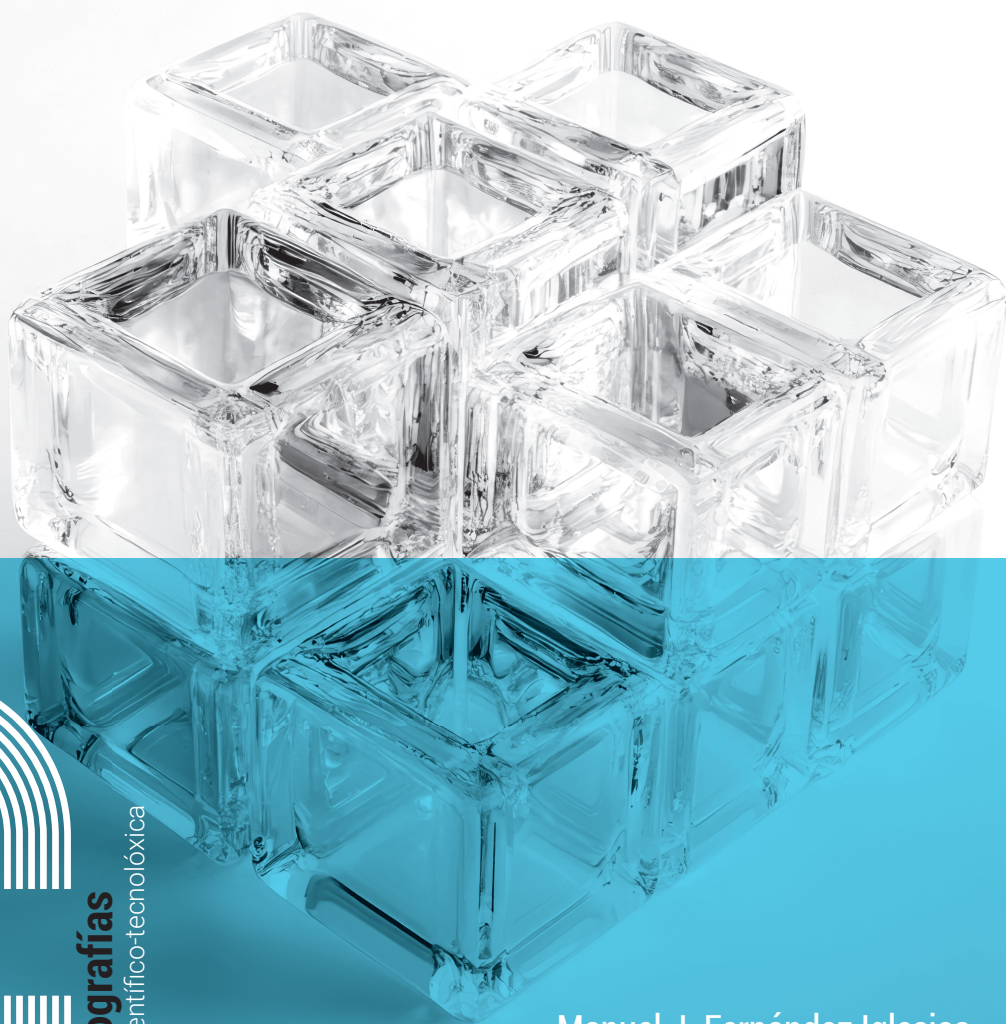


Blockchain, criptomoedas, NFT e aplicaciones descentralizadas

Fundamentos tecnológicos



Monografías

Serie científico-tecnológica

Manuel J. Fernández Iglesias
Christian von Eitzen Delgado
Luis E. Anido Rifón

Manuel J. Fernández Iglesias



Christian von Eitzen Delgado



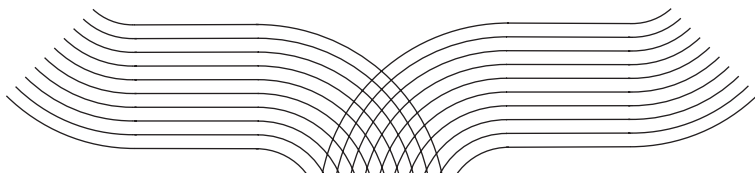
Luis E. Anido Rifón



Christian von Eitzen, Manolo Fernández e Luis Anido son pioneiros da investigación no ámbito das tecnoloxías de computación descentralizada en Galicia. Con contribucións relevantes publicadas en revistas de alto impacto nos campos da xestión de certificacións académicas e a rastrexabilidade, estes tres investigadores do Centro de Investigación en Telecomunicacións da Universidade de Vigo atlantTTic dedicaron unha parte importante da súa actividade académica dos últimos anos a estudar as tecnoloxías blockchain, aínda na súa infancia, co obxectivo de contribuír ao seu desenvolvemento para facilitar a súa adopción por parte do tecido produtivo e da sociedade en xeral. Christian, Manolo e Luis participan na redacción de informes técnicos para consorcios europeos e en comités internacionais de normalización das tecnoloxías blockchain aplicadas ao ámbito educativo.

Servizo de Publicacións

Universidade de Vigo



Monografías

Serie Científico-Tecnolóxica

n.º 032

Edición

Universidade de Vigo
Servizo de Publicacións
Rúa de Leonardo da Vinci, s/n. 36310 Vigo

Director

Jorge Luis Bueno Alonso

Consello editorial e científico

Humanidades e ciencias xurídico-sociais
Marta García González, Benigno Fernández Salgado
e Enrique J. Varela
Ciencia e Tecnoloxía
Ignacio Pérez Juste, Marta Pérez Rodríguez
e Ana Mª Bernabeu Tello

Deseño gráfico

Julinda Molares Cardoso e Tania Sueiro Graña
Área de Imaxe. Vicerreitoría de Comunicacións e
Relacións Institucionais

Maquetación e impresión: Tórculo Comunicación
Gráfica, S. A.

Imaxe da portada: Adobe Stock

ISBN (Libro impreso): 978-84-1188-052-7

Depósito legal: VG 55-2025

© Manuel J. Fernández Iglesias, Christian von Eitzen
Delgado
e Luis E. Anido Rifón
© Servizo de Publicacións da Universidade de Vigo, 2025

Sen o permiso escrito do Servizo de Publicacións da
Universidade de Vigo, queda prohibida a reprodución
ou a transmisión total e parcial deste libro a través
de ningún procedemento electrónico ou mecánico,
incluídos a fotocopia, a gravación magnética ou calquera
almacenamento de información e sistema de recuperación.

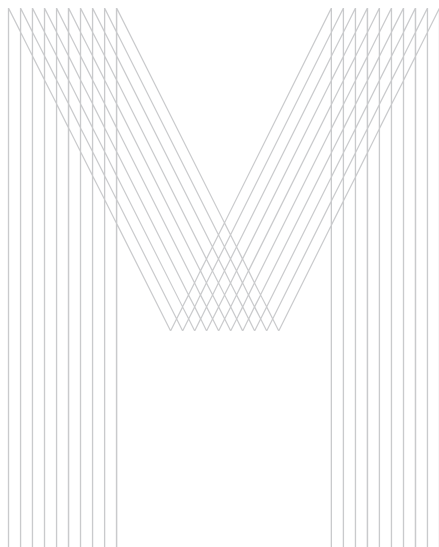
Ao ser esta editorial membro da **une**, garántense a
difusión e a comercialización das súas publicacións no
ámbito nacional e internacional.

Servizo de Publicacións

Universidade de Vigo



HR EXCELLENCE IN RESEARCH



Blockchain, criptomoedas, NFT e aplicacións descentralizadas

Fundamentos tecnolóxicos

Manuel J. Fernández Iglesias
Christian von Eitzen Delgado
Luis E. Anido Rifón

Prefacio

As cadeas de bloques ou *blockchains* e as aplicacións descentralizadas (dApps) son os compoñentes esenciais do campo emerxente das tecnoloxías descentralizadas. Unha cadea de bloques pódese ver como unha base de datos descentralizada que permite rexistrar transaccións seguras e transparentes entre iguais sen necesidade de intermediarios. As aplicacións descentralizadas ou dApps son aplicacións informáticas online que aproveitan a tecnoloxía blockchain para dar soporte a transaccións e servizos descentralizados sen necesidade dun ente central de supervisión. As dApps están deseñadas para funcionar de forma autónoma, sen necesidade dunha autoridade ou control central, e poden utilizarse para unha ampla gama de aplicacións en moi diversos ámbitos como as finanzas, a xestión de cadeas de subministración, a loxística, as redes sociais ou os xogos.

As cadeas de bloques e as dApps ofrecen unha nova e revolucionaria forma de construír e ofrecer servizos de forma descentralizada, segura e transparente. Do mesmo xeito que outras aplicacións e servizos dixitais, como as aplicacións distribuídas e mesmo as aplicacións de oficina, as tecnoloxías descentralizadas tamén poden desenvolverse, aloxarse, despregarse e explotarse na nube. A *nube*, no contexto dos servizos e aplicacións informáticas, refírese a un modelo de computación que permite o acceso baixo demanda a un conxunto compartido de recursos informáticos configurables (redes, servidores, almacenamento, aplicacións e servizos, ...) a través de internet. Estes recursos pódense asignar e e liberar de xeito rápido e áxil, cun mínimo esforzo de xestión ou interacción por parte do provedor de servizos. En esencia, a nube permite ás persoas usuarias e ás empresas acceder e utilizar servizos de computación, almacenamento e software sen necesidade de posuír ou manter a infraestrutura física subxacente. Isto ofrece vantaxes como escalabilidade, flexibilidade, redución de custos e acceso desde calquera lugar con conexión a internet.

As cadeas de bloques, como unha tecnoloxía máis dispoñible na nube, xorden para apoiar o desenvolvemento e despregamento de dApps. A computación en nube proporciona aos desenvolvedores acceso a recursos informáticos escalables e flexibles que poden utilizarse para executar dApps e almacenar datos nunha rede blockchain. Mediante o modelo de provisión na nube, é posible crear e despregar dApps sen necesidade dun investimento inicial significativo en infraestrutura ou coñecementos técnicos, o que facilita o acceso ao mundo das tecnoloxías descentralizadas e a adopción de solucións baseadas nas devanditas tecnoloxías, que á súa vez son a base dunha nova xeración de servizos en liña baseada na descentralización: a Web3 ou Web 3.0.

A pesar do rebumbio que teñen xerado conceptos como as criptomoedas ou os tokens non funxibles ou NFT, as cadeas de bloques representan unha revolución silenciosa que está a remodelar o tecido mesmo da nosa sociedade dixital. Esta tecnoloxía, nacida da innovación e o desexo dun mundo máis transparente e equitativo, promete transformar non só as nosas transaccións financeiras ou o xeito de entender o coleccionismo, senón a forma en que interactuamos, confiamos e colaboramos no mundo dixital. Podemos imaxinar un futuro onde a confianza non dependa de intermediarios que concentran todo o poder, senón que estea codificada na propia estrutura das nosas redes. Un mundo onde a transparencia e a inmutabilidade da información sexan a norma, non a excepción. Onde artistas, emprendedores e visionarios poidan desbloquear novas formas de crear valor e conectar con audiencias globais sen barreiras. Blockchain non é só unha tecnoloxía; é unha porta de entrada a unha novo paradigma de posibilidades, un catalizador para a innovación que ten o potencial de democratizar o acceso a novas oportunidades tecnolóxicas e empoderar ás persoas en todo o planeta. Ao embarcarnos nesta viaxe para entender as cadeas de bloques, non só estamos a aprender sobre unha tecnoloxía; estamos a albiscar os cimentos dun futuro máis aberto, xusto e conectado.

Pero o mundo das blockchains tamén é un mundo de constantes sorpresas e eventos inesperados. Poucos acontecementos cativaron tanto a imaxinación do público como a creación da criptomoeda de Donald Trump, \$TRUMP. Esta mememoeda, lanzada apenas uns días antes da súa toma de posesión como cuadraxésimo sétimo presidente de Estados Unidos de América en xaneiro de 2025, disparouse ata alcanzar unha capitalización de mercado de 10.000 millóns de dólares en apenas unha fin de semana. O seu rápido ascenso, xunto ao da criptomoeda \$MELANIA emitida pola súa dona, desencadearon un bulebule de incredulidade e controversia, mesturando os mundos das finanzas

descentralizadas, a política e o cotilleo dunha forma nunca vista. Os analistas do mercado apresuráronse a interpretar as implicacións, e mesmo algúns interpretaron a xogada como un paso audaz cara á xeneralización das criptomonedas, mentres que e outros cuestionaron as súas ramificacións éticas e normativas.

Máis aló do seu atractivo especulativo, estas moedas puxeron de relevo como a tecnoloxía blockchain estase entretecendo no tecido dos acontecementos mundiais e remodelando o discurso público. Con blockchain á vangarda da innovación tecnolóxica, este manual pretende ofrecer ás lectoras e lectores unha comprensión profunda dos principios, os retos e o potencial transformador desta tecnoloxía, utilizando estes momentos revolucionarios como lente para explorar o seu papel evolutivo na sociedade.

