA256. La debilidad de este tipo de funciones es que se pueden resolver de manera mucho más eficiente utilizando hardware especializado, también conocido como ASIC. Para mitigar este problema, Ethereum optó por hacer que su algoritmo Ethash sea secuencialmente duro para la memoria. Esto significa que el algoritmo está diseñado de tal manera que calcular el nonce requiere mucha memoria y ancho de banda. Los grandes requisitos de memoria dificultan que una computadora use su memoria en paralelo para detectar múltiples nonces simultáneamente, y los requisitos de alto ancho de banda dificultan que incluso una computadora súper rápida detecte múltiples nonces simultáneamente. Esto reduce los riesgos de centralización y crea un campo de juego más nivelado para los nodos que realizan la verificación.

Basado en la cadena de bloques de Bitcoin, Ethereum se parece a este último pero con sus propias especificaciones. Ethereum permite transacciones que pueden conducir a

operaciones más sofisticadas. Por ejemplo, una transacción puede requerir una transferencia condicional, una evaluación, una firma, etc. Esto es posible gracias a los contratos inteligentes.

5.3. Contratos inteligentes

El término contrato inteligente (en inglés, *smarts contracts*) fue introducido en 1997 por Nick Szabo. Un contrato inteligente es un programa de código identificado por una dirección en la red Blockchain. Ethereum es una de las tecnologías preferidas para el desarrollo de contratos inteligentes. Los componentes principales de las transacciones se basan en la máquina de estado y las funciones. Es una plataforma de procesamiento y cumplimiento de contratos. Turing-complete basado en el libro mayor compartido descentralizado de Blockchain. El diseño e implementación de Ethereum es completamente independiente de la criptomoneda Bitcoin. Cada transacción tiene parámetros de entrada que son requeridos por una función en el contrato. Durante la ejecución de una función, el estado de las variables de estado se modifica de acuerdo con la configuración de implementación lógica. El código de contrato inteligente está escrito en lenguajes de alto nivel como Solidity, Serpent o LLL (List Like Language). El código se compila en bytecode utilizando compiladores como Solidity o Serpent. El programador puede crear formatos de transacción, transiciones de estado, funciones de eventos y reglas de propiedad. El código del software se ejecuta en una máquina virtual llamada máquina virtual Ethereum.

Los contratos inteligentes permiten a las contrapartes automatizar las tareas de transacción que normalmente se realizan manualmente y requieren la intervención de intermediarios externos. La tecnología de contrato inteligente puede conducir a procesos más rápidos, más precisos y más rentables. Por lo tanto, los contratos inteligentes cubren una gran cantidad de áreas de aplicación contractual que pueden beneficiarse de una mayor confiabilidad, un procesamiento de transacciones más rápido, costos reducidos y menos pasos de procesos manuales a través de intermediarios.

Es en este sentido que lo adoptamos para asegurar un Dapp inmobiliario. Dicha elección está motivada, además, por las características únicas de blockchain como habilitador de aplicaciones más confiables, a prueba de manipulaciones y resistentes a averías. Además, gracias a los contratos inteligentes, la cadena de bloques de Ethereum incorpora una capa de lógica y cálculo en su infraestructura de confianza. Esto le da una apertura a un número potencial de aplicaciones descentralizadas.

5.3.1. Estructura de un contrato inteligente

Puede considerarse la pieza central del empuje de la cadena de bloques de Ethereum, un contrato es una colección de código (sus funciones) y datos (su estado) que residen en una dirección específica en la cadena de bloques de Ethereum. Este conjunto se ilustra en la Figura 15 y muestra claramente la diferencia entre el mensaje y la transacción.

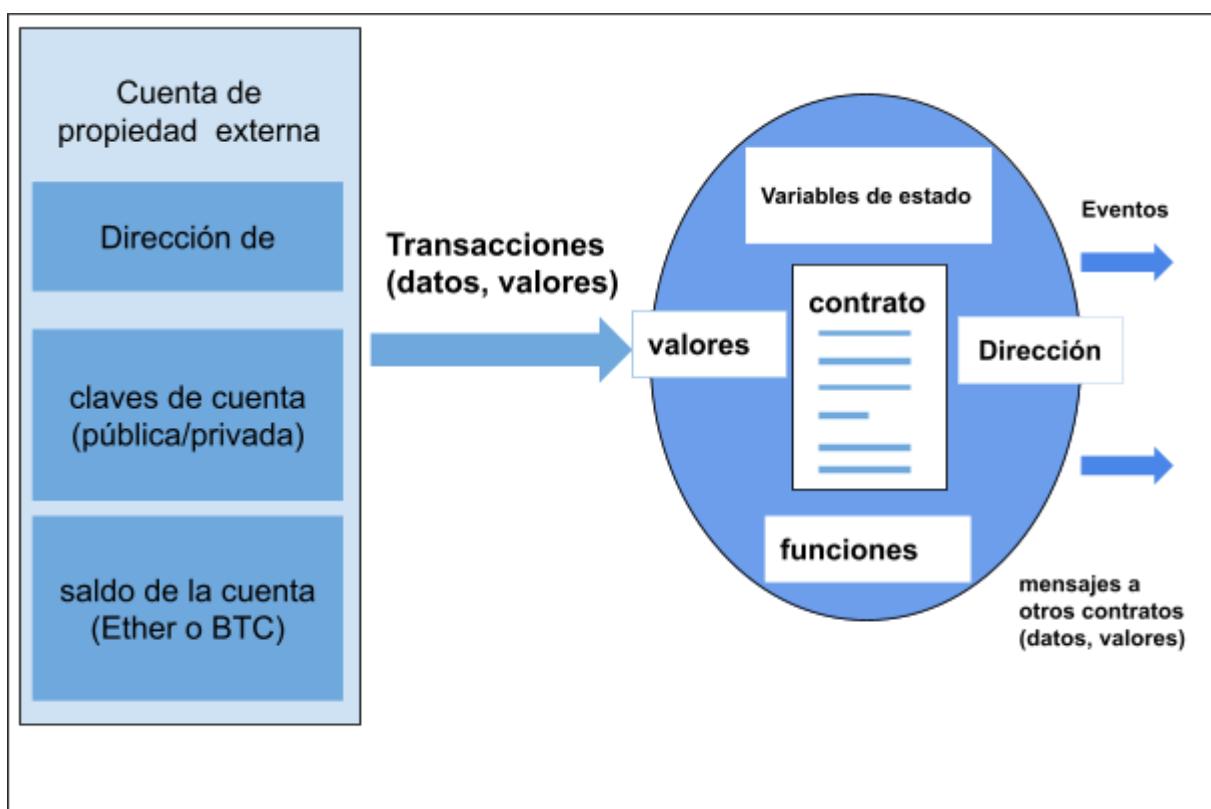


Figura 15. Estructura general de un contrato inteligente

En efecto, los conteos de contratos pueden pasar mensajes entre ellos y realizar virtualmente el cómputo completo de Turing. Los componentes principales del contrato inteligente son variables de estado, funciones, modificadores y eventos. Los contratos rigen el comportamiento de las cuentas dentro del estado Ethereum.

Solidity es un lenguaje orientado a objetos de alto nivel para implementar contratos inteligentes. Es un lenguaje tipificado estáticamente que admite herencia, bibliotecas y tipos complejos definidos por el usuario, entre otras características. Estructuralmente, un contrato inteligente se parece a una definición de clase en el diseño orientado a objetos. Contiene datos, funciones o métodos con modificadores públicos o privados, así como funciones getter

y setter. Veamos un contrato inteligente de Solidity simple en la siguiente figura para comprender su estructura.