Índice xeral

Índice de figuras	9
Índice de cadros	11
1. Introducción	13
2. As cadeas de bloques	17
2.1. Estrutura da cadea de bloques	20
2.2. Participantes nunha rede blockchain	21
2.2.1. Os nodos	21
2.2.2. Os mineiros ou validadores	22
2.2.3. Os usuarios	23
2.2.4. Os desenvolvedores	24
2.2.5. Os reguladores	24
2.3. Tipos de cadeas de bloques	26
2.3.1. Cadeas públicas	26
2.3.2. Cadeas privadas	27
2.4. Mecanismos de consenso	28
2.4.1. Proba de traballo	29
2.4.2. Proba de participación	29
2.4.3. Proba de queima	30
2.4.4. Tolerancia a fallos bizantinos	30

2.5. Creación de novos bloques	31
2.5.1. Minería de bloques e proba de traballo	32
2.5.2. Forxa de bloques e proba de participación	36
2.5.3. Sacrificio e proba de queima	38
2.5.4. Compromiso e tolerancia a fallos bizantina	38
2.6. As bifurcacións nas cadeas de bloques	41
2.7. O reto da escalabilidade	42
2.8. Vantaxes e inconvenientes das cadeas de bloques	44
2.9. Aspectos de seguridade	46
3. Criptomoedas	51
3.1. Definición e historia	51
3.2. Algunhas criptomoedas populares	53
3.3. Criptomoedas e tokenización	59
3.4. Creación de criptomoeda	60
3.5. Aplicacións e casos de uso	63
3.6. Desafíos e implicacións económicas e legais	66
3.7. O futuro das criptomoedas	69
4. Contratos intelixentes	75
4.1. Para que se utilizan os contratos intelixentes	75
4.2. Construindo contratos intelixentes e dApps	79
4.3. O custo de operar en Ethereum: o gas	89
4.4. Contratos intelixentes e criptomoedas	93
4.5. Interacción coa blockchain dende un contrato intelixente . . .	100
4.6. Aspectos de seguridade	101
4.7. Contratos intelixentes e enxeñería do software	104
5. Sistemas de arquivos descentralizados	107

5.1. Limitacións dos sistemas de arquivos tradicionais	108
5.2. Fundamentos dos sistemas de arquivos descentralizados . . .	109
5.3. Arquitectura e funcionamento	110
5.4. InterPlanetary File System	112
5.5. IPFS e as cadeas de bloques: casos de uso	115
5.6. Outros sistemas de arquivos descentralizados	117
6. Oráculos	119
6.1. Arquitecturas de oráculos	122
6.2. Oráculos e finanzas descentralizadas	123
6.3. Os oráculos e a xeración de números aleatorios	128
6.4. Redes de oráculos e oráculos entre cadeas	132
6.5. Aspectos de seguridade	134
7. Tokens non funxibles	137
7.1. Creación dun NFT	139
7.2. Combinando tokens funxibles e non funxibles	148
8. Finanzas descentralizadas	157
8.1. Mercados financeiros descentralizados	158
8.2. Préstamos descentralizados	161
8.3. Minería de liquidez	162
8.4. Derivados DeFi e xestión de activos	163
8.5. Retos e consideracións normativas	165
9. Blockchain como servizo na nube	169
9.1. Como funciona Blockchain como Servizo	170
9.2. Exemplos de plataformas BaaS	172
9.2.1. IBM Blockchain Platform	172
9.2.2. Amazon Managed Blockchain	174

9.3. Como comezar con BaaS	175
10. Recapitulación e tendencias futuras	177
10.1. Para saber máis	178
A. Criptografía de clave pública	181
B. Funciones de hash	187
C. Árbores de Merkle	189
D. Prueba de participación en Ethereum	193
E. Redes peer-to-peer	197
Bibliografía	201
Glosario	209
Índice de materias	229

Índice de figuras

2.1. Exemplo de cadea de bloques	20
3.1. Evolución do prezo do Bitcoin	55
3.2. Can de raza Shiba Inu do meme Doge	56
3.3. Evolución do prezo dalgunhas criptomoedas	58
3.4. Xeración de nova criptomoeda	72
3.5. Compra de criptomoeda	73
4.1. Exemplo de dApp	77
4.2. fallback() ou receive()	97
6.1. Rede de cadea de bloques con oráculos	119
7.1. Activo dixital para crear un NFT	141
7.2. NFT na carteira Metamask	148
7.3. Activo dixital para crear un novo NFT	154
C.1. Exemplo de árbore de Merkle	190
E.1. Rede entre pares	198

Índice de cadros

- 2.1. Exemplo de carga de traballo 35
- 3.1. Criptomoedas populares 54
- 4.1. Propiedades de block, msg e tx 102
- 7.1. NFT en Axie Infinity 138
- 9.1. Provedores BaaS 170

Capítulo 1

Introdución

Unha cadea de bloques é un rexistro de información descentralizado e espallado nun conxunto de ordenadores conectados en rede. Cada bloque da cadea almacena e notariza un conxunto de transaccións ou operacións, e unha vez que un bloque se engade á cadea, non pode alterarse sen o consenso dos nodos que forman a rede. É dicir, unha vez que unha transacción pasa a formar parte da cadea de bloques, é permanente e inmutable.

As transaccións poden representar transferencias de información, activos ou valor entre persoas ou entidades, e tamén poden acreditar a orixe ou a titularidade de calquera activo que teña unha representación dixital. Por tanto, unha cadea de bloques pódese ver como un notario dixital que posúe unha base de datos que almacena todas as transaccións que se produciron na devandita rede desde a súa creación.

Outro símil habitual para describir unha cadea de bloques é o dun libro maior de contabilidade (*ledger* en inglés), distribuído e inmutable, onde se recollen todas as transaccións realizadas. As cadeas de bloques utilizan técnicas criptográficas, como a criptografía de clave pública ou as funcións hash (ver apéndices A e B), para asegurar a rede e garantir a privacidade e autenticidade dos datos almacenados nela.

Os nodos dunha rede blockchain interactúan entre si mediante un protocolo de rede entre iguais ou protocolo peer-to-peer (P2P, ver apéndice E e figura E.1). Cada nodo está conectado a varios nodos da rede e pode enviar e recibir datos cara a e desde estes nodos. Cando un nodo se une á rede, primeiro ten que atopar outros nodos aos que conectarse. Para iso, pode utilizar un nodo de inicio, que é un nodo ben coñecido na rede que ten unha lista doutros nodos aos que o novo nodo pode conectarse. Unha vez que o novo nodo está conectado, pode utilizar un protocolo de descubrimento para atopar outros

nodos aos que conectarse. Unha vez que un nodo está conectado a outros nodos, pode intercambiar mensaxes con eles utilizando o protocolo P2P. Estas mensaxes poden incluír transaccións, bloques e outros datos relacionados coa cadea de bloques. Cada nodo verifica os datos que recibe doutros nodos para asegurarse de que son válidos e non foron manipulados.

Os nodos tamén poden comunicarse entre si para chegar a un consenso sobre o estado da cadea de bloques. Para iso utilízase un algoritmo de consenso. Un algoritmo de consenso é un mecanismo que permite a múltiples participantes ou nodos na rede chegar a un acordo sobre o estado actual da cadea de bloques, sen necesidade dunha autoridade central. Este proceso asegura que todos os nodos da rede manteñan unha visión consistente e actualizada da información, a pesar de posibles fallos, atrasos na comunicación ou intentos de manipulación. Noutras palabras, o algoritmo de consenso garante que todos os nodos están de acordo no mesmo conxunto de transaccións e bloques. Dependendo do algoritmo de consenso utilizado, os nodos poden ter que realizar diferentes funcións, como validar transaccións ou fabricar novos bloques. En xeral, a interacción entre os nodos dunha rede blockchain é esencial para manter a integridade e a seguridade da rede, así como para garantir que todos os nodos teñan unha visión coherente da blockchain.

A natureza descentralizada da cadea de bloques significa que non hai unha autoridade central que a controle, o que a fai resistente a ataques e a manipulación. No seu lugar, cada nodo da rede ten unha copia da cadea completa. Isto garante que non haxa un único punto de fallo na rede e que a propia cadea de bloques sexa resistente á manipulación. Cada bloque da cadea contén un conxunto de transaccións, e unha vez que un bloque se engade á cadea, non pode alterarse. Isto tamén o converte nun sistema que non require unha autoridade central de confianza, o que significa que os participantes poden confiar nos datos almacenados na cadea de bloques sen depender de ningún intermediario. Como xa se dixo, a cadea de bloques utiliza no seu lugar un mecanismo de consenso para garantir que todos os nodos da rede están de acordo sobre o estado da cadea.

As cadeas de bloques xa se utilizan en moi diversas aplicacións, como as criptomoedas (Yuan y Wang, 2018), a xestión de cadeas de subministración (Abdallah y Nizamuddin, 2023; Salah et al., 2019), a xestión de activos financeiros (Dos Santos et al., 2022), a educación (Delgado-von-Eitzen, Anido-Rifón y Fernández-Iglesias, 2021b), a xestión de ecosistemas do Internet das Cousas (Wang et al., 2020) ou os sistemas de votación (Jafar, Aziz y Shukur, 2021), entre moitas outras. A cadea de bloques máis coñecida é Bitcoin, que se utiliza

para almacenar e transferir a criptomoeda Bitcoin (BTC). Esta foi a cadea orixinal proposta por Satoshi Nakamoto (2008a), a persoa á que se lle atribúe a primeira aplicación exitosa desta tecnoloxía.

Outra plataforma moi popular é Ethereum (Buterin 2014), unha cadea de bloques programable que permite aos desenvolvedores crear aplicacións descentralizadas (dApps) mediante contratos intelixentes. Tamén dispón da súa propia criptomoeda (Ethereum, ETH) e é a segunda maior cadea de bloque por capitalización, despois de Bitcoin. Outras cadeas populares, coas súas respectivas criptomoedas, son BNB Chain (BNB) (Chain, 2020), Cardano (ADA) (Hoskinson, 2017), Polkadot (DOT) (Wood, 2016) ou Solana (SOL) (Yakovenko, 2017).

A evolución e expansión do mundo das cadeas de bloques, os contratos intelixentes e as dApps deron lugar a unha ampla gama de tecnoloxías e aplicacións que están a transformar diversos sectores da sociedade. Neste manual, exploraremos os fundamentos esenciais das cadeas de bloques, presentando os diferentes tipos de cadeas de bloques, a súa estrutura e funcionamento básicos, así como as súas vantaxes e inconvenientes. Ademais, profundaremos no mundo dos contratos intelixentes e as dApps, falando da súa programación, seguridade e a súa relación coas criptomoedas e os tokens non funxibles (NFTs). Exploraremos tamén o campo das finanzas descentralizadas ou DeFi, que abarca desde os mercados financeiros descentralizados ata os préstamos e a minería de liquidez. Adicionalmente, examinaremos os oráculos e os sistemas de arquivos descentralizados, así como o concepto de blockchain como servizo (BaaS) dispoñible na nube. Finalmente, concluiremos o noso percorrido explorando os desenvolvementos máis recentes no ámbito das blockchains e ofrecendo recursos adicionais para aquelas persoas interesadas en profundar no tema.

Este manual está cheo de exemplos prácticos de como se fan realidade os distintos conceptos que configuran a tecnoloxía de cadeas de bloques. Estes exemplos requiren de coñecementos básicos de informática e programación, e poden pasarse por alto se aínda non tes estes coñecementos. En calquera caso, ten en conta que o ecosistema blockchain evoluciona rapidamente, con continuos avances en tecnoloxía, estándares e mellores prácticas. Do mesmo xeito, as ferramentas de apoio e os entornos tecnolóxicos e bibliotecas de software para desenvolver estes exemplos e interactuar coas redes blockchain tamén evolucionan a un ritmo rápido. Como resultado, aínda que os exemplos proporcionados aquí foron comprobados e funcionan correctamente no momento que este manual sae da imprenta, poden requirir actualizacións ou

correccións nun futuro próximo para seguir sendo compatibles coas últimas versións das plataformas, bibliotecas e estándares de blockchain. É moi recomendable manterse informados sobre os últimos avances no ámbito do blockchain, comprobar a compatibilidade do código coas versións máis actualizadas do software e sobre todo actuar con cautela ao despreparar o código en contornas de produción.