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity >=0.4.22 <0.9.0;
3
4 contract Mia {
5   struct Contract{
6     string numeroContract;
7     string idProprety;
8     string ownerLand;
9     string buyerLand;
10    uint cost;
11    string desc;
12    string date;
13  }
14  address public owner;
15  uint public PropretyCount = 0;
16
17  mapping(uint => Contract) public Propreties;
18
19  constructor() public{
20    owner = msg.sender;
21  }
22
23  event Add(
24    string numeroContract,
25    string idProprety,
26    string ownerLand,
27    string buyerLand,
28    uint cost,
29    string desc,
30    string date
31  );
32
33
34
35
36
37
38
39  function addLand( string memory numeroContract,
40                    string memory idProprety,
41                    string memory _ownerLand,
42                    string memory _buyerLand,uint _cost,
43                    string memory _desc,
44                    string memory _date)
45  public isOwner {
46    require(_cost > 0, 'Must be a cost');
47    PropretyCount++;
48
49    Propreties[PropretyCount] = Contract( numeroContract,
50                                         idProprety,_ownerLand,
51                                         _buyerLand, _cost,
52                                         _desc,_date);
53
54    emit Add( numeroContract,
55              idProprety,_ownerLand,
56              _buyerLand, _cost,
57              _desc,_date);
58  }
59
60
61
62  function getNumberLands() public view returns(uint){
63    return PropretyCount;
64  }
65
66
67

```

Figura 16. Ejemplo de un contrato inteligente

La primera línea con pragma indica la versión de idioma de solidity 0.8.14 o cualquier otra versión más reciente (hasta la versión 0.9.0). Esto asegura que el contrato no sea compilable con una nueva versión del compilador, donde podría comportarse de manera diferente. El pragma es la instrucción que le dice a los compiladores cómo procesar el código fuente del contrato.

El nombre del contrato **MIA** está precedido por la palabra clave **contract**. Este contrato en particular es para el almacenamiento de un contrato vendido. Los datos para el número de contrato, el id del contrato, el propietario; quien va a comprar esta vivienda; desc y date se definen con el tipo cadena y para el precio con el tipo uint. Se definen dos funciones para completar el numeroContract; idProprety; ownerLand; buyerLand; desc; date; del contrato getNumberLands y para leer su precio PropretyCount.

5.3.2. Ejecución e implementación de un contrato inteligente

La ejecución de un contrato inteligente se inicia mediante un mensaje incrustado en la transacción, por ejemplo, una solicitud de transferencia de moneda digital, una simple suma y resta. Cada nodo en la red de Ethereum debe poder ejecutar código independientemente del tipo de hardware subyacente o del sistema operativo subyacente que se ejecute en la máquina virtual de Ethereum (EVM). Un contrato inteligente, escrito en un lenguaje de programación de alto nivel, se traduce a código de bytes y luego se implementa en el EVM. Cada nodo albergará los mismos códigos de contrato inteligente que en el EVM. Un nodo Ethereum es un sistema informático que representa una entidad comercial o un participante individual. Un nodo completo de Ethereum alberga el software necesario para el inicio de transacciones, la validación, la extracción, la creación de bloques y la ejecución de contratos inteligentes. La siguiente figura ilustra el despliegue de un contrato inteligente y la invocación de un contrato inteligente.

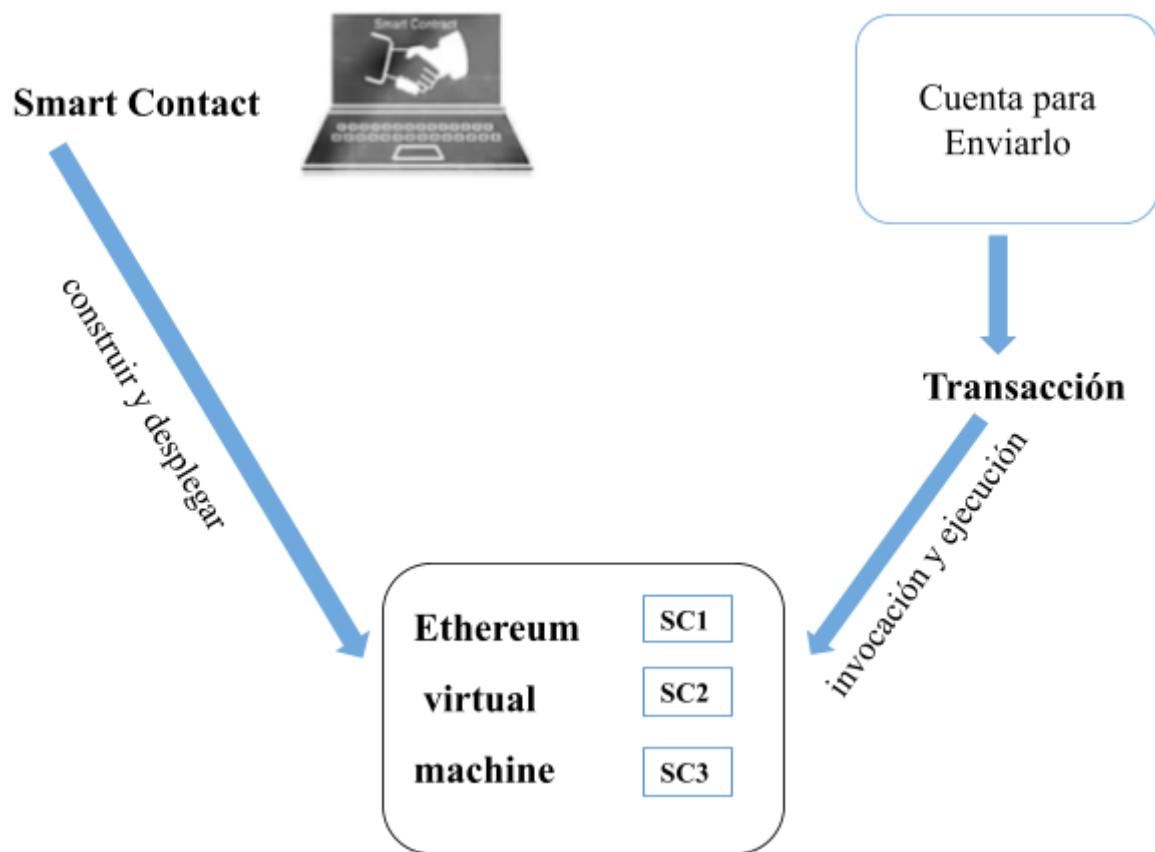


Figura 17. Implementación de una transacción de Ethereum

El contrato inteligente está diseñado, desarrollado, compilado e implementado en el EVM; puede tener más de un contrato inteligente en un EVM. Cuando la dirección de destino en una transacción es un contrato inteligente, el código de ejecución correspondiente al contrato inteligente se activa y está ejecutándose en el EVM. Los datos necesarios para esta ejecución se extraen del campo de carga útil de la transacción. El estado actual del contrato inteligente corresponde a los valores de las variables definidas en él. El estado del contrato inteligente puede ser actualizado por esta ejecución. Una cadena de bloques mantiene tanto el hash de estado como el hash de recepción. Todas las transacciones generadas son validadas. Validar la transacción implica verificar la validez de la combinación de marca de tiempo, nonce y la disponibilidad de tarifas de ejecución suficientes. Los nodos menores de la red reciben, verifican, recopilan y ejecutan transacciones. El código de contrato inteligente en ejecución es ejecutado por todos menores de edad.