Capítulo 2

As cadeas de bloques

Unha cadea de bloques é un conxunto descentralizado e distribuído de rexistros dixitais (os bloques) que están enlazados se xeito secuencial (é dicir, sempre podemos identificar o primeiro bloque da cadea e o derradeiro bloque que temos engadido) e asegurados mediante criptografía. Decimos que unha cadea de bloques funciona de xeito descentralizado e distribuído porque dita cadea está espallada polos nodos que forman unha rede, de xeito que cada nodo ten unha copia completa de toda a cadea con todos os bloques, e todos os nodos funcionan de xeito coordinado sen que ningún deles teña un papel preponderante ou central. Os nodos ponse de acordo para ter todos a mesma versión válida da cadea de bloques en cada momento.

Cada bloque contén un conxunto de transaccións e unha referencia ao bloque anterior, creando unha cadea inmutable de información. As transaccións son basicamente mensaxes que xera un usuario e que están dirixidas a outro usuario. Normalmente, as transaccións están asinadas polo usuario que as xera mediante unha sinatura baseada nun sistema de criptografía de clave pública (ver apéndice A). Os usuarios identifícanse exclusivamente pola súa dirección, que é basicamente a clave pública que se corresponde coa clave privada utilizada para asinar as transaccións. Polo tanto, calquera pode comprobar que determinada transacción foi efectivamente xerada por determinada dirección da blockchain.

As transaccións agrúpanse en bloques e engádense á cadea de bloques en orde cronolóxica. Cada bloque contén un hash criptográfico único que o vincula ao bloque anterior, creando un libro de rexistro inmutable, seguro e a proba de manipulacións. A seguridade, integridade e validez das transaccións mantéñense a través dun consenso entre todos participantes da rede, utilizando os mecanismos de consenso que veremos no apartado 2.4. Un mecanismo de consenso é un conxunto de regras e protocolos que garanten que todos os

participantes da rede están de acordo na validez das transaccións incluídas nos bloques e a orde en que se engaden estes á cadea de bloques.

O proceso para engadir transaccións á cadea de bloques comeza entón cando un usuario asina a transacción coa súa clave privada, por exemplo unha solicitude para transpasar certa cantidade dunha criptomoeda a outra dirección da blockchain. Esa transacción asinada envíase a un dos nodos da rede blockchain elixido polo usuario que inicia a transacción.

Unha vez que a transacción chega ao nodo inicial, este fai unha primeira verificación da mesma para comprobar que a firma é válida e que ten o formato correcto. Se é válida, o nodo engádea á súa pila de transaccións pendentes de procesar, coñecida como *mempool*, e propágaa a outros nodos conectados á rede de nodos que dá soporte á cadea de bloques.

Uns nodos especiais, chamados mineiros ou validadores dependendo dos mecanismos de consenso que veremos máis adiante, seleccionan transaccións do *mempool* para incluílas nun novo bloque. Xeralmente, elixen primeiro as transaccións que van dar maior beneficio aos devanditos mineiros ou validadores. Unha vez que un mineiro ou validador seleccionou o número necesario de transaccións para xerar un novo bloque, tenta xerar devandito bloque para engadilo á cadea. Para iso competirá con outros mineiros ou validadores.

Cando se minou ou validou o novo bloque, este propágase a través da rede de nodos. Outros nodos da rede verifican a validez do bloque e de todas as transaccións incluídas, e engádeno á súa copia da cadea. Máis tarde ou máis cedo, o nodo utilizado orixinalmente para enviar a transacción recibirá a súa copia do novo bloque e engadirao á súa copia da cadea. Nese momento, a dirección que iniciou a transacción obtén unha primeira confirmación, xa que neste punto a transacción xa está *na blockchain*, pero aínda non se considera totalmente garantido que a transacción se teña completado definitivamente.

Cada novo bloque engadido á cadea con outras transaccións posteriores despois do que contén a transacción orixinal conta como unha confirmación adicional. O número de confirmacións necesarias para considerar unha transacción *rexistrada inmutablemente* varía segundo a blockchain e o valor das transaccións. En calquera caso, despois dun certo número de confirmacións (que varía segundo a rede), a transacción considérase *final* ou inmutable.

Unha vez que un bloque foi confirmado definitivamente, o estado da blockchain actualízase para reflectir os cambios introducidos polas transaccións incluídas nel (por exemplo, os novos saldos das contas orixe e destino).

É común esperar varias confirmacións para considerar que un bloque foi engadido de maneira definitiva á cadea debido á posibilidade de que ocorran reestruturacións ou bifurcacións temporais na rede. Isto ocorre cando varios mineiros ou validadores xeran un bloque válido case simultaneamente, o que pode resultar en múltiples cadeas temporais (ou bifurcacións) que se distribúen entre os nodos.

Os mecanismos de comunicación entre os nodos están deseñados para que, máis tarde ou máis cedo, a rede seleccione a cadea máis longa e descarte as outras. Se dous mineiros minan simultaneamente un bloque válido, créase unha bifurcación temporal na cadea de bloques e a rede terá dúas versións válidas da cadea de bloques. O resto dos nodos da rede engadirán o bloque que reciban primeiro á súa copia local da cadea de bloques e, a partir de aí, seguirán ampliando a cadea. Isto crea unha situación na que algúns nodos teñen unha versión da cadea de bloques e outros nodos teñen a outra versión.

Nalgún momento, a rede converxerá elixindo a cadea máis longa, xa que os nodos sempre deben elixir estender a cadea máis longa que coñezan. Isto significa que un dos dous bloques en competencia será descartado, e a rede seguirá construíndo sobre a cadea máis longa. En consecuencia, os mineiros ou validadores deben esperar a que o seu bloque sexa confirmado pola rede antes de dar por feito que se engadiu definitivamente á cadea de bloques. Pola contra, arríscanse a malgastar recursos nun bloque que, en última instancia, pode ser descartado en favor dun bloque competidor.

Esperar varias confirmacións reduce o risco de que unha transacción validada nun bloque que podería ser descartado termine quedando *orfa* (sen un bloque onde rexistrarse) ou substituída por outra no bloque que finalmente sobreviva. Se a transacción está nun bloque que queda nunha rama que finalmente non é aceptada pola rede, anúlase co resto das transaccións dese bloque. Canto máis arriba na cadea atópase o bloque coa transacción (é dicir, cantas máis confirmacións ten), menor é a probabilidade de que a cadea con ese bloque sexa substituída.

Este proceso pode tomar desde segundos ata varios minutos, dependendo da conxestión da rede (cantas transaccións vaian entrando no mempool pendentes de minar ou validar), as tarifas pagas (canto máis se paga, máis rápido minarase ou validarse unha transacción), e o mecanismo de consenso específico da blockchain en cuestión.

Nos apartados seguintes analizaremos polo miúdo os diferentes elementos das cadeas de bloques que acabamos de presentar e que fan que unha

blockchain funcione. Comezaremos explorando a estrutura e os elementos básicos dunha cadea de bloques, seguido da presentación dos diversos actores que dan vida a estas redes. Analizaremos os diferentes tipos de blockchains e os mecanismos de consenso que aseguran a súa integridade e profundaremos no proceso de creación de novos bloques, comentando o fenómeno das bifurcacións. Tamén examinaremos os desafíos de escalabilidade que enfrontan estas tecnoloxías e sopesaremos as súas vantaxes e inconvenientes. Finalmente, falaremos dos aspectos relacionados coa seguridade destas redes descentralizadas. A través deste percorrido, obteremos unha comprensión integral de como funciona unha blockchain e por que captou a atención de industrias e gobernos en todo o mundo.

2.1 Estrutura da cadea de bloques

Como acabamos de ver, unha cadea de bloques está composta dunha serie de bloques enlazados uns con outros. Un bloque da cadea de bloques é un conxunto de datos que empaqueta un grupo de transaccións que foron validadas e engadidas á cadea de bloques. Cada bloque está vinculado ao bloque anterior, formando unha cadea que contén un historial completo e inalterable de todas as transaccións realizadas na rede, desde a súa creación.

Como podemos ver na figura 2.1, cada bloque comeza cunha cabeceira que contén metadatos sobre o bloque, incluído un resume criptográfico ou hash do bloque, o hash do bloque anterior, a marca de tempo do bloque (que non aparece na figura para non sobrecargala) e, de ser necesario, un *nonce*. O *nonce* é un valor aleatorio utilizado no proceso de minería das redes de proba de traballo, diferente para cada bloque minado, necesario para resolver o desafío computacional do devandito mecanismo de consenso (Wang et al., 2019).

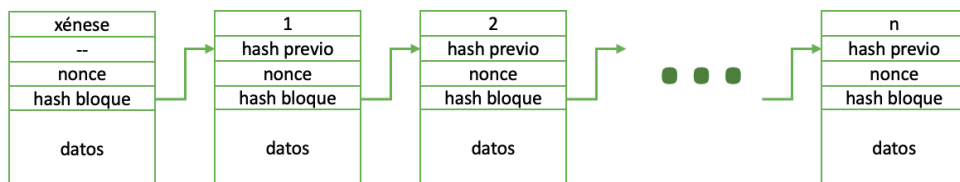


Figura 2.1. Exemplo ilustrativo da estrutura dunha cadea de bloques.

Ademais, cada bloque contén unha lista de transaccións validadas que se incluíron no bloque. Para protexer este contido fronte a manipulacións e

outros ataques e garantir a súa integridade e inmutabilidade, o bloque tamén contén unha clave hash que resume todas as transaccións do bloque. Dita clave correspóndese coa raíz da árbore de Merkle de todas as transaccións do bloque (ver apéndice C e figura C.1).

Aínda que tampouco aparece na figura, o bloque ten tamén un campo que indica o tamaño de bloque en bytes. Trátase dun elemento importante nas redes blockchain con capacidade de almacenamento limitada. Outro campo indica a posición do bloque na cadea de bloques, chamada *altura do bloque*. Cada bloque da cadea de bloques ten unha altura única, e a altura de cada bloque baséase na altura do bloque anterior. O campo de altura do bloque xunto co hash do bloque anterior e a marca temporal da cabeceira, serven para garantir o ordenamento dos bloques na cadea e a súa inmutabilidade. O hash do propio bloque serve para garantir que ningún dos campos do bloque foi alterado.

Finalmente, nas redes de proba de traballo (ver apartado 2.4) inclúese un campo de dificultade que indica o nivel de dificultade computacional requirido para minar o bloque. Este valor axústase periodicamente para manter un ritmo constante de creación de bloques nestas redes.

2.2 Participantes nunha rede blockchain

Os participantes clave nunha rede blockchain poden variar en función do tipo de cadea de bloques, pero os elementos comúns á maioría delas son os nodos da rede, os mineiros ou os validadores en función do mecanismo de consenso, os usuarios, os desenvolvedores de software ou programas informáticos para a blockchain, e os reguladores. Todos os participantes nunha rede blockchain traballan conxuntamente para garantir a seguridade, fiabilidade e funcionalidade da rede. Cada un ten as súas funcións e responsabilidades, pero entre todos desempeñan un papel fundamental no funcionamento e no éxito da cadea de bloques. Nos parágrafos seguintes describimos brevemente cada un destes perfís.

2.2.1 Os nodos

Os nodos son ordenadores ou servidores conectados á rede que validan as transaccións, manteñen unha copia da cadea e transmiten información a outros nodos da rede. Utilizan un protocolo de comunicación entre pares ou protocolo P2P para comunicarse entre eles (ver apéndice E). Cada nodo

da rede blockchain ten unha dirección única que o identifica dentro da rede, e ademais de realizar as tarefas indicadas para garantir o bo funcionamento da cadea, comparte co resto dos nodos información sobre o seu estado e as súas capacidades. Por exemplo, a versión do software de blockchain que está a executar o nodo é relevante porque os nodos que executan software obsoleto poden correr o risco de sufrir vulnerabilidades, e probablemente serán excluídos da rede blockchain. Ademais, cada nodo leva conta do número de bloques da cadea que ese nodo validou, o que axuda a outros nodos da rede a determinar a validez e precisión da cadea de bloques.

A información sobre o estado de conexión indica se o nodo está actualmente conectado a outros nodos da rede. Os nodos que non están conectados á rede non poden validar transaccións nin transmitir información. A cantidade de memoria e almacenamento dispoñibles no nodo está relacionado directamente coas súas contribucións á rede, xa que os nodos con memoria ou capacidade de almacenamento limitados poden non ser capaces de participar en certas funcións da mesma.

2.2.2 Os mineiros ou validadores

Os mineiros son aqueles nodos nunha blockchain que compiten para resolver desafíos matemáticos complexos co obxectivo de engadir novos bloques á cadea e con iso gañar recompensas, en nova criptomoeda (recompensa xerada pola propia blockchain por minar un novo bloque), e a través das taxas cobradas aos usuarios polas transaccións que realizan. Pola súa banda, os validadores son nodos dunha cadea de bloques que se encargan de validar as transaccións e engadir novos bloques á cadea de bloques, pero sen requirir resolver os desafíos matemáticos que teñen que resolver os mineiros. En vez de pola súa capacidade de computación, os validadores elíxense en función da súa participación na rede, medida habitualmente en función da contía dun aval en criptomoeda, e tamén obteñen recompensas pola súa participación, neste caso proporcionais ao aval depositado.