5.4. Dapps: aplicaciones descentralizadas

La aparición de Ethereum ha cambiado la forma en que se ve la cadena de bloques. Ethereum es la cadena de bloques hecha para el desarrollo de aplicaciones descentralizadas DApps (Aplicaciones Descentralizadas). Una DApp es un servicio que permite la interacción directa entre usuarios finales y proveedores (por ejemplo, conectar compradores y vendedores en ciertos mercados). Las aplicaciones de Ethereum descentralizadas generalmente interactúan con los usuarios a través de una aplicación web (HTML, Javascript, CSS) usando una API Javascript, Web3 para comunicarse con la cadena de bloques. Luego, el front-end se implementa públicamente y la aplicación es accesible para todos los usuarios de Ethereum. Por otro lado, el back-end (Smart Contracts) se despliega en Blockchain.

Una aplicación descentralizada (DApp) es una aplicación que utiliza contratos inteligentes que proporcionan una interfaz fácil de usar para contratos inteligentes. Un ejemplo típico de DApp es una aplicación de criptomonedas que se ejecuta en una red cadena de bloques. La estructura de una aplicación descentralizada se compone de una interfaz de usuario (navegador web, HTML, CSS) y una interfaz de usuario (Web3 JavaScript). La aplicación DApp interactúa con el nodo Ethereum (EVM) mediante JSON RPC. JSON RPC es un protocolo de llamada de procedimiento remoto liviano y sin estado que utilizan los clientes de Ethereum para interactuar con un nodo de Ethereum.

Una DApp no necesita una autoridad central para operar: por lo tanto, hace posible las interacciones directas entre usuarios entre usuarios, a través de contratos inteligentes.

Muestran el potencial de las aplicaciones en la cadena de bloques de Ethereum. La mayoría de las DApps requieren la instalación de un cliente Ethereum o el uso de MetaMask [7], una billetera liviana de Ether, como Google Chrome u otro navegador.

En resumen, el proyecto de creación de Ethereum está inspirado en Bitcoin. Por lo tanto, Ethereum retoma los conceptos básicos de Bitcoin pero con un modo de operación bastante diferente. Además de cuentas y mensajes, Ethereum integra formalmente contratos inteligentes. En este sentido, los contratos inteligentes agregan una capa de lógica y computación a la infraestructura de confianza respaldada por la cadena de bloques de Ethereum. Los contratos inteligentes permiten la ejecución de códigos y la mejora de la capacidad de transferencia de valor básico de la cadena de bloques de Bitcoin. Gracias a los contratos, Ethereum permite la apertura de una amplia gama de aplicaciones descentralizadas.

Esto hace posible diseñar y crear una aplicación descentralizada de la agencia inmobiliaria Mia. Dicha aplicación permitirá el almacenamiento seguro de la vivienda, la validación o verificación de propiedad así como descentralización del sistema (accesible para todos).

Capítulo 6

6. Caso de USO: Una DApp para la gestión inmobiliaria

Hemos llegado ahora a la parte más importante que constituye el corazón de la obra. En el capítulo anterior vimos algunos elementos técnicos del funcionamiento del Ethereum Blockchain y sus características. De hecho, los contratos han hecho posible implementar cualquier tipo de aplicación descentralizada utilizando la plataforma Ethereum. Así, elegimos Ethereum para ofrecer un sistema autónomo, seguro y robusto para garantizar su autenticidad y confiabilidad.

El uso de una cadena de bloques pública que sirva como almacenamiento y verificación de contratos de inmuebles. La realización de un sistema descentralizado que satisfaga la necesidad de asegurar los contratos y la propiedad, no es más que la implementación de contratos inteligentes que permiten la importación, en Blockchain Ethereum, de los datos del contrato almacenados en la base de datos.

En este capítulo, en primer lugar, mostraremos las especificaciones de las necesidades funcionales, el desglose y la arquitectura del sistema. En segundo lugar, implementaremos el sistema DApp mostrando algo de programación. Finalmente presentaremos las interfaces de la aplicación.



6.1. Análisis y diseño de sistemas

La implementación de nuestra aplicación descentralizada sigue una secuencia lógica de fases y pasos, desde la especificación de necesidades funcionales hasta la implementación o la realización (escritura y prueba de programas). Este diseño revela los contratos que entran en juego y su comportamiento.

6.1.1. Especificación de necesidades funcionales.

Este paso está marcado por el diseño de contratos que describen el funcionamiento del sistema para facilitar la realización de la aplicación.