Que unha blockchain teña mineiros ou validadores depende do deseño desta. A diferenza clave entre mineiros e validadores é que, por unha banda, os mineiros utilizan hardware especializado (chips especializados, procesadores gráficos potentes) para resolver os quebracabezas ou desafíos matemáticos, e en consecuencia adoitan consumir unha cantidade importante de enerxía. Podemos dicir que a confianza na seguridade e inmutabilidade dunha cadea de bloques con mineiros baséanse na potencia computacional distribuída destes

mineiros. Por outra banda, os validadores non requiren hardware especializado e poden operar utilizando computadores normais, polo que consomen significativamente menos enerxía que os mineiros. Neste caso, a seguridade da rede fundaméntese no valor económico dos avais que depositan os validadores.

Partindo do feito de que a característica fundamental dunha blockchain é a descentralización, a ameaza principal das cadeas de bloques baseadas na minería é o risco de centralización por concentración de potencia de cálculo, mentres que o risco principal das redes baseadas en avais é o risco de centralización por concentración de riqueza.

2.2.3 Os usuarios

Os usuarios son persoas ou organizacións que utilizan a rede blockchain para enviar e recibir transaccións. Os usuarios participan libremente e de maneira anónima nas redes públicas e requiren de identificación e rexistro na redes privadas.

Como veremos noutros capítulos, os usuarios de blockchain teñen á súa disposición unha variedade de accións e oportunidades dentro deste ecosistema. Poden realizar transaccións de criptomoedas directamente entre eles, enviando e recibindo activos dixitais sen a necesidade dos intermediarios tradicionais. Ademais, teñen a capacidade de participar na gobernanza da propia blockchain e de outros proxectos descentralizados, votando sobre propostas e cambios nos protocolos. Os usuarios poden involucrarse en finanzas descentralizadas, prestando, pedindo prestado ou proporcionando liquidez a diferentes proxectos. Teñen a opción de crear, comprar, vender e coleccionar tokens non funxibles que representan arte dixital, obxectos virtuais ou mesmo propiedades tokenizadas. Poden participar na seguridade da rede a través da minería ou o mediante avais, dependendo do deseño da blockchain.

Os usuarios tamén poden desenvolver e despregar contratos intelixentes, creando aplicacións descentralizadas que operan de maneira autónoma na blockchain. A través dos contratos intelixentes, teñen a posibilidade de almacenar datos de maneira segura e descentralizada, aproveitando a inmutabilidade da cadea de bloques. Poden participar en mercados predictivos, apostando sobre resultados futuros en diversos campos. Ademais, os usuarios poden involucrarse en xogos blockchain, onde poden posuír e comerciar con activos do xogo que teñen valor real. Tamén poden participar en sistemas de identidade descentralizada, controlando e compartindo os seus datos persoais de maneira selectiva.

En resumo, os usuarios de blockchain teñen un amplo espectro de posibilidades que abarcan desde simples transaccións ata complexas interaccións con sistemas financeiros e aplicacións descentralizadas, todo iso mentres manteñen un alto grao de control sobre os seus activos e datos dixitais.

2.2.4 Os desenvolvedores

Os desenvolvedores son individuos ou organizacións que crean e manteñen o software blockchain e as aplicacións que se executan na capa superior da rede, como os contratos intelixentes e as aplicacións descentralizadas.

Os desenvolvedores teñen á súa disposición unha ampla gama de oportunidades para innovar e construír. Poden crear e despregar contratos intelixentes, que son a columna vertebral de moitas aplicacións descentralizadas, permitindo automatizar procesos e crear novas formas de interacción dixital. Teñen a capacidade de desenvolver carteiras, mellorando a experiencia do usuario na xestión de activos dixitais. Os desenvolvedores poden construír interfaces de usuario para aplicacións descentralizadas, facendo que as aplicacións blockchain sexan máis accesibles para o público en xeral.

A parte de desenvolver aplicacións para o público en xeral, tamén teñen a opción de traballar na capa base da blockchain, mellorando os protocolos existentes ou creando uns novos para abordar desafíos como a escalabilidade e a interoperabilidade. Poden desenvolver oráculos, que son fundamentais para conectar datos do mundo real con contratos intelixentes.

Tamén teñen a posibilidade de crear ferramentas de análise e visualización de datos blockchain, proporcionando información valiosa sobre as transaccións e o comportamento da rede, e teñen a opción de desenvolver entornos de traballo e bibliotecas que faciliten a outros desenvolvedores a creación de aplicacións blockchain.

En esencia, os desenvolvedores teñen o poder de dar forma ao futuro da tecnoloxía blockchain, creando as ferramentas, aplicacións e protocolos que definirán como interactuamos cos sistemas descentralizados nos vindeiros anos.

2.2.5 Os reguladores

Os reguladores son axencias gobernamentais ou outras organizacións que supervisan e regulan o uso da rede blockchain, especialmente nos casos

nos que se dá soporte ao intercambio de valores, materias primas ou outros activos regulados. O seu papel é máis relevante nas redes empresariais, sobre todo naquelas que operan en mercados regulados.

Os reguladores enfróntanse a unha serie de desafíos complexos e en constante evolución no contexto das cadeas de bloques. Deben equilibrar a necesidade de protexer aos consumidores e manter a estabilidade financeira co desexo de non sufocar a innovación neste campo emerxente. Un dos principais retos é a natureza transfronteiriza das blockchain, que dificulta a aplicación de leis e regulacións nacionais nun ecosistema inherentemente global. Os reguladores tamén se enfrontan á tarefa de clasificar e categorizar os diversos tipos de criptoactivos, como as moedas dixitais ou os tokens funxibles e non funxibles, cada un coas súas propias implicacións legais e financeiras.

Deben abordar as preocupacións sobre o lavado de diñeiro e o financiamento do terrorismo, desenvolvendo marcos que permitan a rastrexabilidade das transaccións sen comprometer a privacidade dos usuarios lexítimos. A volatilidade dos criptoactivos expón desafíos para a protección do investidor e a estabilidade do mercado, requirindo novos enfoques para a supervisión e a xestión do risco.

Pola outra banda, os reguladores deben manterse ao día coas rápidas innovacións tecnolóxicas, como os contratos intelixentes e as finanzas descentralizadas, que poden expor novos riscos e desafíos regulatorios.

Ademais, deben abordar as cuestións de gobernanza nas organizacións autónomas descentralizadas e outros sistemas blockchain, onde a responsabilidade e a xurisdición poden non estar claramente definidas. Os reguladores tamén se enfrontan ao desafío de fomentar a interoperabilidade entre diferentes blockchains e cos sistemas financeiros tradicionais, á vez que garanten a seguridade e a integridade do sistema no seu conxunto.

Finalmente, deben navegar polas implicacións fiscais das transaccións e ganancias relacionadas coas criptomonedas, desenvolvendo marcos que sexan xustos e aplicables nunha contorna dixital descentralizada. En última instancia, os reguladores deben atopar un equilibrio delicado entre a protección e a innovación, adaptando constantemente os seus enfoques a medida que a tecnoloxía blockchain continúa evolucionando e transformando o panorama financeiro e tecnolóxico global.

2.3 Tipos de cadeas de bloques

A clasificación máis xeral das cadeas de bloques baséase nos requisitos que se impoñen aos usuarios para poder realizar transaccións nelas (Miller, 2019). De acordo con este criterio, podemos dividir as cadeas de bloques en públicas e privadas. En xeral, aínda que tanto as cadeas de bloques privadas como as públicas utilizan a tecnoloxía de libro maior distribuído e descentralizado, teñen diferentes consideracións de deseño e serven a diferentes propósitos. As cadeas de bloques públicas priorizan a descentralización, a transparencia e a seguridade para as transaccións entre pares, mentres que as cadeas de bloques privadas priorizan a escalabilidade, a privacidade e a interoperabilidade para o intercambio seguro e eficiente de datos dentro dunha rede ou consorcio composto por entidades ou persoas ás que explicitamente se lles dá permiso para operar na rede.

Nos parágrafos seguintes describimos ambos tipos, xunto cos seus usos e características máis salientables.

2.3.1 Cadeas públicas

Unha blockchain pública, sen permisos ou non permissionada (*non-permissioned blockchain* en inglés) está aberta a calquera que desexe participar na mesma, sen restricións nin requisitos de verificación de identidade. As cadeas públicas operan sen unha autoridade central, permitindo a calquera participante verificar e engadir transaccións de forma transparente. Calquera pode darse de alta, conectarse á rede e enviar transaccións (pode que a cambio dunha taxa). Estas redes considéranse xeralmente máis descentralizadas e ofrecen maior transparencia, pero tamén poden ser máis susceptibles a ataques ou outras actividades maliciosas debido á súa natureza aberta.

As cadeas públicas máis populares son as utilizadas para facer realidade as criptomoedas (ver capítulo 3). Unha cadea de bloques de criptomoedas é unha tecnoloxía de libro maior descentralizado e distribuído que se utiliza para rastrexar e rexistrar transaccións relativas a unha moeda dixital, como Bitcoin ou Ethereum. Permite transaccións entre pares sen necesidade de intermediarios, como bancos ou institucións financeiras, e está deseñada para ser segura, transparente e inmutable. As criptomoedas teñen unha gran popularidade entre investidores e o público en xeral como unha tecnoloxía que ten o potencial de revolucionar os sistemas financeiros tradicionais e transformar a forma en que realizamos transaccións económicas e intercambiamos valor.

Ao ser públicas, están deseñadas para ser transparentes e auditaes publicamente, o que significa que os detalles das transaccións son visibles para calquera. En canto á súa capacidade para crecer e soportar cada vez máis usuarios, as cadeas de bloques públicas poden xestionar un gran volume de transaccións, pero tamén poden experimentar problemas de escalabilidade debido a mecanismos de consenso que consomen moita enerxía, como a proba de traballo.

2.3.2 Cadeas privadas

Nunha rede blockchain privada, autorizada ou permissionada (*permissioned blockchain* en inglés), o acceso á rede e a participación na validación e verificación das transaccións están restrinxidos a un grupo específico de participantes autorizados. Estes participantes adoitan ser identificados e verificados mediante algún tipo de proceso de verificación de identidade. As blockchains privadas utilízanse habitualmente en aplicacións empresariais ou gobernamentais nas que a privacidade, a seguridade e o cumprimento da normativa son requisitos operativos fundamentais, xa que están deseñadas para facilitar o intercambio seguro e transparente de datos e a colaboración entre entidades entre as que existe unha relación de confianza.

Tamén é habitual que un conxunto de entidades se poñan de acordo para despregar unha cadea de bloques restrinxida aos membros de dito conxunto de entidades. É o que se coñece como unha cadea de bloques consorciada ou de consorcio. Trátase de cadeas de bloques que requiren autorización dunha entidade central para poder operar, e en moitos casos están deseñadas para dar soporte a aplicacións empresariais específicas.

Estas cadeas de bloques privadas ofrecen varias vantaxes ás entidades que as explotan, como unha maior transparencia, seguridade e eficiencia. Ao utilizar un rexistro de información compartido e distribuído, os participantes poden acceder aos datos e verificalos en tempo real, o que reduce o risco de erros, fraudes e disputas. Ademais, as cadeas de bloques empresariais contribúen a axilizar os procesos empresariais e a reducir custos, eliminando intermediarios, automatizando a transferencia de información e axilizando os procesos.

Entre os casos de uso comúns para as cadeas de bloques privadas, podemos identificar a xestión de cadeas de subministración, a verificación da identidade dixital, o financiamento do comercio e os pagos. Por exemplo, unha cadea de subministración baseada na tecnoloxía blockchain podería permitir

aos fabricantes, distribuidores e retalistas rastrexar o movemento de mercadorías desde a súa produción ata a entrega ao cliente, garantindo a transparencia e reducindo o risco de falsificación de produtos. Do mesmo xeito, unha cadea de bloques de identidade dixital podería permitir a verificación segura e eficiente da identidade de particulares e empresas, reducindo o risco de fraude e a usurpación de identidade. En definitiva, as cadeas de bloques empresariais ofrecen ás empresas unha forma de aproveitar as vantaxes da tecnoloxía de rexistro descentralizado, á vez que abordan os retos e requisitos de casos de uso específicos.

As cadeas de bloques empresariais poden ofrecer maior privacidade e confidencialidade, utilizando técnicas como o cifrado de datos ou o intercambio selectivo de información. Tamén adoitan a ser máis escalables (é máis doado engadir participantes ou bloques sen comprometer as prestacións), xa que poden deseñarse cunha aplicación concreta en mente e non necesitan soportar o mesmo volume de transaccións que unha cadea de bloques pública.