6.1.1.1. Diagrama de casos de uso

Un actor representa la abstracción de un papel desempeñado por entidades internas.

En nuestra aplicación, distinguimos principalmente dos actores:

- Administrador: Es la persona que gestiona la administración de la aplicación.
 - Cliente: Es la persona que utiliza la aplicación.
- Gestión Inmobiliaria

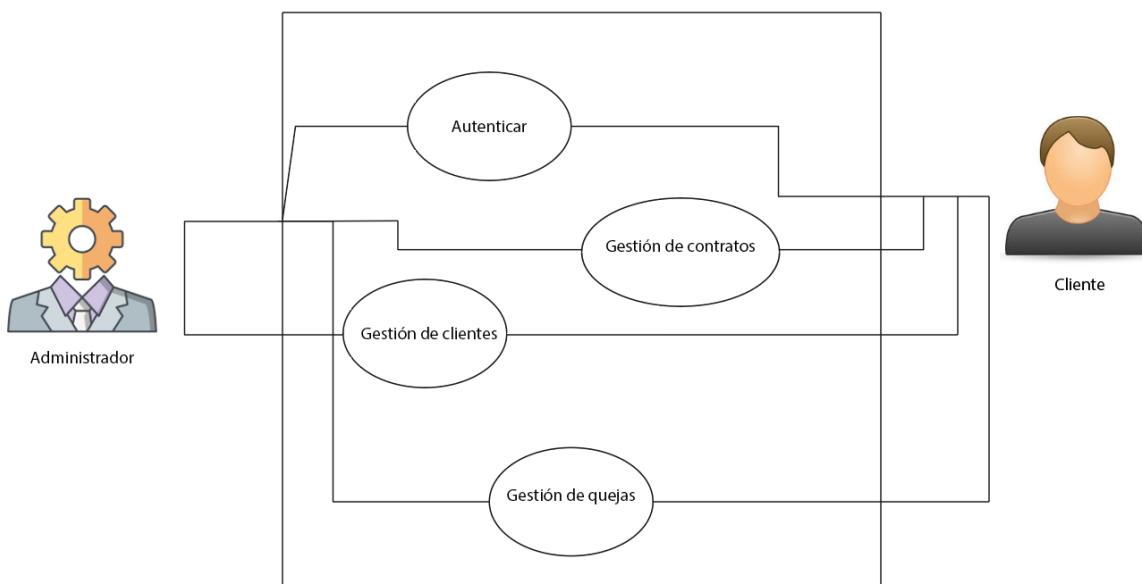
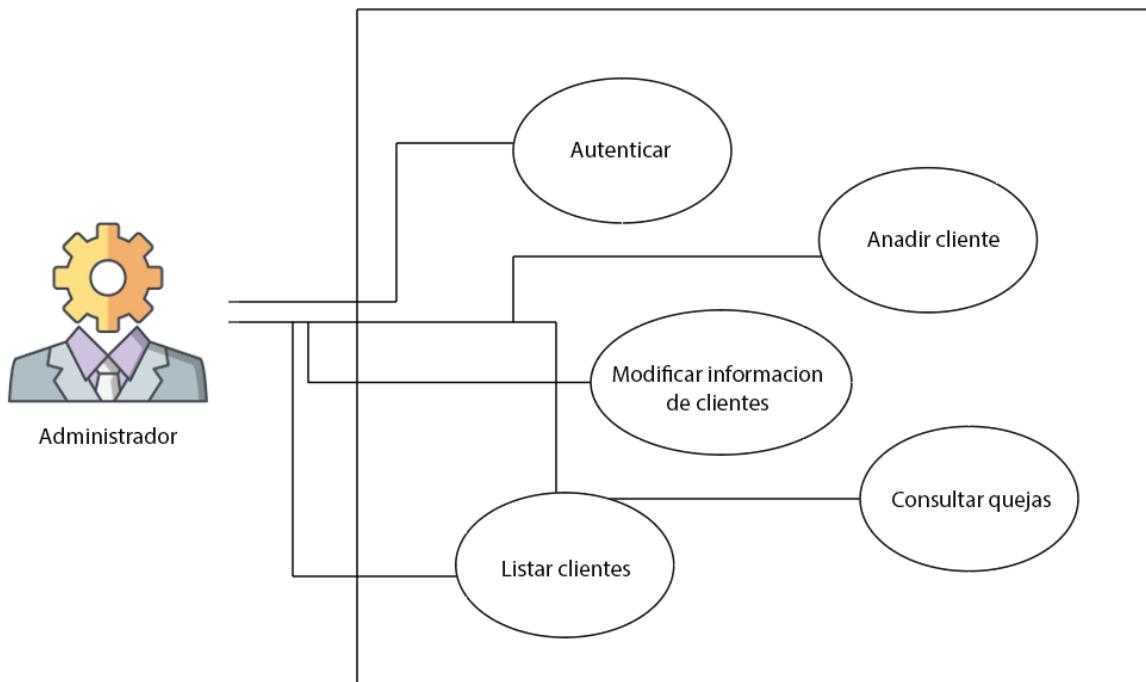
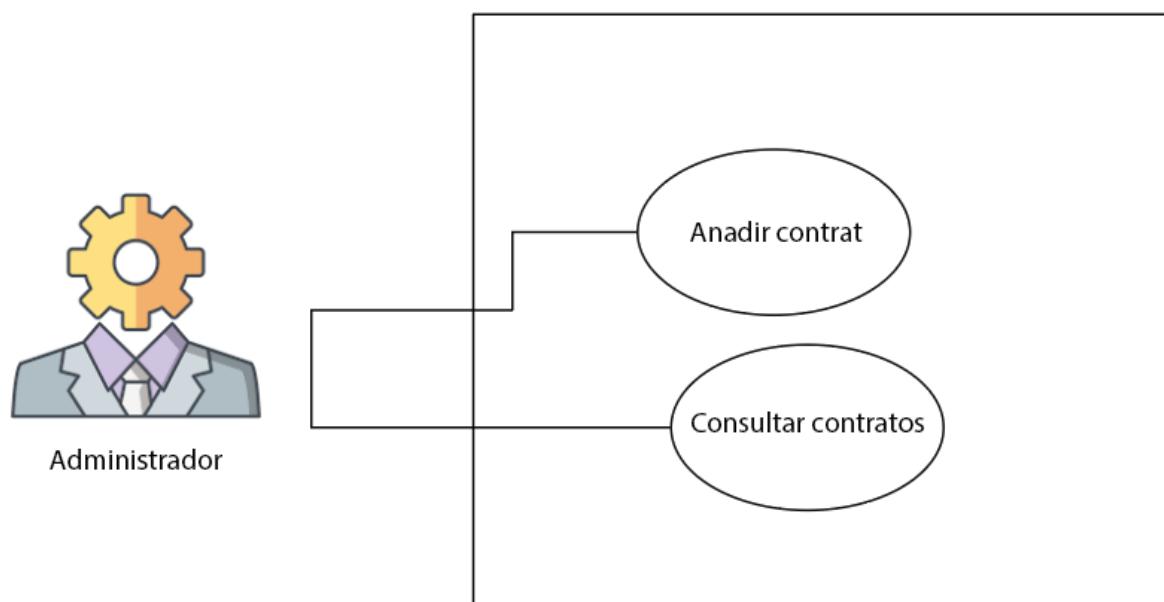


Figura 18. Diagrama de caso de uso general

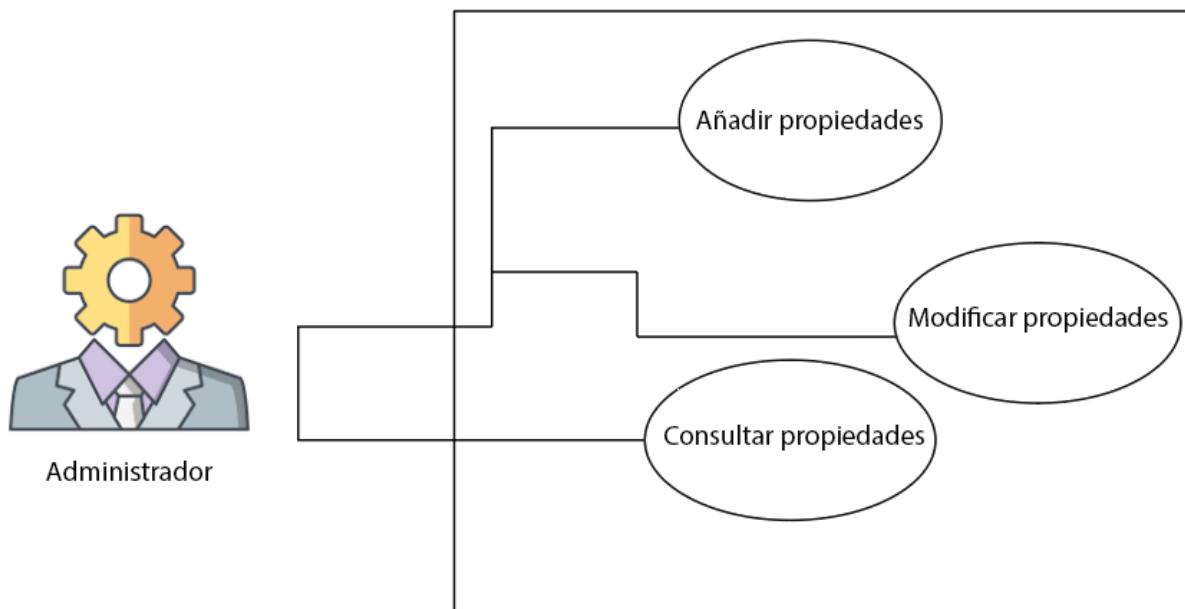
→ Gestión de usuarios

**Figura 19. Gestión de clientes**

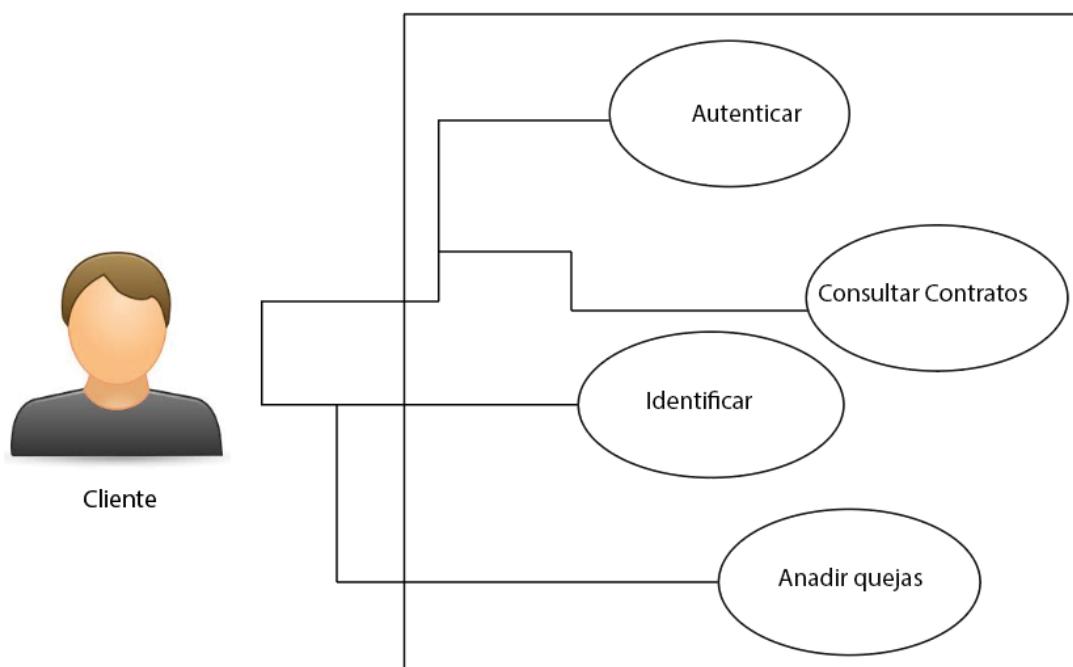
→ Gestión de contratos

**Figura 20. Gestión de contratos**

→ Gestión de propiedades

**Figura 21.** Gestión de propiedades

→ Gestión de cliente

**Figura 22.** Gestión de cliente

6.1.1.2. Diagrama de clase

El diagrama de clases describe claramente la estructura del sistema al modelar sus clases, atributos, operaciones y relaciones entre sus objetos.

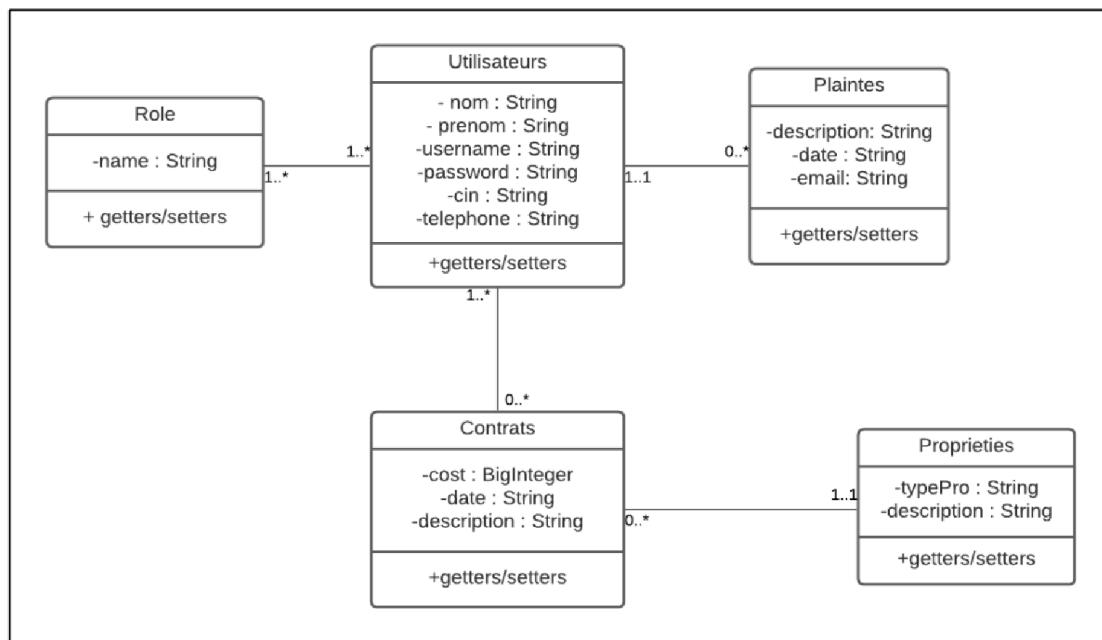


Figura 23. Gestión de clase

6.1.2. Arquitectura del sistema

Las aplicaciones descentralizadas que se ejecutan en Blockchain generalmente están estructuradas como se muestra en la Figura 23 a continuación. Una aplicación descentralizada (DApp) se compone de aplicaciones front-end y back-end. Los usuarios interactúan con la DAPP desde el front-end, que a su vez se comunica con la parte de back-end de la aplicación. Este último entra en contacto con los nodos Ethereum (la máquina virtual) a través de la interfaz RPC.

Esta arquitectura muestra los componentes de software de nuestro sistema descentralizado para asegurar MIA. Pero el funcionamiento de este sistema está íntimamente ligado al comportamiento de nuestro sistema MIA. De hecho, para proporcionar las entradas de las funciones de los contratos, la aplicación descentralizada en su capa comercial debe interrogar la base de datos del sistema centralizado. Entonces decidimos importar los datos para que se hicieran al final de cada año. Tal decisión se toma con el objetivo de optimizar el

número de transacciones para reducir costos. En este sentido, sí existe un compromiso entre seguridad y coste: “no existe la seguridad gratuita”.

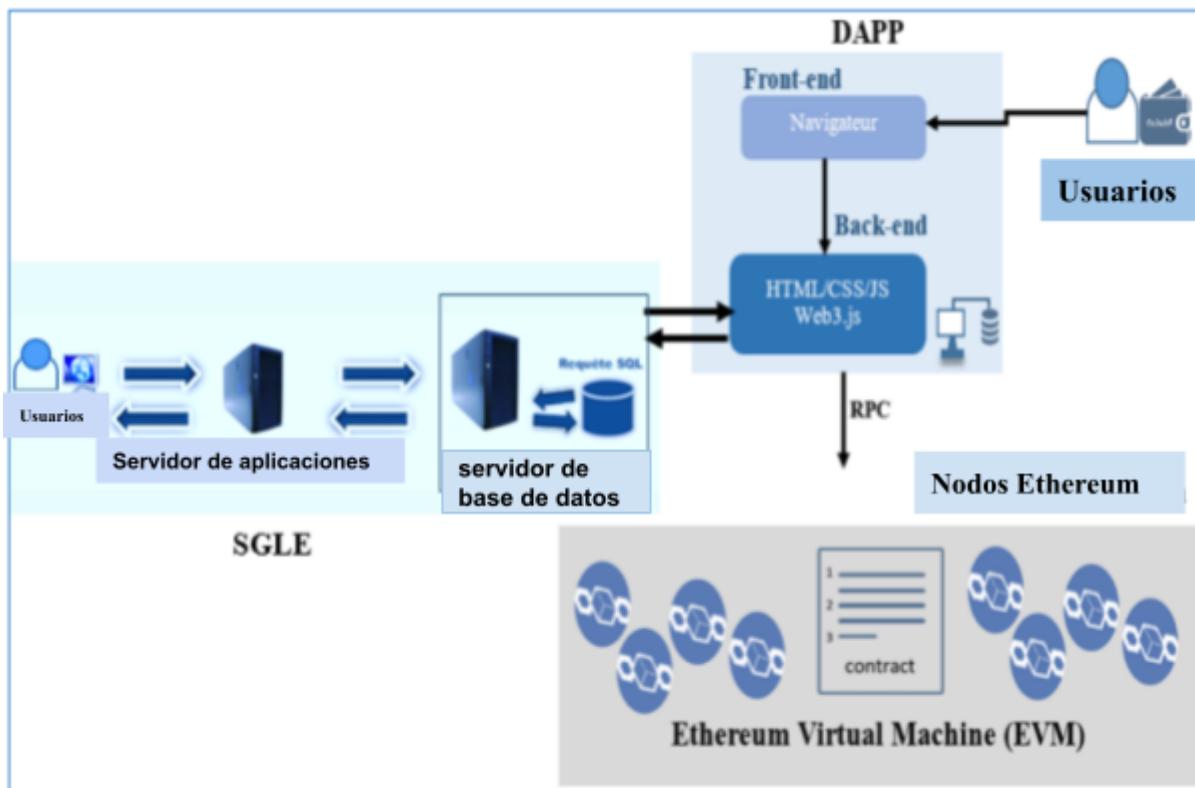


Figura 24. Arquitectura de la aplicación

6.2. Implementación del sistema

Las funciones y procesos que se definen durante la fase de diseño se traducen en código de programa. La implementación del sistema es en este sentido la codificación de las especificaciones técnicas del diseño. Así presentamos en esta parte los códigos clave para el funcionamiento del sistema; los detalles se darán en los anexos. Pero antes de eso, veamos las tecnologías y los lenguajes de programación utilizados para la codificación.

6.2.1. Herramientas de desarrollo y tecnologías utilizadas.

Para completar el desarrollo de nuestro Folleto DApp, se requiere más investigación para comprender la API JavaScript de Web3, la API JSON RPC y el lenguaje de programación Solidity.

De hecho, la API Web3Js, un SDK de JavaScript, se utiliza para interactuar con un nodo Ethereum. La API JSON RPC, por otro lado, sirve como interfaz entre usuarios y nodos; es utilizado por Web3. Esta interfaz permite acceder a los datos almacenados por los Smarts Contracts y reaccionar a los eventos que ocurren en Blockchain. La interfaz JSONRPC, que es una interfaz RPC que utiliza JSON como estructura de datos para modelar los datos que se enviarán a la cadena de bloques. Solidity es el lenguaje de programación para codificar contratos inteligentes.

Existen herramientas de desarrollo que ayudan a desarrollar, probar e implementar aplicaciones de una manera que utiliza automáticamente los recursos de estas API. Entre los IDE (entornos de desarrollo integrado) o Framework que existen hasta el momento podemos mencionar:

→ Remix-IDE:



un reemplazo para Mix-IDE, remix es un entorno para desarrollar contratos inteligentes en Solidity. Integra un compilador, un entorno de ejecución, depuración y pruebas.

→ Truffle Framework



TRUFFLE

Es un entorno de desarrollo integrado, una infraestructura de prueba y una cartera de activos de clase mundial para Blockchains utilizando Ethereum Virtual Machine (EVM), con el objetivo de facilitar la vida de los desarrolladores.

Genera el esqueleto de una aplicación Ethereum y un contrato inteligente, así como un front-end. Truffle [15] permite la implementación de Dapps en el cliente que se ejecuta en la máquina (la cadena de bloques local). Esta cadena de bloques local puede ser un cliente de Ethereum o una lanzada por GANACHE (parte de la suite Truffle).

→ Ganache



Ganache

Ganache te permite tener una blockchain de Ethereum en tu máquina para la ejecución de pruebas, comandos y la inspección de los estados de la blockchain. Además, el marco Truffle proporciona RPC de prueba al ejecutar Dapp localmente. Estos

paquetes de NPM ofrecen herramientas clásicas para desarrolladores: automatización de la compilación de Solidity, integración con herramientas de prueba unitaria, automatización de la implementación y comunicación con la cadena de bloques. Estas muchas características motivaron nuestra elección de este Framework.

→ Metamask



MetaMask es una extensión o complemento para navegadores web que permite a los usuarios interactuar fácilmente con dApps desde el Cadena de bloques de Ethereum.

→ Spring Boot



Spring Boot es un marco de desarrollo JAVA. Es una variación del marco Spring clásico que esencialmente hace posible la creación de microservicios (la mayoría de las veces, los servicios web se agrupan en API).

Para garantizar la comunicación entre la parte frontal y la base de datos de folletos escolares, utilizamos Java a través de Spring Framework.

→ Angular



Angular es un marco JavaScript de código abierto escrito en TypeScript. Google lo mantiene y proporciona una estructura estándar para que trabajen los desarrolladores. Angular usa la sintaxis HTML para definir claramente los componentes del programa. Permite a los desarrolladores crear aplicaciones grandes que sean fáciles de mantener.

→ MongoDB



MongoDB es una base de datos NoSQL multiplataforma y abierta fuente, utilizada por muchas aplicaciones web modernas basadas en nodos para conservar los datos.

6.2.2. Requisitos

La figura 24 muestra la estructura de nuestro proyecto:

- la carpeta de compilación contiene los archivos Json de los contratos que se generan durante la implementación (con el comando "Migrar trufa").
- La carpeta de contratos como su nombre indica contiene los contratos inteligentes.
- Src es la carpeta principal del proyecto; contiene clases, servicios, componentes divididos en módulos. Las clases sirven como facilitadores de la manipulación de los datos recibidos de la base de datos o de la cadena de bloques por los servicios.

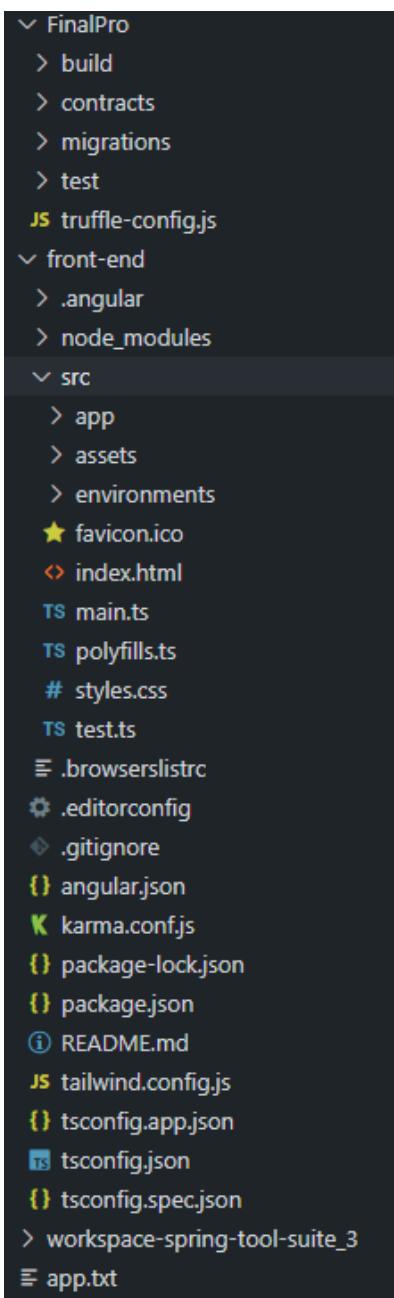


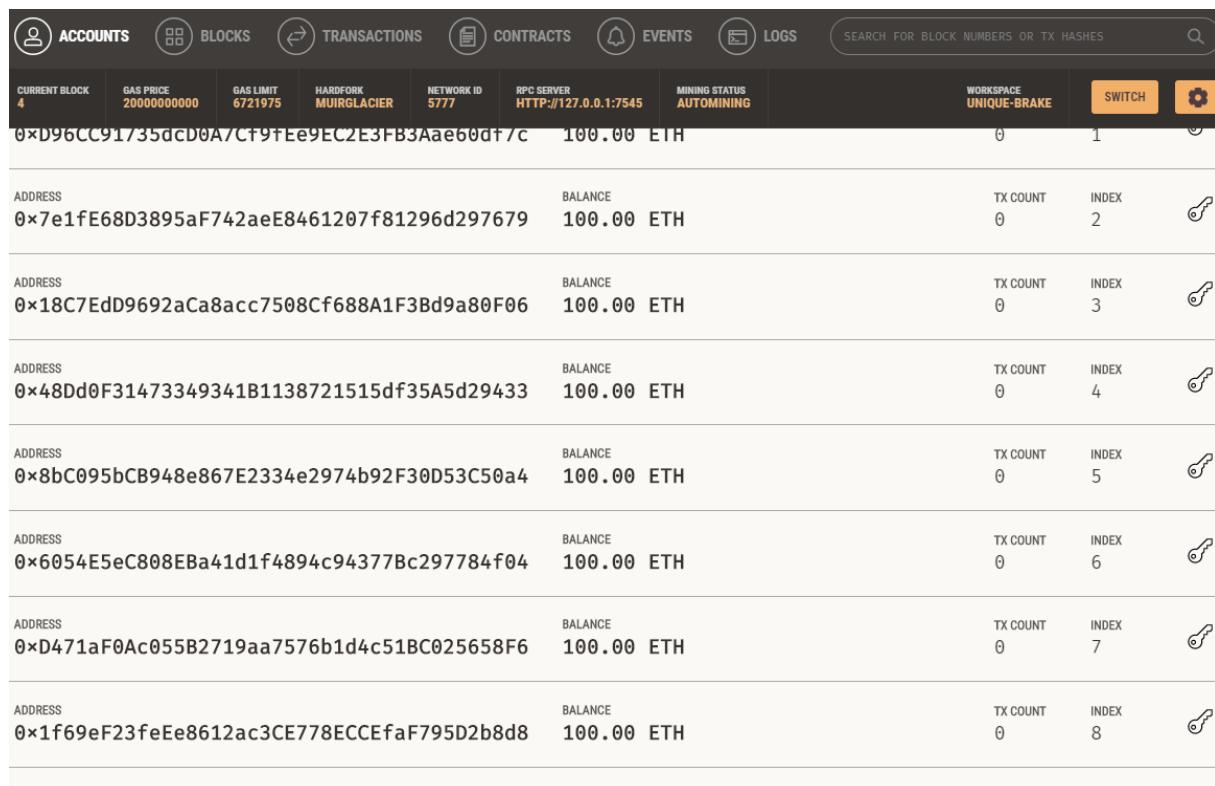
Figura 25. Estructura del proyecto

Además, el componente del paquete Truffle Suite, Ganache, nos permite ejecutar nuestra cadena de bloques Ethereum personal. Genera las cuentas con un monto inicial de 100 ETH. El archivo truffle-config.js proporciona el enlace entre ganache y nuestra aplicación. La Figura 26 muestra algunas interfaces de cómo funciona el ganache.

6.2.3. Codificación de servicios

- **Instanciando Web3**

El siguiente código crea una instancia Web3 y define un proveedor. Un navegador compatible con Ethereum (en nuestro caso estamos usando la extensión MetaMask) tendrá disponible web3.currentProvider.



The screenshot shows the Ganache interface with the following details:

CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE	SWITCH
4	2000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:7545	AUTOMINING	UNIQUE-BRAKE	<input checked="" type="button"/>
0xD96CC91735dCD0A7Cf9fEe9EC2E3FB3Aae60dt/c							100.00 ETH	<input type="button"/> 0 1 <input type="button"/>
ADDRESS 0x7e1fE68D3895aF742aeE8461207f81296d297679							BALANCE 100.00 ETH	TX COUNT 0 INDEX 2 <input type="button"/>
ADDRESS 0x18C7EdD9692aCa8acc7508Cf688A1F3Bd9a80F06							BALANCE 100.00 ETH	TX COUNT 0 INDEX 3 <input type="button"/>
ADDRESS 0x48Dd0F31473349341B1138721515df35A5d29433							BALANCE 100.00 ETH	TX COUNT 0 INDEX 4 <input type="button"/>
ADDRESS 0x8bC095bCB948e867E2334e2974b92F30D53C50a4							BALANCE 100.00 ETH	TX COUNT 0 INDEX 5 <input type="button"/>
ADDRESS 0x6054E5eC808EBa41d1f4894c94377Bc297784f04							BALANCE 100.00 ETH	TX COUNT 0 INDEX 6 <input type="button"/>
ADDRESS 0xD471aF0Ac055B2719aa7576b1d4c51BC025658F6							BALANCE 100.00 ETH	TX COUNT 0 INDEX 7 <input type="button"/>
ADDRESS 0x1f69eF23feEe8612ac3CE778ECCEfaF795D2b8d8							BALANCE 100.00 ETH	TX COUNT 0 INDEX 8 <input type="button"/>

Figura 26. Ganache

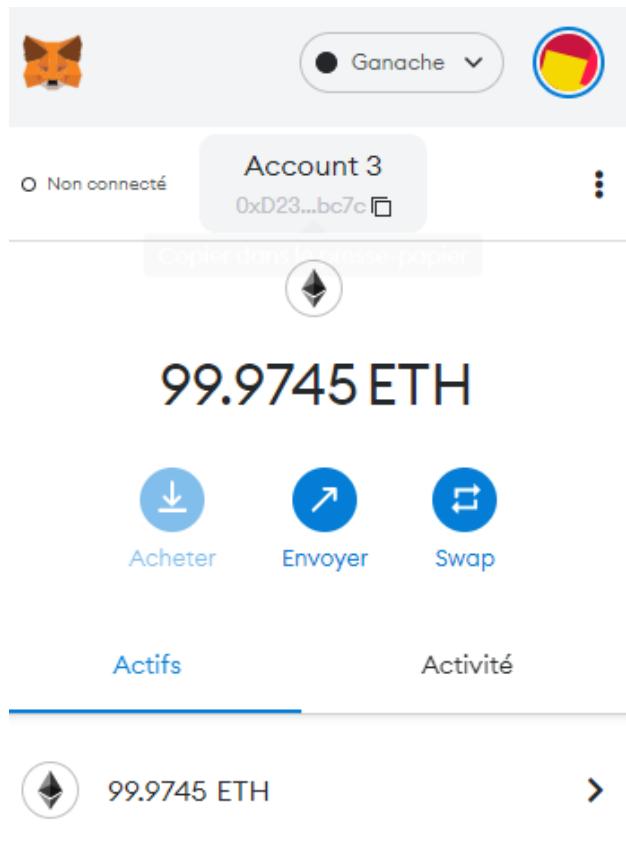


Figura 27. Metamask

6.2.4. Arquitectura de microservicios

El microservicio es un método de desarrollo de aplicaciones de software como un conjunto de servicios modulares e independientes, en el que cada servicio ejecuta un proceso que comunica a través de un mecanismo previamente definido.

En nuestro proyecto hemos adaptado esta arquitectura, la figura 28 muestra la estructura de nuestros micro servicios de Java.