2.4 Mecanismos de consenso

Como apuntabamos no capítulo 1, os mecanismos de consenso son as regras e procesos utilizados para garantir que todos os nodos dunha rede de cadea de bloques están de acordo sobre o estado da blockchain nun momento dado. Estes mecanismos serven para afrontar as vulnerabilidades das redes blockchain como o problema dos xenerais bizantinos ou os ataques de Sybil ou do 51 % (ver apartado 2.9). Ademáis, o mecanismo de consenso elixido vai depender moito do tipo de blockchain. As cadeas públicas adoitan utilizar mecanismos de consenso que implican un investimento económico por parte dos axentes que participan en dito consenso, mentres que nas cadeas privadas os mecanismos baséanse máis na confianza existente entre os participantes na rede.

En consecuencia, na práctica utilízanse varios mecanismos de consenso, cada un coas súas propias vantaxes e inconvenientes. Os mecanismos de consenso máis utilizados son a proba de participación (*proof-of-stake*, PoS), a proba de traballo (*proof-of-work*, PoW), e a tolerancia a fallos bizantinos (*bizantine fault tolerance*, BFT), nas súas diferentes implementacións prácticas. A elección do mecanismo de consenso depende das necesidades e obxectivos específicos da rede blockchain (Wang et al., 2019). Nos parágrafos seguintes presentamos brevemente os máis utilizados, e introduciremos tamén a proba de queima (*proof-of-burn*, PoB) como un algoritmo orixinal a medio camino

entre a PoW e a PoS . Ademáis dos descritos a continuación, outros mecanismos de consenso que podemos atopar son a proba de autoridade (PoA, Manolache, Manolache y Tapus, 2022), proba de peso (PoWe, Bada et al., 2021) ou proba de importancia (Pol, Xiao et al., 2022).

2.4.1 Proba de traballo

Nunha blockchain de proba de traballo, os nodos compiten para resolver problemas matemáticos complexos utilizando potencia computacional. O primeiro nodo que resolve o problema é recompensado con criptomoeda e o dereito para crear o seguinte bloque da cadea. Bitcoin utiliza un mecanismo de consenso de proba de traballo (Nakamoto, 2008a). En moitos casos, como veremos no apartado 2.5, o desafío matemático consiste en xerar un resumo criptográfico ou hash do novo bloque con determinadas propiedades. Este mecanismo é típico das blockchains públicas, porque non require de coñecemento ou confianza previa entre os axentes que participan no consenso.

2.4.2 Proba de participación

Nunha blockchain de proba de participación, os nodos selecciónanse para crear novos bloques en función da cantidade de criptomoeda que ofrecen como aval. Os validadores selecciónanse aleatoriamente para crear novos bloques e, se se descobre que actúan de forma maliciosa, execútase o aval. Ethereum pasou de usar un mecanismo de proba de traballo a un de proba de participación en setembro de 2022 (Buterin y Schneider, 2022). O apéndice D describe o mecanismo de consenso actual de Ethereum.

A proba de participación delegada (DPoS) é unha variante do mecanismo PoS onde os posuidores de criptomoedas elixen a un grupo máis reducido de validadores encargados de crear novos bloques (Snider, Samani y Jain, 2018). A estes validadores págaselles polos seus servizos e poden ser cesados das súas funcións se se descobre que actúan en contra dos intereses da rede. As criptomoedas EOS e Lisk utilizan un mecanismo de consenso de proba de participación delegada (Skh Saad y Raja Mohd Radzi, 2020). A principal diferenza entre PoS e DPoS é a forma en que se seleccionan os validadores. Nunha blockchain PoS, os validadores selecciónanse en función da cantidade de criptomoeda que posúen e achegan como aval, mentres que nunha blockchain DPoS, os validadores son elixidos polos titulares de criptomoeda. Isto proporciona a estes últimos un control máis directo sobre o proceso de

consenso e permítelles responsabilizar aos validadores das súas accións. Outra diferenza é o número de validadores da rede. As redes PoS poden ter centos ou miles de validadores, mentres que as redes DPoS adoitan ter un número limitado de validadores. Isto pode facer que as redes DPoS sexan máis eficientes e rápidas que as redes PoS, xa que un menor número de validadores implica un consenso máis rápido.

2.4.3 Proba de queima

A proba de queima (Proof of Burn, PoB) é un mecanismo de consenso alternativo ou complementario á proba de participación utilizado nalgunhas criptomoedas e redes blockchain cuxa idea central é que os participantes *queiman* ou destrúen parte da súa criptomoeda enviándoa a unha dirección especial e irrecuperable, *sacrificio* que lles outorga dereitos para participar na rede, minar novos bloques ou validar transaccións (Karantias, Kiayias y Zindros, 2020). Pode verse como un híbrido entre a proba de traballo e a proba de participación, onde a queima de criptomoeda sería unha especie de minería virtual que simula o gasto en recursos da minería real da proba de traballo. Por outra banda, este mecanismo complementa algunhas versións da proba de participación, onde a queima de moedas substitúe ou complementa o depósito dun aval.

O proceso de queima implica que as criptomoedas enviadas a estas direccións elimínanse permanentemente da subministración circulante, o que reduce a posibilidade de inflación da moeda e aumenta o valor da criptomoeda restante. A cambio de queimar moedas, o usuario recibe certos beneficios dentro da rede, como permisos para xerar novos tokens ou participar na gobernanza do protocolo. Este mecanismo demostra o compromiso do usuario coa rede, xa que a queima de moedas representa un sacrificio real.

2.4.4 Tolerancia a fallos bizantinos

Nunha cadea de bloques tolerante a fallos bizantinos, os nodos colaboran para poñerse de acordo sobre o estado da cadea de bloques. Uns nodos encárganse de propoñer novos bloques, e outros nodos votan sobre a validez dos bloques propostos. Se unha maioría absoluta de nodos (alomenos o 51 %, podéndose requirir máis en función da súa implementación práctica) está de acordo coa validez dun bloque, se engade á cadea de bloques. BFT é un concepto xeral que describe a capacidade dun sistema distribuído para

funcionar correctamente e chegar a un consenso mesmo cando algúns nodos do sistema fallan ou se comportan de maneira maliciosa, é dicir, teñen un comportamento bizantino. O nome que se lle deu a este tipo de comportamento derívase do *problema dos xenerais bizantinos*, un problema teórico en ciencias da computación (Lamport, Shostak y Pease, 2019). Polo tanto, BFT é un concepto teórico que establece as condicións para a tolerancia a fallos bizantinos, e polo tanto non define un procedemento ou algoritmo concreto para lograr o consenso nestas situacións. De todos os xeitos, BFT serve como base teórica para moitos algoritmos de consenso en sistemas distribuídos.

Por exemplo, a tolerancia a fallos bizantinos práctica (ractical BFT, PBFT) é unha solución concreta da BFT onde os nodos que propoñen bloques e os que validan os bloques propostos vanse turnando (Castro y Liskov, 2002). Trátase dunha forma de facer que BFT sexa práctico e eficiente en sistemas reais con un protocolo de consenso concreto con pasos definidos que presenta unha alta eficiencia en condicións normais (cando non hai nodos maliciosos), podendo tolerar ata un terzo de nodos maliciosos ou defectuosos na rede. Quorum, unha plataforma de cadea de bloques baseada en Ethereum, utiliza este mecanismo de consenso (Baliga et al., 2018) e Hyperledger Fabric utiliza un mecanismo de consenso baseado nel (Barger et al., 2021).

A PBFT utiliza un proceso de tres fases, chamadas pre-preparación, preparación e compromiso, que garante que tendo $3n + 1$ nodos en total podemos tolerar n nodos defectuosos ou maliciosos. Noutras palabras, polo menos dous terzos dos nodos da rede teñen que ser honestos e poñerse de acordo na validez dun bloque para que se engada á cadea de bloques, puidendo tolerar un terzo de nodos deshonestos. Ademáis, PBFT utiliza un proceso de votación complexo que pode eliminar as mensaxes duplicadas e reducir a cantidade de datos que hai que transmitir entre nodos. PBFT é relativamente eficiente e económico, pero só funciona en redes privadas, xa que é necesario coñecer a identidade dos nodos.

2.5 Creación de novos bloques

O proceso de creación de novos bloques está intrinsecamente ligado ao mecanismo de consenso da blockchain, xa que este último determina como se selecciona e valida cada novo bloque na cadea. En esencia, o mecanismo de consenso decide quen ten o dereito de propoñer o seguinte bloque e como se verifica e acepta ese bloque polo resto da rede. Como vimos, na proba de traballo, os mineiros compiten para resolver un quebracabezas criptográfi-

co, e o primeiro en resolvelo gaña o dereito de crear o seguinte bloque. Pola súa banda, na proba de participación, os validadores selecciónanse baseándose na cantidade de criptomoedas que poñen como aval para propoñer e validar novos bloques. A PBFT conleva un proceso de votación entre nodos para acordar sobre o seguinte bloque. En todos os casos, o mecanismo de consenso asegura que a creación de novos bloques sexa un proceso seguro, descentralizado e resistente a manipulacións, mantendo así a integridade e a confiabilidade da cadea de bloques.

Nos parágrafos seguintes explicamos máis en detalle como se crean bloques no caso dos mecanismos de consenso que vimos de presentar.

2.5.1 Minería de bloques e proba de traballo

Nunha cadea de bloques de proba de traballo como Bitcoin, denomínase minería (*mining*) ao proceso de engadir definitivamente novos bloques á cadea de bloques a través dun proceso computacional que implica a resolución de problemas matemáticos complexos. Os mineiros compiten para resolver un enigma criptográfico utilizando potencia de cálculo para atopar unha clave hash específica para o novo bloque que cumpra un determinado requisito de dificultade. O primeiro mineiro que atopa a solución ao enigma e transmítea á rede recibe unha recompensa polo bloque así minado, que consiste xeralmente en criptomoeda de nova creación e/ou comisións por cada transacción validada. Esta estratexia protexe razoablemente á cadea de bloques fronte aos ataques, xa que resultaría demasiado custoso involucrar a un número suficiente de mineiros para ter éxito.

O proceso de minado desenvólvese en varias etapas:

1. Validación de transaccións: Os mineiros reciben novas transaccións da rede e válidas para asegurarse de que son lexítimas e cumpren as normas da rede.
2. Creación dun bloque: unha vez que un mineiro validou un conxunto de transaccións, agrúpaas nun bloque e engade unha transacción especial para cobrarse a recompensa por minar o bloque e abonar dita recompensa na carteira do mineiro.
3. Resolución do reto: o mineiro utiliza a súa potencia de cálculo para tentar atopar unha solución ao enigma criptográfico asociado ao bloque.

4. Difusión da solución: unha vez que un mineiro atopou unha solución válida, o bloque recentemente minado transmítese á rede, e outros nodos da rede verifican que efectivamente dita solución é correcta.
5. Adición do bloque á cadea de bloques: unha vez que se verifica que a solución é válida, o novo bloque se engade á cadea de bloques e o mineiro recibe a recompensa por minar ese bloque.

A dificultade do crebacabezas matemático axústase dinamicamente para garantir que se engaden novos bloques á cadea de bloques a un ritmo constante. O proceso de minería está deseñado para ser computacionalmente intensivo e consumir moitos recursos para evitar a fraude e garantir a seguridade e integridade da rede de cadeas de bloques.

A continuación ilustramos cun exemplo sinxelo o minado mediante proba de traballo. Imaxinemos que en determinada rede blockchain, para contribuír un novo bloque, esíxese como reto que a clave hash do mesmo comece cun determinado número de bits a cero. A función seguinte en JavaScript ilustra como se calcularía tal clave hash a partir dunha cadea de datos.

```
1 // A función findNonce devolve unha clave hash SHA-256
2 // para a cadea de caracteres 'payload' que cumpre a
3 // condición de ter os 'n' primeiros bits a cero.
4 //
5 // A función devolve un obxecto co nonce utilizado e
6 // o valor da clave de hash. O valor do nonce devolto
7 // é unha indicación da carga de traballo necesaria
8 // (a maior valor, maior carga de traballo).
9 //
10 // A función usa o módulo crypto para obter a clave hash.
11
12 const crypto = require('crypto');
13 function findNonce(payload, n) {
14   let nonce = 0;
15   while (true) {
16     // creamos un obxecto hash SHA-256 de crypto
17     const hash = crypto.createHash('sha256');
18     // concatenamos os datos de entrada co nonce
19     const data = payload + nonce;
20     // calculamos a clave hash
21     hash.update(data);
22     const hex = hash.digest('hex');
23     // comprobamos se cumprimos a condición
24     if (hex.slice(0, n / 4) === '0'.repeat(n / 4)) {
25       return { nonce, hash: hex };
26     }
27     nonce++;
28   }
29 }
```

 pow.js


```
26 |     }  
27 |     // o valor do nonce increméntase ata que se obtén  
28 |     // unha clave que cumpre a condición.  
29 |     nonce++;  
30 | }  
31 | }
```

Esta función engade á cadea de entrada un valor (nonce) e calcula a clave hash do conxunto. Vai probando con sucesivos nonces ata que consegue xerar unha clave co número de ceros requirido. Por exemplo, se lle pasamos a cadea **Datos para proba de traballo** con tres valores de *n* diferentes (8, 16 e 20) e medimos os tempos de execución en cada caso:

```
1 | // pow-test.js  
2 | // Extracto de código para probar a función anterior  
3 | //  
4 | const st2 = new Date().getTime();  
5 | const r2 = findNonce('Datos para proba de traballo', 8);  
6 | const end2 = new Date().getTime();  
7 | console.log('Nonce: ${r2.nonce}, Hash: ${r2.hash},  
8 |           Time: ${end2 - st2}ms');  
9 |  
10 | const st3 = new Date().getTime();  
11 | const r3 = findNonce('Datos para proba de traballo', 16);  
12 | const end3 = new Date().getTime();  
13 | console.log('Nonce: ${r3.nonce}, Hash: ${r3.hash},  
14 |           Time: ${end3 - st3}ms');  
15 |  
16 | const st4 = new Date().getTime();  
17 | const r4 = findNonce('Datos para proba de traballo', 20);  
18 | const end4 = new Date().getTime();  
19 | console.log('Nonce: ${r4.nonce}, Hash: ${r4.hash},  
20 |           Time: ${end4 - st4}ms');
```

Obtemos os resultados do cadro 2.1.