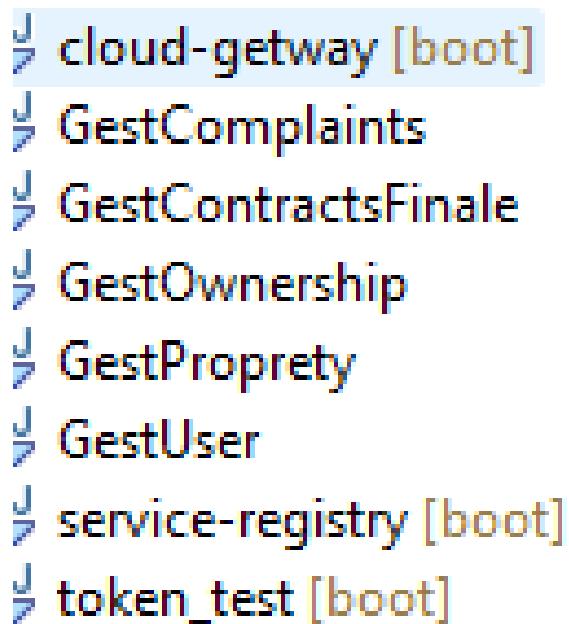


Figura 28. Estructura de los microservicios Java

6.3. Descripción general del DAPP

6.3.1. Interfaces del DAPP

En esta etapa presentamos nuestra aplicación web a través de varias serigrafías realizadas.

- **Inicio**

Permite regresar a la página de inicio que explica brevemente la plataforma;



Figura 29. Página de inicio

El usuario puede activar o desactivar el tema oscuro, utilizando el botón en la parte superior izquierda de la página de inicio:



Figura 30. Activación del modo oscuro

- Quien somos

Permite navegar a la página "Quien somos" que, como su nombre indica, muestra el contexto y los objetivos a los que apunta la aplicación.



Figura 31. Página quien somos

- Servicio

Que nos lleva a la lista de servicios que ofrece la aplicación.



Figura 32. Página servicios

- **Contáctenos**

Este menú nos lleva a la información de contacto que es útil en caso de que se necesite soporte técnico.

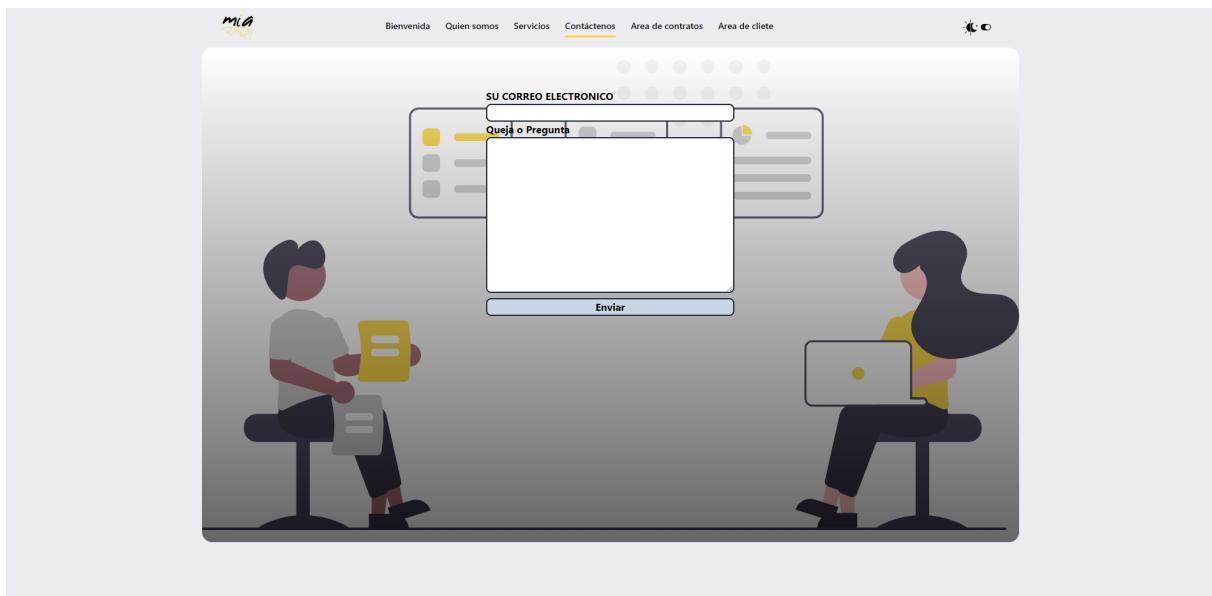


Figura 33. Página contáctenos

- **El menú “Área de contratos”:**

Presenta la página donde hay una lista de contratos registrados por MIA. La lista contiene solo tres campos: la fecha del contrato, su número y el identificador del producto (casa, piso, etc.). Pero si quieres más detalles puedes hacer clic en el ícono del ojo.

Fecha	Número de Contrato	Identificación de producto	Mas detalles
02/01/2022	61d738b03de8f33ad9a87755	61d71047ef830d1ff901e25	👁
02/01/2022	61d8392a3de8f33ad9a87759	61d71047ef830d1ff901e25	👁
02/01/2022	61d8479c3de8f33ad9a8775a	61d94745ef830d1ff901e26	👁
01/01/2022	61d84883de8f33ad9a8775b	61d84862ef830d1ff901e27	👁
01/01/2022	61dab774d908dd3695210884	61d94862ef830d1ff901e27	👁

Figura 34. Página área de contrato

Los datos del contrato son el coste de los productos vendidos, el tipo de producto, la descripción (el lugar, la superficie, etc.) y el identificador de las partes contratantes.



Figura 35. Detalles de un contrato

El usuario puede buscar un contrato por identificador de producto



Figura 36. Buscar un contrato

El usuario puede ordenar la visualización de los contratos por fecha o precio y de forma ascendente o descendente.



Figura 37. Ordenar contratos

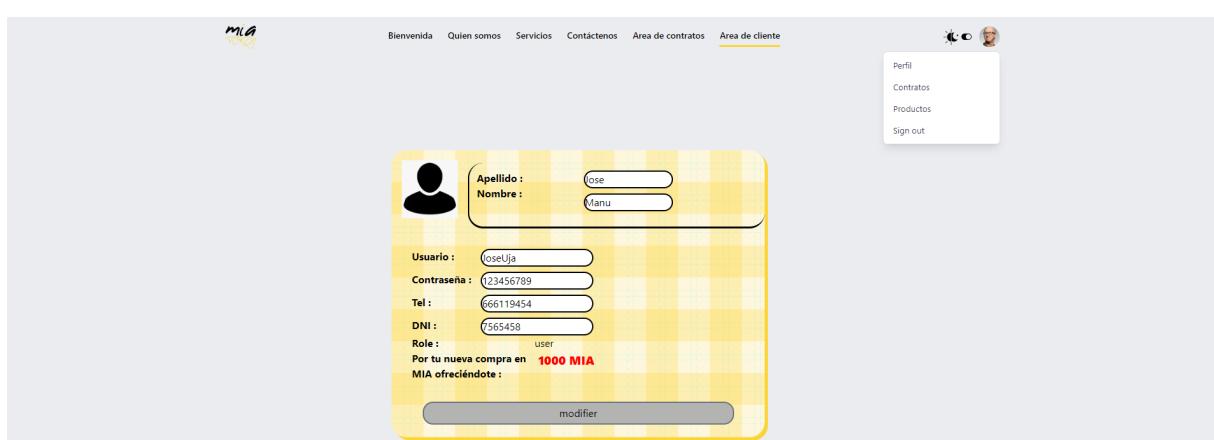


Figura 38. Dashboard cliente

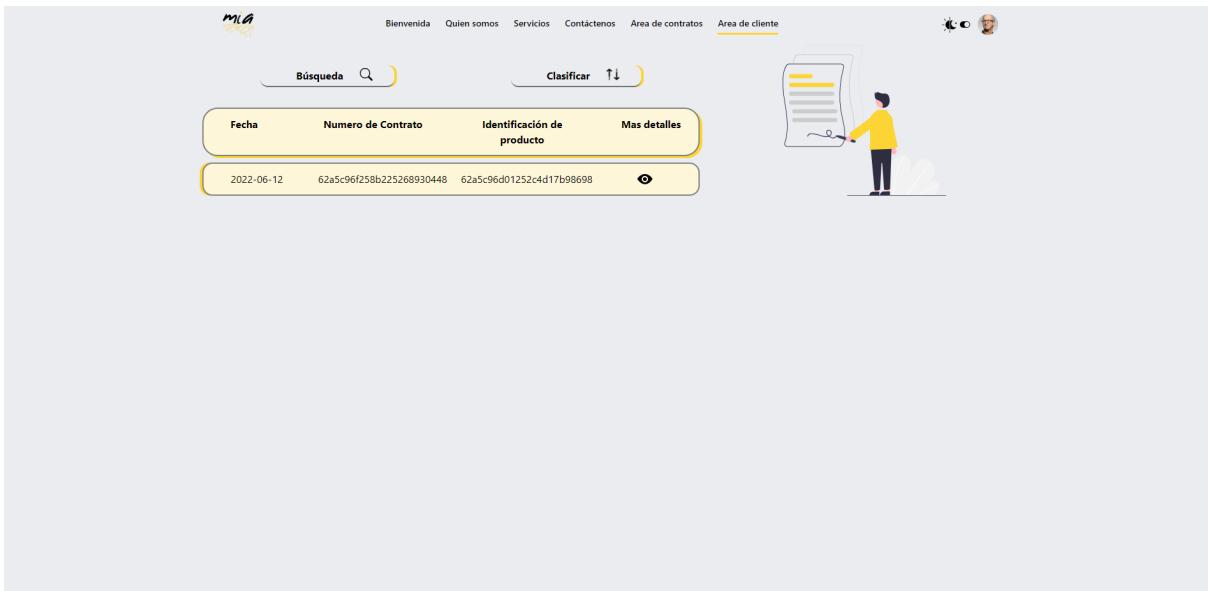


Figura 39. Espacio contratos del cliente

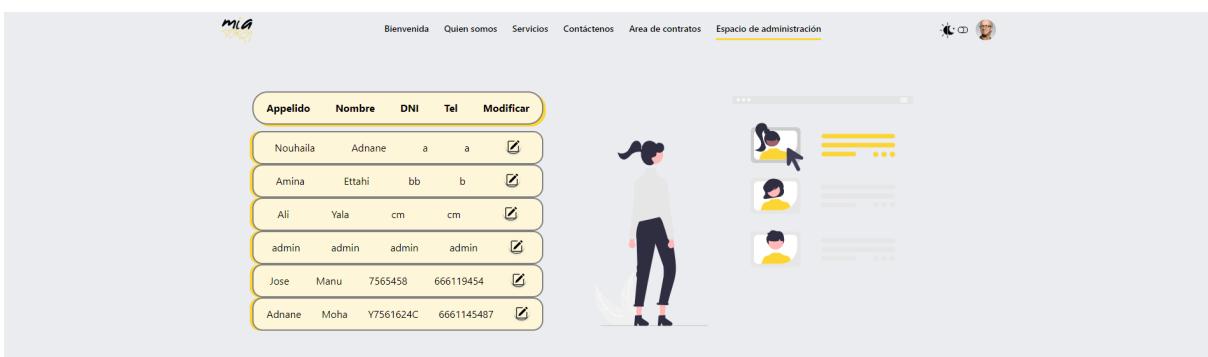


Figura 40. Espacio contratos de clientes

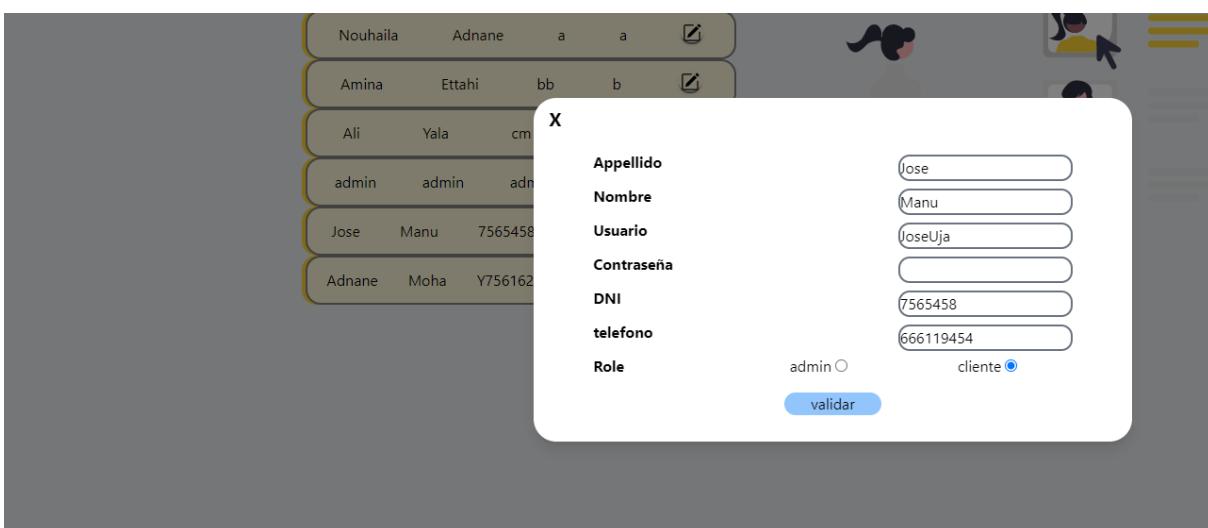


Figura 41. Editar un perfil

- **Tutoriales**

Para facilitar y que el administrador entienda bien el funcionamiento de la plataforma, he desarrollado un tutorial que muestra a cada nuevo administrador como puede usar la plataforma. Puede seguir como puede salir de la tutorial si antes ha usado la plataforma. El nuevo administrador se detecta de su navegador si es un navegador nuevo sale si no no sale.

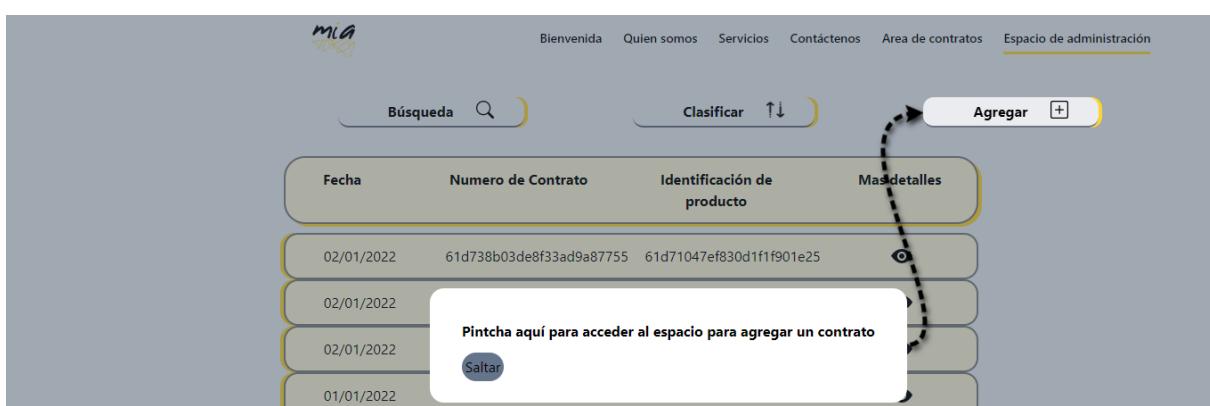


Figura 42. Tutorial como agregar un contrato



Figura 43. tutorial de venta

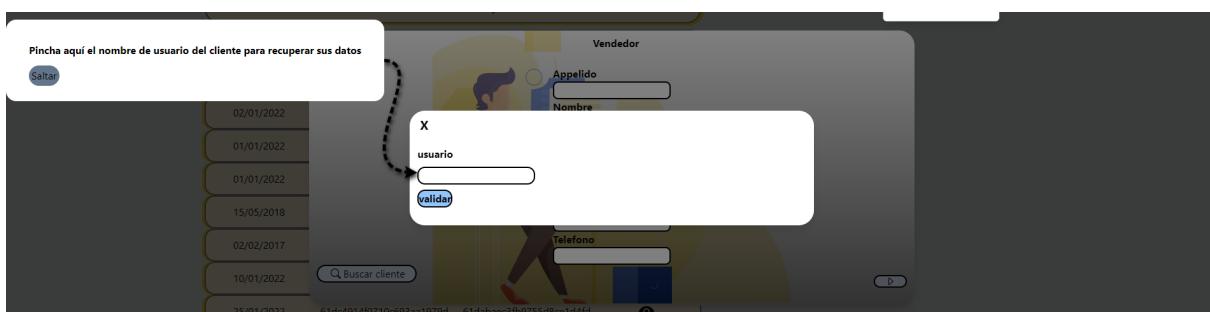


Figura 44.tutorial de búsqueda

6.4. Token de la DApp desarrollada

El token Mia [8] está escrito con el idioma de programación Solidity [12].

En la primera parte se declara el tipo de nuestro token y en este caso es el estándar ERC20, donde tenemos que especificar el **totalSupply()** es el max supply, **balanceOf** el balance que tiene el contrato es decir el número de tokens que contiene, **allowance** está function para dar el aprobacion y los permisos necesarios para intercambiar este token dentro de las plataformas como Uniswap [16], **transfer** para transferir el token, **approve** esta función trabaja con allowance su utilidad y dar permiso, **transferFrom** es para que podamos transferir tokens desde mi dirección a otra o el contrario.

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity >=0.4.22 <0.9.0;
3
4 interface IERC20 {
5
6     function totalSupply() external view returns (uint256);
7     function balanceOf(address account) external view returns (uint256);
8     function allowance(address owner, address spender) external view returns (uint256);
9
10    function transfer(address recipient, uint256 amount) external returns (bool);
11    function approve(address spender, uint256 amount) external returns (bool);
12    function transferFrom(address sender, address recipient, uint256 amount) external returns (bool);
13
14
15    event Transfer(address indexed from, address indexed to, uint256 value);
16    event Approval(address indexed owner, address indexed spender, uint256 value);
17 }
18

```

Figura 45. funciones del contrato

La segunda parte son parámetros de nuestro token, empieza con el nombre del contrato este nombre no aparecerá en la blockchain simplemente es para manejar el token.

Los parámetros son los siguientes: la variable **name** es el nombre de nuestro token, **symbol** el símbolo del token y **decimals** los decimales que va a tener el token y el muy importante es el **totalSupply** en nuestro caso he puesto “ veintiún millones “ eso lo multiplico por 10 y elevado los decimales que son 18.

```
19  contract MiaToken is IERC20 {
20      using SafeMath for uint256;
21
22      string public constant name = "MiaToken";
23      string public constant symbol = "Mia";
24      uint8 public constant decimals = 18;
25
26      mapping(address => uint256) balances;
27      mapping(address => mapping (address => uint256)) allowed;
28
29      uint256 private totalSupply_ = 21000000*10**uint256(decimals);
30
31  constructor() public {
32      balances[msg.sender] = totalSupply_;
33  }
34
35  function totalSupply() public override view returns (uint256) {
36      return totalSupply_;
37  }
38
39  function balanceOf(address tokenOwner) public override view returns (uint256) {
40      return balances[tokenOwner];
41  }
42
43
44  function transfer(address receiver, uint256 numTokens) public override returns (bool) {
45
46      require(numTokens <= balances[msg.sender]);
47      balances[msg.sender] = balances[msg.sender].sub(numTokens);
48      balances[receiver] = balances[receiver].add(numTokens);
49      emit Transfer(msg.sender, receiver, numTokens);
50      return true;
51  }
52
53
54  function approve(address delegate, uint256 numTokens) public override returns (bool) {
55      allowed[msg.sender][delegate] = numTokens;
56      emit Approval(msg.sender, delegate, numTokens);
57      return true;
58  }
59
60  function allowance(address owner, address delegate) public override view returns (uint) {
61      return allowed[owner][delegate];
```

Figura 46. Parámetros del contrato

```
constructor() public{
    owner = msg.sender;
}

event Add(
    string numeroContract,
    string idProprety,
    string ownerLand,
    string buyerLand,
    uint cost,
    string desc,
    string date
);

modifier isOwner {
    require(msg.sender == owner);
    _;
}

function addLand( string memory numeroContract,
                  string memory idProprety,
                  string memory _ownerLand,
                  string memory _buyerLand,uint _cost,
                  string memory _desc,
                  string memory _date)
public isOwner {

    require(_cost > 0, 'Must be a cost');

    PropretyCount++;

    Propreties[PropretyCount] = Contract( numeroContract,
                                         idProprety,_ownerLand,
                                         _buyerLand, _cost,
                                         _desc,_date);

    emit Add( numeroContract,
              idProprety,_ownerLand,
              _buyerLand, _cost,
              _desc,_date);
}

function getNumberLands() public view returns(uint){
    return PropretyCount;
}
```

Figura 47. constructor public

En esta parte muestro las etapas seguidas para deploy mi contrato en el Ropsten Network [10] como se ve en la figura 48 el gas fee para deploy un contrato es 0.002199 ETH.

Para pagar este fiz hay muchas paginas para solicitar algún eth del test la mejor una que he encontrado es teth bitaps [13] manda 1 Eth por cada minuto.

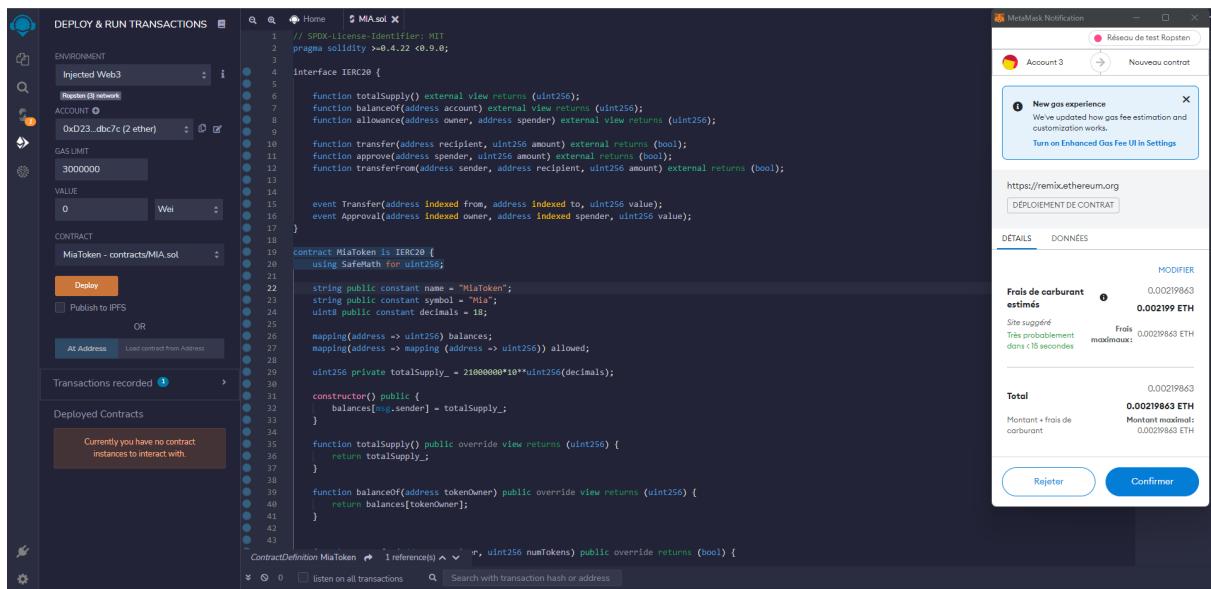


Figura 48. Deploy Mia contract

Después de un ratio sale este mensaje (figura 49) que dice que todo ha sido bien ejecutado y que nuestro contrato está bien en el testnetwork de Ropsten.

Luego cuando entremos a buscar nuestro contrato en Ropsten (figura 50) nos sale nuestro contrato.

```
Welcome to Remix 0.24.1
Your files are stored in indexedDB, 3.37 MB / 534.45 GB used

You can use this terminal to:
• Check transactions details and start debugging.
• Execute JavaScript scripts:
  - Input a script directly in the command line interface
  - Select a Javascript file in the file explorer and then run `remix.execute()` or `remix.executeCurrent()`
  - Right click on a JavaScript file in the file explorer and then click 'Run'

The following libraries are accessible:
• web3 version 1.5.2
• ethers.js
• remix

Type the library name to see available commands.
creation of MiaToken pending...

view on etherscan

[✓] [block:12418861 txIndex:6] from: 0xD23...dbc7c to: MiaToken.(constructor) value: 0 wei data: 0x608...f0033 logs: 0 hash: 0x44d...61d77
```

Figura 49. Log

The screenshot shows the Etherscan interface for the MiaToken contract. At the top, there's a navigation bar with 'All Filters', a search bar ('Search by Address / Txn Hash / Block / Token / Ens'), and a magnifying glass icon. Below the navigation is a dropdown menu with 'Home', 'Blockchain', 'Tokens', 'Misc', and 'Ropsten'. The main content area has a header 'Token MiaToken' with a circular icon. It's divided into two sections: 'Overview [ERC-20]' on the left and 'Profile Summary' on the right. In the Overview section, it shows 'Max Total Supply: 21,000,000 Mia', 'Holders: 0', and 'Transfers: 0'. In the Profile Summary section, it shows 'Contract: 0xe509125a7946787b6bd60134dabf9bc86737c3e1', 'Decimals: 18', and a QR code.

Figura 50. Mia Token Etherscan [48]

En el etherscan hay una posibilidad de verificar nuestro contrato si está segura y si no contiene ningún error y como se ve en la figura 51 nuestro contrato está bien verificado por la plataforma.

The screenshot shows the Etherscan interface for the MiaToken contract under the 'Contract' tab. At the top, there's a 'Transactions' button, a 'Contract' button (which is active and highlighted in blue), and an 'Events' button. Below the tabs, there are buttons for 'Code', 'Read Contract', and 'Write Contract'. To the right is a search bar labeled 'Search Source Code' with a magnifying glass icon. The main content area shows a green checkmark next to 'Contract Source Code Verified (Exact Match)'. It displays the 'Contract Name: MiaToken', 'Compiler Version: v0.8.15+commit.e14f2714', 'Optimization Enabled: No with 200 runs', and 'Other Settings: default evmVersion, MIT license'. At the bottom right, there are buttons for 'Outline', 'More Options', and some icons. A small note at the bottom left says 'Contract Source Code (Solidity)'.

Figura 51. Mia Token verified contract

Capítulo 7

7. Conclusión y trabajo futuro

7.1. Conclusión

Este trabajo de fin de máster consistió en la realización de una aplicación descentralizada, DApp, soportada por la tecnología de cadena de bloques, Blockchain, y una arquitectura de microservicios, junto a una aplicación SPA/WPA que consume estos microservicios. En el sector inmobiliario, la venta de viviendas y su posterior registro seguro y confiable resulta atractivo, para lo cual la tecnología de Blockchain ofrece unas inmensas posibilidades.

A comienzo de esta memoria se propuso una solución informática para superar los problemas que aquejan la gestión segura de los contratos de compra-venta, para lo cual se ha diseñado y desarrollado una DApp para fortalecer la seguridad y facilitar al vendedor y al comprador una compra segura.

El trabajo de investigación realizado tuvo por objetivo comprender mejor el campo y el funcionamiento de los establecimientos. Esto permitió escribir el primer capítulo de este informe de memoria. Además, es necesario un perfecto conocimiento de la tecnología blockchain para conseguir los objetivos que nos planteamos al inicio de esta memoria, para lo cual también se dedicó un tiempo importante a inicios del TFM. Por otro lado, los capítulos segundo, tercero, cuarto y quinto se centraron en describir y ampliar los conocimientos acerca de esta tecnología novedosa.

En el sexto capítulo se mostraron los desarrollos técnicos llevados a cabo para la realización de la DApp, así como la propuesta de un token concreto para la misma, el token MIA. Durante esta etapa de desarrollo, se alcanzaron los objetivos planteados para dicha aplicación centrada en la gestión descentralizada, confiable y segura de venta de bienes inmuebles (e.g. casas, pisos, terrenos, etc), junto a:

- Reducción de costes y tiempos de transacción.