Aínda que o tempo necesario para calcular o hash depende da cadea de entrada, en media o tempo de cálculo, e por tanto o traballo necesario para resolver o reto, crecerá exponencialmente co número de bits iniciais a cero requiridos. Axustando o número de ceros modelamos a dificultade do reto. A función seguinte ilustra o minado dun bloque mediante este sistema.

```
1 | // create-node.js  
2 | // Función ilustrativa de minado dun bloque mediante proba de
```

Cadro 2.1. Carga de traballo para distintos valores de 'n' en 'findNonce'.

n	Nonce	Hash	Tempo
8	117	0x00a95eb3d2fe63b...	1 ms
16	24077	0x0000be76a2ce33d...	172 ms
20	185001	0x0000001825166dc6...	555 ms
24	16857739	0x00000009091214cb...	13.383 ms

```

3 // traballo.
4 //
5 // Esta función utiliza a función findNonce() dos exemplos
6 // anteriores para calcular o hash e o nonce do novo bloque.
7 // Toma catro parámetros:
8 // * transaction (a cadea que representa a transacción para
9 //   incluír no bloque),
10 // * prev_hash (o hash do bloque anterior),
11 // * height (a altura do bloque anterior),
12 // * diff (o parámetro n a pasar á función findNonce()
13 //   para atopar o hash deste bloque).
14 //
15 // A función concatena os catro parámetros de entrada nunha
16 // cadea e pásaa a findNonce() para atopar o nonce e
17 // o hash do novo bloque.
18 // O tempo necesario para atopar o nonce e o hash tamén
19 // se rexistra utilizando new Date().getTime(). A transacción
20 // resultante, prev_hash, height, hash, nonce, diff, o tempo
21 // requirido para atopar o nonce e o hash almacénanse nun
22 // obxecto blockNode e devólvese.
23 //
24 const crypto = require('crypto');
25
26 function createBlockNode(transaction, prev_hash, height, diff) {
27   // momento do minado
28   const start = new Date().getTime();
29   // incrementamos a altura en 1.
30   height++;
31   // calculamos o hash coa dificultade pasada como parámetro
32   const { nonce, hash } = findNonce(transaction + prev_hash +
33     height + diff, diff);
34   // momento de finalización do minado.
35   const end = new Date().getTime();
36   // tempo necesario para o minado en milisegundos
37   const time = end - start;
38   // construímos o novo bloque
39   const blockNode = {

```

```
40     height,  
41     nonce,  
42     diff,  
43     time,  
44     prev_hash,  
45     hash,  
46     transaction  
47 };  
48 // devolvemos 0 bloque construído  
49 return blockNode;  
50 }
```

O *hashrate* ou taxa de hash é unha métrica utilizada no ámbito das cadeas de bloques que se refire á potencia computacional total que os mineiros dunha rede blockchain están a utilizar para resolver os algoritmos criptográficos que xeran novos bloques. En termos máis sinxelos, o *hashrate* mide a velocidade á que se están realizando cálculos de hash dentro dunha rede blockchain co obxectivo de validar transaccións e asegurar a rede.

Canto maior sexa o *hashrate* dun mineiro ou dunha rede completa, maior será a capacidade de procesar máis hashes por segundo, o que aumenta as posibilidades de atopar a solución correcta e agregar o seguinte bloque á cadea. O *hashrate* adoita expresarse en hashes por segundo (H/s), e pódense utilizar múltiplos como kilohashes (KH/s), megahashes (MH/s) ou gigahashes (GH/s) por segundo para expresar o rendemento total da rede.

O *hashrate* é un indicador importante para a seguridade da rede, xa que canto maior é o *hashrate*, máis segura é a rede, xa que é máis difícil para os atacantes lanzar ataques como o do 51 %. Tamén é un indicador da dificultade de minería, porque un *hashrate* máis alto a miúdo conduce a un aumento na mesma, o que axusta o tempo que toma xerar un novo bloque. Finalmente, o *hashrate* tamén inflúe na rendibilidade dos mineiros, xa que un maior *hashrate* individual pode traducirse nunha maior probabilidade de resolver un bloque e obter recompensas.

2.5.2 Forxa de bloques e proba de participación

Nunha blockchain de proba de participación, o proceso de creación de novos bloques é diferente da minería, e a maioría das veces denomínase simplemente creación de bloques aínda que tamén podemos atopar expresións como *forging* (forxa) ou *minting* (acuñado). A diferenza da proba de traballo, que se basea na potencia computacional e a resolución de quebra cabezas

criptográficos, a proba de participación baséase nun conxunto diferente de regras para lograr o consenso e validar as transaccións. Neste caso, os participantes que posúen unha certa cantidade da criptomoeda nativa da cadea de bloques poden converterse en validadores. Os validadores son responsables de verificar as transaccións e engadilas á cadea de bloques, e selecciónanse en función da súa participación, entendida como a cantidade de criptomoeda que depositan como aval ou a cantidade de criptomoeda que posúen. Cantos máis criptomoeda posúa ou achega un validador, maiores serán as súas posibilidades de ser seleccionado para construír o seguinte bloque.

O proceso de forxado nunha blockchain de proba de participación adoita implicar os seguintes pasos:

1. Depósito do aval: os validadores colocan unha certa cantidade de criptomoeda como aval, que serve como garantía e incentiva o bo comportamento. Se se descobre que un validador actúa de forma malintencionada, o seu aval será executado como penalización.
2. Selección: selecciónanse forxadores para crear un novo bloque en función da súa participación, e convértense en responsables de verificar as transaccións e crear un bloque.
3. Forxado do bloque: os validadores seleccionados crean un novo bloque e transmíteno á rede para a súa verificación.
4. Verificación: outros validadores da rede verifican o bloque e engádeno á cadea de bloques se é válido.

Do mesmo xeito que no caso anterior, os validadores que engaden con éxito un bloque á cadea de bloques son recompensados con comisións por transacción e criptomoeda recentemente acuñada.

A proba de participación considérase unha alternativa máis eficiente energeticamente que a proba de traballo, xa que non require o mesmo nivel de potencia de cálculo e consumo eléctrico. Tamén se considera máis respectuosa co medio ambiente, xa que non require hardware especializado como circuitos integrados específicos ou procesadores gráficos. Por outra banda, o proceso de selección de validadores evita na maioría dos casos a creación de bifurcacións na cadea de bloques. De todos os xeitos, a proba de participación non está exenta de dificultades, como o risco de centralización e o reto de garantir que os validadores actúen en interese da rede e non no seu propio.

2.5.3 Sacrificio e proba de queima

A proba de queima é un mecanismo interesante que tenta combinar algunhas das vantaxes da PoW e da PoS, aínda que a súa adopción é máis experimental e limitada en comparación con estes mecanismos de consenso. Na creación de bloques mediante PoB, os nodos *queiman* moedas enviándoas a unha dirección inaccesible. Isto dálles o dereito para competir para engadir novos bloques á cadea. Cada queima rexístrase na blockchain, creando un historial verificable, e a probabilidade de seren elixido para crear o seguinte bloque é proporcional á cantidade de moedas queimadas, tendo en conta tamén cando se realizaron as queimas. En consecuencia, os participantes compiten baseándose no seu poder de queima acumulado, poder que diminúe gradualmente co tempo, incentivando queimas continuas.

O participante seleccionado crea un novo bloque con transaccións pendentes e inclúe unha referencia ao seu historial de queimas como proba do seu dereito para propoñer novos bloques. Os outros nodos da rede verifican que o propoñente ten o dereito lexítimo baseado no seu historial de queimas e comprobán que as transaccións no bloque son válidas. Se todo é correcto, o bloque engádese á cadea e o creador do bloque recibe unha recompensa, xeralmente en forma de novas moedas e/o taxas de transacción. Finalmente, a rede actualiza o estado global, incluíndo os novos saldos e o poder de queima dos participantes.

2.5.4 Compromiso e tolerancia a fallos bizantina

Nunha cadea de bloques con tolerancia a fallos bizantinos, e en concreto PBFT, os nodos traballan xuntos para acordar o estado da rede e crear novos bloques. O proceso de creación de novos bloques nunha cadea de bloques PBFT consta de varios pasos:

1. Selección de nodos: nunha blockchain PBFT, selecciónase un subconxunto de nodos para propoñer novos bloques á rede. O número de nodos seleccionados e o proceso de selección depende da implementación específica da variante da PBFT e o seu algoritmo de consenso, pero adoita estar entre unhas ducias e uns centos.
2. Proposta: os nodos seleccionados forxan un novo bloque difundíndoo á rede. O bloque contén unha lista de transaccións que o nodo propoñente considerou válidas.

3. Validación: unha vez proposto un bloque, os demais nodos da rede válidano. Os nodos realizan unha serie de comprobacións criptográficas para asegurarse de que o bloque é válido e de que as transaccións que contén tamén o son. Se o bloque non é válido, rexéitase e comeza de novo o proceso de creación dun novo bloque.
4. Precompromiso: se a rede considera válido un bloque, os nodos entran nunha fase de precompromiso. Durante esta fase, os nodos sinalan a súa intención de engadir ou non o bloque á cadea de bloques emitindo unha mensaxe á rede.
5. Compromiso: unha vez que unha supermayoría de nodos (normalmente dous terzos ou máis) manifestou a súa intención de engadir o bloque á cadea de bloques, o bloque se engade á cadea de bloques e os nodos pasan ao seguinte bloque.

O código seguinte ilustra a evolución dun algoritmo BFT sinxelo con tres nodos. A función comeza xerando unha mensaxe inicial aleatoria para cada nodo, para indicar se ese nodo acepta o novo bloque. Esa decisión pode ser honesta (o bloque é realmente válido ou inválido, e o nodo di a verdade) ou deshonesta (o nodo pretende enganar ao resto dos nodos). A continuación, cada nodo envía a súa mensaxe aos demais nodos. Cada nodo recibe as mensaxes dos outros nodos e decide un veredicto final baseado nas mensaxes recibidas utilizando a función de axuda **decideMensaxe**. Por último, a función comproba se todos os nodos acordaron o mesmo mensaxe e mostra o resultado na consola.

```
1 //
2 // Esta función toma tres argumentos que representan os
3 // tres nodos do sistema. Suponse que cada nodo ten
4 // un método de envío con dous argumentos (as dúas mensaxes
5 // a enviar aos outros dous nodos) e un método de recepción
6 // que recibe as mensaxes dos outros nodos.
7 //
8 // Supoñemos que os nodos que participarán no consenso foron
9 // seleccionados previamente (nodo1, nodo2 e nodo3) e
10 // que os nodos forxadores propuxeron un novo bloque.
11 //
12 function byzantineFaultTolerant(nodo1, nodo2, nodo3) {
13     // Comezo da fase de Validación. Neste exemplo
14     // cada nodo valida ou non o bloque aleatoriamente
15     // cunha mensaxe inicial (0 ou 1).
16     var nodo1Mensaje = Math.round(Math.random());
```

 bft.js

```
17     var nodo2Mensaje = Math.round(Math.random());
18     var nodo3Mensaje = Math.round(Math.random());
19
20     // e difunde o seu veredicto á rede, é dicir,
21     // cada nodo envía a súa mensaxe aos demais nodos.
22     nodo1.send(nodo2Mensaje, nodo3Mensaje);
23     nodo2.send(nodo1Mensaje, nodo3Mensaje);
24     nodo3.send(nodo1Mensaje, nodo2Mensaje);
25
26     // Cada nodo recibe as mensaxes dos outros nodos.
27     var nodo1MensajesRecibidos = [nodo2Mensaje, nodo3Mensaje]
28     var nodo2MensajesRecibidos = [nodo1Mensaje, nodo3Mensaje]
29     var nodo3MensajesRecibidos = [nodo1Mensaje, nodo2Mensaje];
30
31     // Fase de Precompromiso. Cada nodo expresa a súa intención
32     // de engadir ou non o novo bloque, en función das
33     // mensaxes recibidas.
34     var nodo1MensajeFinal = decideMensaje(nodo1MensajesRecibidos);
35     var nodo2MensajeFinal = decideMensaje(nodo2MensajesRecibidos);
36     var nodo3MensajeFinal = decideMensaje(nodo3MensajesRecibidos);
37
38     // Fase de Compromiso. Se todos os nodos acordan engadir o
39     // bloque (acordo de mensaxe 1), o nodo engádese á rede.
40     // Comprobamos se todos os nodos acordaron o mesmo
41     // mensaxe e que a mensaxe é 1.
42     if (nodo1MensajeFinal === nodo2MensajeFinal
43         && nodo1MensajeFinal === nodo3MensajeFinal) {
44         if (nodo1MensajeFinal == 1) {
45             console.log("Bloque correcto. Engádase");
46         } else {
47             console.log("Bloque incorrecto. Descártese");
48         }
49     } else {
50         console.log("Bloque incorrecto. Descártese");
51     }
52 }
53
54 // Fase de Validación.
55 // Función para decidir unha mensaxe final en base
56 // ás mensaxes recibidas.
57 function decideMensaje(mensajesRecibidos) {
58     // Cuenta el número de mensajes 0 y 1
59     var count0 = 0
60     var count1 = 0;
61     for (var mensaje of mensajesrecibidos) {
62         if (mensaje === 0) {
63             count0++;
64         } else if (mensaje === 1) {
65             count1++;
66         }
67     }
68     return count1 > count0 ? 1 : 0;
69 }
```

```
66     }  
67   }  
68  
69   // Se hai máis mensaxes 0 que 1, decidímonos por 0;  
70   // En caso contrario, decidímonos por 1  
71   if (count0 > count1) {  
72     return 0;  
73   } else {  
74     return 1;  
75   }  
76 }
```

Trátase dunha versión simplificada da PBFT que non ten en conta todos os escenarios de fallo posibles. Na práctica, o algoritmo é bastante máis complexo para garantir a tolerancia a fallos e a seguridade. Ademais, neste exemplo sinxelo consideramos que a maioría para aceptar o novo bloque alcázase por consenso de todos os nodos (máis esixente que a maioría de 51 % do modelo xeral ou mesmo da maioría de dous terzos comentada antes).

2.6 As bifurcacións nas cadeas de bloques

Unha bifurcación ou *fork* no contexto das cadeas de bloques é unha división dunha cadea en dous ou máis cadeas que, a partires do momento da división, toman camiños diferentes. Típicamente, isto ocorre cando un cambio nas regras de consenso ou no protocolo dunha criptomoeda provoca a creación dunha nova versión da cadea de bloques orixinal. Os forks poden clasificarse principalmente en dous tipos:

- **Hard Forks (Bifurcacións duras).** Ocorren por un cambio radical no protocolo da cadea de bloques, que pasa a non ser compatible coa versión anterior. Isto significa que os nodos que non actualicen o seu software para seguir as novas regras do protocolo non poderán validar novos bloques, o que na práctica termina na creación dunha cadea separada. Un exemplo famoso de hard fork é o que ocorreu con Bitcoin e a creación de Bitcoin Cash en 2017 (Kwon et al., 2019).
- **Soft Forks (Bifurcacións brandas).** A diferenza dos hard forks, un soft fork é un cambio no protocolo que é compatible cara atrás. Isto significa que os nodos que non actualicen o seu software aínda poden validar e participar na nova cadea de bloques, pero con algunhas limitacións ou restricións. Os nodos actualizados seguirán as novas regras, pero

os non actualizados poderán seguir operando dentro do sistema sen romper a integridade da rede.

Os forks prodúcense a miúdo como resultado de debates sobre melloras de protocolo, disputas entre comunidades ou actualizacións necesarias para escalar ou mellorar a rede. Aparte do famoso Bitcoin Cash fork de 2017, debido a diferenzas na comunidade de Bitcoin sobre como escalar a rede, outros fork famosos foron o da rede Ethereum en 2016 (Kiffer, Levin y Mislove, 2017), o de Bitcoin SV en 2018 (Bouraga, 2022) e o de Ethereum 2.0 (Ahn, Yi y Kim, 2024). En 2016, despois dun hackeo importante ao proxecto DAO, a comunidade de Ethereum votou para implementar un hard fork que reverteu as transaccións involucradas no hackeo. Os que non estiveron de acordo con esta decisión continuaron usando a cadea orixinal, que pasou a chamarse Ethereum Classic. Aínda que non é un fork clásico, Ethereum foi implementando cambios importantes na súa rede cara a Ethereum 2.0, que incluíu a transición dun mecanismo de consenso de proba de traballo a proba de participación. A reforma completouse en 2023.

2.7 O reto da escalabilidade

A escalabilidade é un dos maiores desafíos que enfrenta a tecnoloxía blockchain na súa adopción xeneralizada (Sanka y Cheung, 2021). A medida que aumenta o uso de redes descentralizadas, as cadeas de bloques, como Ethereum e Bitcoin, enfrentan problemas de conxestión e lentitude nas transaccións debido ao seu deseño (en moitos casos cada nodo da rede ten unha copia completa da cadea de bloques que hai que manter actualizada mediante mecanismos de consenso) e ao límite de procesamento de transaccións por bloque. As cadeas laterais ou *sidechains* e as cadeas de capa 2 ou *layer 2 blockchains* xurdiron como enfoques innovadores para abordar este reto. As sidechains son cadeas independentes que se conectan a unha blockchain principal, permitindo maior flexibilidade e capacidade de procesamento sen sobrecargar a cadea principal. Doutra banda, as blockchains de capa 2 constrúense sobre a blockchain principal para procesar transaccións fóra dela e logo consolidar os resultados na cadea base, mellorando significativamente a velocidade e reducindo custos.

Unha blockchain de capa 2 é unha solución deseñada para mellorar a escalabilidade e eficiencia dunha blockchain principal (coñecida como capa 1, como Bitcoin ou Ethereum) ao procesar transaccións fóra da cadea principal,

pero mantendo a seguridade e a descentralización da capa 1 ou capa base. As blockchains de capa 2 permiten realizar transaccións máis rápidas e económicas ao descargar a carga de procesamento da blockchain principal. Estas solucións impleméntanse sobre a blockchain orixinal, e logo consolídanse ou *resumen* as transaccións nun só bloque que se rexistra na blockchain principal, o que reduce a conxestión e os altos custos das transaccións. Exemplos de solucións de layer 2 inclúen Lightning Network en Bitcoin (Divakaruni y Zimmerman, 2023) e Optimistic Rollups ou zk-Rollups en Ethereum (Thibault, Sarry y Hafid, 2022).

O propósito final dunha blockchain de capa 2 é mellorar a experiencia dos usuarios na rede principal, xa que aborda problemas de escalabilidade, tempo de confirmación e custos de transacción, sen comprometer a seguridade inherente á blockchain de capa base.

Por outra banda, unha sidechain é unha cadea de bloques que corre en paralelo a unha blockchain principal, coa capacidade de interactuar con ela mediante un protocolo de interoperabilidade. O obxectivo destas cadeas laterais é mellorar a escalabilidade e a funcionalidade das blockchains principais, permitindo que certas transaccións ou procesos se realicen de forma independente da rede principal.

As sidechains permiten que os activos ou tokens se movan entre a blockchain principal e a sidechain sen comprometer a seguridade ou a integridade da rede. Isto é posible grazas a anclaxes ou *two-way pegs*, que son sistemas que aseguran a correcta transferencia de activos entre as dúas redes.

Estas cadeas laterais permiten a experimentación con novas características e melloras, como a introducción de novos mecanismos de consenso, sen poñer en risco a blockchain principal. Un exemplo de sidechain é Liquid, que está vinculada á rede de Bitcoin e se usa para mellorar a velocidade e eficiencia das transaccións (Nick, Poelstra y Sanders, 2020).

A principal diferenza entre unha sidechain e unha blockchain de capa 2 radica na forma en que interactúan coa blockchain principal e nos seus obxectivos. Unha sidechain é unha blockchain independente que está conectada á blockchain principal (layer 1) a través dunha relación de interoperabilidade. Funciona de forma autónoma e ten o seu propio mecanismo de consenso, o que lle permite ter características e regras diferentes ás da blockchain principal. Para transferir activos entre a blockchain principal e a sidechain, utilízanse pontes ou anclaxes que bloquean e liberan tokens entre ambas as cadeas. As sidechains non dependen directamente da seguridade da blockchain principal,

senón que confían no seu propio sistema de seguridade. Por outra banda, as solucións de capa 2 constrúense sobre a blockchain principal para mellorar a súa escalabilidade, reducindo a carga de procesamento na cadea basee. A diferenza das sidechains, as blockchains de capa 2 dependen directamente da seguridade e o consenso da cadea principal, o que significa que as transaccións nesta segunda capa resúmense ou agrúpanse, e logo verifícanse na primeira capa da blockchain ou blockchain base para garantir a súa integridade.

2.8 Vantaxes e inconvenientes das cadeas de bloques

As principais vantaxes da tecnoloxía blockchain nacen das súas características distintivas, é dicir, da súa natureza descentralizada, a súa inmutabilidade, a súa transparencia, as súas características de seguridade e a súa eficiencia computacional.

A natureza descentralizada da cadea de bloques significa que non necesita dunha autoridade central que a controle ou a xestione, o que a fai especialmente resistente aos ataques informáticos e á manipulación. Mesmo no caso de redes privadas, onde unha autoridade central autoriza a determinados perfís de usuario a interactuar coa rede, a súa naturaliza descentralizada faina resistente tanto a ataques internos como externos porque non existe un único punto de fallo ou risco de ataque.

A inmutabilidade das cadeas de bloques implica que, unha vez que un bloque se engade á cadea de bloques, non pode alterarse nin borrarse sen o consenso de toda a rede. No caso de que se decida modificar ou eliminar un bloque ou unha transacción dentro dun bloque, a acción materialízase mediante unha nova transacción ou un novo bloque que declara inválida a transacción orixinal ou materializa a versión modificada da transacción orixinal. Noutras palabras, os cambios na rede vanse apuntando nun libro maior de maneira incremental. As entradas orixinais non se modifican. Isto garante a integridade, a rastrexabilidade e a inmutabilidade dos datos almacenados na cadea de bloques.

Como consecuencia do anterior, dado que as cadeas de bloques son descentralizadas e inmutables, non necesitan dun actor que achegue confianza, o que significa que os participantes poden fiarse dos datos almacenados na cadea de bloques sen depender dun intermediario de confianza. Todas as transaccións na cadea de bloques son transparentes e poden rastrexarse ata a súa orixe, ben polo público en xeral nas redes públicas, ben polas entidades

autorizadas a interaccionar no caso das redes privadas (usuarios, reguladores, aplicacións, oráculos, etc.). Isto proporciona un nivel de transparencia e unha capacidade de rendición de contas sen precedentes. En canto á seguridade, toda blockchain utiliza técnicas criptográficas avanzadas para asegurar a rede e garantir a privacidade e seguridade dos datos almacenados na mesma.

En canto á eficiencia operativa, a tecnoloxía blockchain, ao eliminarse os intermediarios e os puntos singulares de control, facilitanse transaccións máis rápidas e eficientes. Ademais, a tecnoloxía de cadeas de bloques pode reducir significativamente os custos operativos ao eliminar a necesidade de intermediarios e reducir as comisións por transacción. Finalmente, coa aparición dos contratos intelixentes tratados no capítulo 4, dan soporte a transaccións automatizadas e poden utilizarse para unha ampla gama de aplicacións, aumentando a eficiencia e reducindo a posibilidade de erros ou fraudes.

De todos os xeitos, tamén podemos identificar algúns inconvenientes das redes de cadeas de bloques. Por exemplo, a tecnoloxía actual de blockchain pode ser lenta e ter unha escalabilidade limitada, o que dificulta a xestión dun gran número de transaccións á vez. Isto pode dar lugar a tempos de transacción máis pausados e comisións máis elevadas, a pesar dos aforros obtidos coa eliminación dos intermediarios e a simplificación de cada transacción. Como vimos no apartado anterior, unha maneira de abordar esta limitación é mediante cadeas de capa 2 e as cadeas laterais.

Aínda que as redes de cadeas de bloques supoñen en xeral unha mellora na eficiencia operativa, poden non selo tanto en canto a eficiencia enerxética. No caso das redes baseadas en proba de traballo, o proceso de minado de bloques pode chegar a requirir unha cantidade significativa de electricidade, o que resulta nun alto consumo de enerxía e o rexeitamento social debido ao seu posible impacto ambiental (Islam et al., 2023).