- Apertura de la inversión inmobiliaria a un público amplio, ya sea comprando o alquilando.
- Facilitación de la gestión de carteras inmobiliarias a particulares y profesionales.
- Digitalización completa del sector, en conexión con la inteligencia artificial y el Internet de las Cosas.

7.2. Trabajos futuros

Creación de un Marketplace donde los clientes puedan vender y comprar su propiedad puede facilitar la tarea y luchar contra el fraude. Permitir a los clientes vender el token MIA y enviarlo entre ellos usando la billetera en línea “Metamask” o extender la funcionalidad del sistema actual para que los clientes puedan intercambiar el token con un piso o tierra.

Por último, resulta de interés el desarrollo de una API para compartir los pisos vendidos para que, por ejemplo, otras páginas o empresas puedan verificar y garantizar la propiedad de una persona o de un grupo, y así luchar contra el fraude.

Referencias

- [1]. « Bitcoin - Argent P2P libre et ouvert ». [En línea]. Disponible en: <https://bitcoin.org/fr/> [Consultado el: 09-05-2022].
- [2]. « bitcoin-blockchain-infographic-fr.jpg (1457×1030) ». [En línea]. Disponible en: <https://bitconseil.fr/wp-content/uploads/2016/03/bitcoin-blockchain-infographic-fr.jpg> [Consultado el: 22-05-2022].
- [3]. « Dash Site Officiel | Dash Crypto Currency - Dash ». [En línea]. Disponible en: <https://www.dash.org/> [Consultado el: 06-05-2022].
- [4]. « Ethereum Project ». [En línea]. Disponible en: <https://www.ethereum.org/> [Consultado el: 02-04-2022].
- [5]. « Hyperledger – Open Source Blockchain Technologies ». [En línea]. Disponible en: <https://www.hyperledger.org/> [Consultado el: 11-05-2022].
- [6]. « Litecoin - Monnaie numérique P2P Open Source ». [En línea]. Disponible en: <https://litecoin.org/> . [Consultado el: 14-04-2022].
- [7]. « MetaMask ». [En línea]. Disponible en: <https://metamask.io/> [Consultado el: 18-04-2022].
- [8]. « Mia token contract ». [En línea] [Consultado el: 18-06-2022] <https://ropsten.etherscan.io/token/0xe509125a7946787b6bd60134dabf8bc86737c3e1> .
- [9]. « Namecoin ». [En línea]. Disponible en: <https://namecoin.org/> . [Consultado el: 15-04-2022].

[10]. « Ropsten Testnet Network » [En línea] [Consultado el: 18-06-2022] disponible en : <https://ropsten.etherscan.io/> .

[11]. « Slock.it - Landing ». [En línea]. Disponible en: <https://slock.it/> [Consultado el: 10-04-2022].

[12]. « Solidity — Solidity 0.5.4 documentation ». [En línea]. Disponible en: <https://solidity.readthedocs.io/en/latest/> [Consultado el: 02-04-2022].

[13]. « Teth bitaps » [En línea] [Consultado el: 17-06-2022] [En línea]. <https://teth.bitaps.com/> .

[14]. «The Elliptic Curve Digital Signature Algorithm (ECDSA) | SpringerLink ». [En línea]. Disponible en: <https://link.springer.com/article/10.1007/s102070100002> [Consultado el: 04-04-2022].

[15]. « Truffle Suite | Documentación ». [En línea]. Disponible en: <https://truffleframework.com/docs>. [Consultado el: 06-05-2022].

[16]. « Uniswap » [En línea] [Consultado el: 17-06-2022]. <https://uniswap.org/> .

[17]. «Mastering Bitcoin: Programming the Open Blockchain» de Andreas M. Antonopoulos E- 2017.

[18]. «The Basics of Bitcoins and Blockchains: An Introduction to Cryptocurrencies and the Technology That Powers Them» by Antony Lewis E-2018.

[19]. A. Kiayias, E. Koutsoupias, M. Kyropoulou, et Y. Tselekounis, « Blockchain Mining Games », in Proceedings of the 2016 ACM Conference on Economics and Computation, New York, NY, USA, 2016, p. 365–382.

[20]. A.Narayanan et J. Clark, « Bitcoin's Academic Pedigree The concept of cryptocurrencies is built from forgotten ideas in research literature », p. 30.

[21]. An empirical study of Namecoin and lessons for decentralized namespace design. .

[22]. Beyond the hype: Blockchains in capital markets

<https://www.mckinsey.com/industries/financial-services/our-insights/beyond-the-hype-blockchains-in-capital-markets#>.

[23]. Bitcoin Core integration/staging tree. Contribute to bitcoin/bitcoin development by creating an account on GitHub. Bitcoin, 2019.

[24]. Blockchain France, La Blockchain décryptée: les clefs d'une révolution. Paris: Observatoire Netexplo, 2016.

[25]. D. G. Wood, « ETHEREUM: A SECURE DECENTRALIZED GENERALIZED TRANSACTION LEDGER », p. 32.

[26]. E. Androulaki y al., « Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains », in Proceedings of the Thirteenth EuroSys Conference, New York, NY, USA, 2018, p. 30:1–30:15.

[27]. Ethereum Foundation. [En línea]. [Consultado el: 05-04-2022] Disponible en: <https://docs.ethhub.io/ethereum-basics/ethereumfoundation/> .

[28]. F. Pianese, M. Signorini, et S. Sarkar, « Small Transactions with Sustainable Incentives », in 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 2018, p. 1-5.

[29]. F.Tschorsch et B. Scheuermann, « Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies », IEEE Commun. Surv. Tutor., vol. 18, no 3, p. 2084-2123, thirdquarter 2016.

[30]. G. Florin et S. Natkin, « LES TECHNIQUES DE CRYPTOGRAPHIE », p. 79, 2001.

[31]. G. MARIN-DAGANNAUD, « Comprendre la blockchain Ethereum – Article 1 : Bitcoin, première implémentation de la blockchain (1/2) », Ethereum France, 03-juin 2016. [En línea]. Disponible en: <https://www.ethereum-france.com/tout-savoir-sur-ethereum/> [Consultado el: 25-05-2022].

[32]. J. DAVIS, « The crypto-currency: Bitcoin and its mysterious inventor. », vol. 10, p. 6, 2011.

[33]. M. Belotti, N. Bozic, G. Pujolle, et S. Secci, « A Vademeum on Blockchain Technologies: When, Which and How ». 04-oct-2018.

[34]. M. Carlsten, H. Kalodner, S. M. Weinberg, et A. Narayanan, « On the Instability of Bitcoin Without the Block Reward », in Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, New York, NY, USA, 2016, p. 154–167.

[35]. M.E. Gladden, « Cryptocurrency with a Conscience: Using Artificial Intelligence to Develop Money that Advances Human Ethical Values », déc. 2015.

[36]. M. Laurent, « La blockchain est-elle une technologie de confiance », in Signes de confiance : l'impact des labels sur la gestion des données personnelles, Institut Mines Télécom, 2018, p. 179-198.

[37]. M. Swan, Blockchain: Blueprint for a New Economy. O'Reilly Media, Inc., 2015.

[38]. N. Atzei, M. Bartoletti, et T. Cimoli, « A Survey of Attacks on Ethereum Smart Contracts (SoK) », in Principles of Security and Trust, vol. 10204, M. Maffei et M. Ryan, Éd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, p. 164-186.

[39]. N. Szabo, « Formalizing and Securing Relationships on Public Networks », First Monday, vol. 2, no 9, sept. 1997

[40]. Ö. Gürcan, A. Del Pozzo, et S. Tucci-Piergiovanni, « On the Bitcoin Limitations to Deliver Fairness to Users », in On the Move to Meaningful Internet Systems. OTM 2017 Conferences, 2017, p. 589-606.

[41]. R. Matzutt et al., A Quantitative Analysis of the Impact of Arbitrary Blockchain Content on Bitcoin,. [En línea]. [Consultado el: 06-05-2022] Disponible en: <https://fc18.ifca.ai/preproceedings/6.pdf>.

[42]. R. Subramanian et T. Chino, « The State of Cryptocurrencies, Their Issues and Policy Interactions », J. Int. Technol. Inf. Manag., vol. 24, no 3, janv. 2015.

[43]. S. Nakamoto, « Bitcoin: A Peer-to-Peer Electronic Cash System », p. 9.

[44]. T. Gibbs et S. Yordchim, « Thai Perception on Litecoin Value », Int. J. Econ. Manag. Eng., vol. 8, no 8, p. 3, 2014.

[45]. The Ethereum Wiki - Ehash. ethereum, 2019.

[46]. Transacción bitcoin. [En línea]. Disponible en: https://www.blockchain.com/fr/btc/tx/bc3ba69beecc84e3155a2308b61b7e5e33ce6f9d1b86f74c1bfc9873e87a7399?show_adv=true [Consultado el: 19-05-2022].

[47]. V. Buterin, « A NEXT GENERATION SMART CONTRACT & DECENTRALIZED APPLICATION PLATFORM », p. 36. 48. V. Buterin, On public and private blockchains (2015). 2015.

Índice de figuras

Figura 1. Diagrama de Gantt	11
Figura 2. Blockchain	17
Figura 3. Concepto de base de datos	17
Figura 4. Arquitectura de la Blockchain	18
Figura 5. Ledger	21
Figura 6. cadena de bloques	37
Figura 7. Forma simplificada de una transacción de Bitcoin	37
Figura 8. Ejemplo de transacción de Bitcoin	39
Figura 9. Forma simplificada de un bloque de Bitcoin	41
Figura 10. Principio de criptografía asimétrica	46
Figura 11. árbol de Merkle	48
Figura 12. Criptografía asimétrica: fases de firma-cifrado y verificación-descifrado	51
Figura 13. El estado de una cuenta en Ethereum	58
Figura 14. Componentes de una transacción de Ethereum	60
Figura 15. Estructura general de un contrato inteligente	69
Figura 16. Ejemplo de un contrato inteligente	70
Figura 17. Implementación de una transacción de Ethereum	71
Figura 18. Diagrama de caso de uso general	75
Figura 19. Gestión de clientes	76
Figura 20. Gestión de contratos	76
Figura 21. Gestión de propiedades	77
Figura 22. Gestión de cliente	77
Figura 23. Gestión de clase	78
Figura 24. Arquitectura de la aplicación	79
Figura 25. Estructura del proyecto	82
Figura 26. Ganache	83

Figura 27. Metamask	84
Figura 28. Estructura de los microservicios Java	85
Figura 29. Página de inicio	86
Figura 30. Activación del modo oscuro	86
Figura 31. Página quien somos	87
Figura 32. Página servicios	87
Figura 33. Página contáctenos	88
Figura 34. Página área de contrato	88
Figura 35. Detalles de un contrato	89
Figura 36. Buscar un contrato	89
Figura 37. Ordenar contratos	89
Figura 38. Dashboard cliente	89
Figura 39. Espacio contratos del cliente	90
Figura 40. Espacio contratos de clientes	90
Figura 41. Editar un perfil	90
Figura 42. Tutorial como agregar un contrato	91
Figura 43. tutorial de venta	91
Figura 44. tutorial de búsqueda	91
Figura 45. funciones del contrato	92
Figura 46. Parámetros del contrato	93
Figura 47. constructor public	94
Figura 48. constructor public	95
Figura 49. Log	95
Figura 50. Mia Token Etherscan	96
Figura 51. Mia Token verified contract	96

Índice de Tablas

Tabla 1. Estimación temporal	10
Tabla 2. Presupuesto Hardware	12
Tabla 3. Presupuesto Software	12
Tabla 4. Costes Indirectos	13