As redes de cadeas de bloques tamén expoñen importantes desafíos regulatorios. A natureza descentralizada e anónima das blockchains públicas, habituais no campo das criptomoedas, pode dificultar aos reguladores a supervisión das actividades e aplicación das leis. Esta situación faise máis patente aínda debido á actual falta de estandarización na industria de blockchain, con diferentes plataformas que utilizan diferentes protocolos, o que dificulta a interoperabilidade entre diferentes blockchains, complica o desenvolvemento de aplicacións e entorpece a monitorización e vixilancia por parte do regulador.

Outro reto reside no cumprimento dos requisitos das normas de protección da información persoal, como a normativa europea de protección de datos (é

dicir, o Regulamento Xeral de Protección de Datos, RGPD). Dereitos como o dereito de oposición ao tratamento de datos persoais ou o dereito ao esquecemento no mundo dixital son difíciles de conseguir cando se almacenan datos persoais nunha blockchain, debido ás súas características de inmutabilidade (Delgado-von-Eitzen, Anido-Rifón y Fernández-Iglesias, 2021a).

A falta de estandarización xunto coa complexidade da tecnoloxía blockchain, pode dificultar a súa comprensión e uso efectivo por parte de usuarios non técnicos, o que limita a súa potencial adopción e aumenta os riscos en canto a seguridade. Aínda que a cadea de bloques considérase segura, sempre existe o risco de que se produzan ataques e fallos de seguridade debidos a vulnerabilidades ou erros de programación, sobre todo nas cadeas de bloques máis novas ou menos consolidadas, e as cadeas de bloques privadas. Dedicamos o apartado seguinte a comentar os aspectos relativos á seguridade máis importantes.

Finalmente, a operación de todas as cadeas de bloques, dado a súa natureza descentralizada, dependen dunha boa conectividade a Internet, o que pode supoñer un reto en zonas con infraestruturas deficientes ou conectividade limitada.

2.9 Aspectos de seguridade

A seguridade dunha rede blockchain depende de diversos factores, por exemplo o mecanismo de consenso utilizado, o tamaño e a distribución dos nodos da rede, os algoritmos criptográficos utilizados e o comportamento dos participantes. Se unha rede blockchain utiliza un mecanismo de consenso seguro e descentralizado como os do apartado 2.4, e conta cunha rede de nodos ampla e diversa, pode considerarse relativamente segura. Se os algoritmos criptográficos utilizados seguen os estándares habituais e os participantes se comportan honestamente, a seguridade da rede refórzase aínda mais.

Con todo, ningún sistema é completamente inmune aos ataques, e as redes blockchain non son unha excepción. Por exemplo, un atacante decidido e con recursos suficientes podería, polo menos en teoría, lanzar un ataque do 51 % a unha rede baseada en proba de traballo ou un ataque de longo alcance a unha rede baseada en proba de participación, entre outros ataques. De feito, hai varios tipos de ataques que poden dirixirse a unha rede blockchain. A continuación citamos os máis habituais.

O ataque dos nodos bizantinos ocorre cando un nodo malicioso (chamado nodo bizantino) pode tentar interromper o proceso de consenso difundindo un bloque diferente ou retendo bloques. Isto pode dar lugar a bifurcacións na cadea de bloques, onde diferentes nodos teñen diferentes puntos de vista do estado da cadea de bloques. O nome provén do problema dos xenerais bizantinos formulado por Leslie Lamport, Robert Shostak e Marshall Pease (1982) do que xa falamos antes. O problema consiste en que un grupo de xenerais bizantinos acampados nos arredores dunha cidade, cada un ao mando dunha parte do exército, deben coordinar o seu ataque ou retirada sobre a cidade, pero só poden comunicarse entre si enviándose mensaxes a través de mensaxeiros. Algúns dos xenerais son traidores e poden enviar mensaxes contraditorias ou falsas para confundir aos xenerais leais. Os xenerais leais deben chegar a un consenso sobre se atacar ou retirarse, mesmo en presenza de traidores. Este problema ten importantes implicacións para os sistemas distribuídos e as redes informáticas, onde as mensaxes poden perderse ou corromperse, e os nodos poden fallar ou comportarse maliciosamente. Unha solución a este problema no contexto de blockchain son os protocolos tolerantes a fallos bizantinos como o descrito PBFT.

O comentado ataque do 51 % prodúcese cando unha única entidade ou grupo de entidades controla o 51 % ou máis da potencia de cálculo da rede de proba de traballo, o que lles permite manipular a rede creando transaccións fraudulentas, realizando un dobre gasto ou excluindo a outros mineiros da rede (Shanaev et al., 2019).

Un ataque de longo alcance é un tipo de ataque que se basea en crear unha nova blockchain a partir do bloque xénese, utilizando un histórico alternativo diferente do da rede orixinal (Deirmentzoglou, Papakyriakopoulos y Patsakis, 2019). Este tipo de ataque é posible porque a seguridade de moitas redes blockchain, especialmente as redes de proba de participación, baséase na suposición de que unha gran proporción dos validadores teñen un historial longo e continuo de participación na rede. Se un atacante pode obter unha cantidade significativa de participacións ou controlar un gran número de nodos da rede, pode crear potencialmente unha nova cadea de bloques máis longa que a orixinal e converterse así na cadea dominante.

O ataque de Sybil prodúcese cando un único usuario crea múltiples identidades ou nodos na rede, o que lle permitiría controlar unha parte significativa dos nodos avalistas ou da potencia de cálculo da rede e manipular as transaccións. Este tipo de ataque é posible porque moitas redes blockchain, especialmente as redes públicas que se basean na proba de traballo ou na proba de partici-

pación, non esixen ningún procedemento de verificación de identidade ou de coñecemento do cliente para unirse á rede. Un atacante pode crear múltiples identidades ou nodos falsos e utilízaos para obter unha influencia desproporcionada sobre a rede, por exemplo controlando o proceso de consenso ou censurando transaccións. O nome deste ataque está inspirado nun libro de 1973 titulado *Sybil*, unha muller á que se diagnostica un trastorno de identidade dissociativo (Schreiber, 1973).

O ataque de eclipsado consiste en controlar as comunicacións dun nodo vítima, de modo que só reciba datos dun nodo malicioso, illándoo de feito do resto da rede (Yves-Christian et al., 2018). Os ataques de eclipsado son posibles nas redes blockchain porque estas redes baséanse nun protocolo de comunicación peer-to-peer (P2P) para propagar información e transaccións entre nodos. O protocolo P2P permite a calquera nodo conectarse a calquera outro nodo da rede, e non existen mecanismos integrados para verificar a identidade ou reputación dos nodos conectados. Ademais, moitas redes blockchain, especialmente as que utilizan o consenso de proba de traballo, dependen dun gran número de conexións peer-to-peer para validar as transaccións e lograr o consenso. Isto faíñas vulnerables aos ataques de eclipsado, xa que un atacante pode controlar unha pequena parte dos nodos da rede e manipular o fluxo de información, dificultando que os nodos obxectivo reciban información correcta e participen no proceso de consenso.

Os atacantes tamén poden aproveitar vulnerabilidades ou erros de programación nos contratos intelixentes. Como veremos no capítulo 4, os contratos intelixentes son programas almacenados na blockchain que poden executarse automaticamente cando se cumpren determinadas condicións. Se un contrato intelixente contén vulnerabilidades, os atacantes poden aproveitálas para executar código malicioso.

O ataque de secuestro de marca de tempo ou *timejacking* consiste en manipular a marca de tempo dun bloque, o que permitiría a un atacante manipular a orde das transaccións ou crear transaccións fraudulentas (Ma, Ge y Zhou, 2020).

Do mesmo xeito que en calquera sistema distribuído, os ataques de denegación de servizo (DOS) perseguen sobrecargar a rede cun gran número de transaccións non válidas, dificultando ou imposibilitando o procesamento das transaccións lexítimas (Carl et al., 2006). Outro ataque xenérico contra sistemas distribuídos é o ataque de encamiñamento, que se produce cando un atacante manipula as táboas de encamiñamento da rede para redirixir o tráfico

a un nodo malicioso, o que en caso de blockchain facilitaría a manipulación de transaccións ou a substracción de criptomoedas (Aggarwal y Kumar, 2021).

Para mitigar estes ataques, as redes blockchain utilizan diversas medidas de seguridade, como os algoritmos de consenso descritos, a criptografía e mecanismos de supervisión da rede. Ademais, as actualizacións periódicas de software, as auditorías de código e as probas de penetración poden axudar a identificar e abordar as vulnerabilidades do código da blockchain ou dos contratos intelixentes. En definitiva, é fundamental supervisar e mellorar continuamente a seguridade dunha rede blockchain, a través de medidas como as auditorías periódicas, o establecemento de recompensas por erros e vulnerabilidades atopadas e mediante actualizacións frecuentes do software e os protocolos utilizados.

Capítulo 3

Criptomoedas

As criptomoedas son probablemente un dos elementos máis destacados e transformadores da tecnoloxía de cadeas de bloques. Desde a aparición do Bitcoin (Nakamoto, 2008b) en 2009 como a primeira criptomoeda descentralizada, este innovador medio de intercambio capturou a atención do mundo enteiro e revolucionou a forma en que concibimos o diñeiro, as transaccións financeiras e a economía dixital en xeral. A tecnoloxía de cadeas de bloques permitiu a creación dun ecosistema financeiro descentralizado e global, onde as fronteiras tradicionais deixan de existir e a innovación acelérase. Neste capítulo, exploraremos o mundo das criptomoedas, desde os seus fundamentos conceptuais ata as súas aplicacións máis disruptivas, pasando polos desafíos e oportunidades que presentan para o futuro das finanzas e mesmo das tecnoloxías dixitais.

3.1 Definición e historia

Podemos definir as criptomoedas como un tipo de activo dixital que utiliza técnicas criptográficas para asegurar as súas transaccións, controlar a creación de novas unidades e verificar a súa transferencia dun lugar a outro. A diferenza do diñeiro tradicional, respaldado por gobernos e controlado por bancos centrais e outras entidades financeiras, as criptomoedas operan de maneira descentralizada utilizando a tecnoloxía de cadea de bloques. Esta tecnoloxía permite rexistrar todas as transaccións de maneira transparente e segura, eliminando a necesidade dos intermediarios do sistema financeiro tradicional. As criptomoedas pódense utilizar como calquera outra moeda, é dicir, para pagar por bens e servizos, como investimento, ou como medio de intercambio.

A historia das criptomoedas comezou en 2009 coa creación de Bitcoin por unha persoa ou grupo de persoas baixo o pseudónimo de Satoshi Nakamoto. Bitcoin é o primeiro desenvolvemento funcional dunha criptomoeda, utilizando unha rede peer-to-peer e a tecnoloxía de blockchain definir as características de dita criptomoeda (cómo se crea, canta pode chegar a haber, etc.) e para rexistrar todas as transaccións de maneira segura e descentralizada. O concepto de criptomoeda fora proposto anteriormente por varios investigadores e entusiastas da criptografía, pero Bitcoin foi o primeiro sistema que resolveu o problema do dobre gasto das moedas dixitais, sen necesidade de recorrer a unha autoridade central que levase conta das transaccións.

O problema do dobre gasto é unha vulnerabilidade asociada ás moedas dixitais en xeral. Unha moeda dixital é simplemente un conxunto de uns e ceros almacenado nalgún lugar, algo que non ten existencia física. A diferenza das moedas ou billetes físicos, en principio poderíase facer calquera número de copias dunha *moeda* ou *billete* dixital e utilizarse máis dunha vez, comprometendo así a integridade do sistema. A diferenza das transaccións físicas nas que os billetes ou moedas non poden ser replicados e usados simultaneamente en transaccións diferentes, as moedas dixitais, ao non ser máis que conxuntos de bits, poderíanse replicar e gastar múltiples veces.

Nun sistema centralizado tradicional, o dobre gasto do diñeiro dixital evítase porque existe un organismo de control (p. ex., un banco central) que leva conta do diñeiro en circulación, e un conxunto de entidades (p. ex., os bancos) que levan conta de quen ten que diñeiro. Se transferimos diñeiro físico o problema do dobre gasto non é relevante porque desfacémonos del (e por tanto non podemos volver utilizalo) para darllo a outra persoa. No caso do diñeiro dixital, ao transferir unha cantidade o noso banco descontarao da nosa conta para engadilo á conta que a persoa destinataria ten no seu propio banco. Hai un conxunto de entidades, nas que todos confiamos, que levan conta de qué diñeiro ten cada quen.

Con todo, o problema do dobre gasto é crítico en sistemas descentralizados onde non existe unha autoridade central para verificar a validez das transaccións. Bitcoin e outras criptomoedas resolven este problema utilizando unha rede de consenso e a tecnoloxía de blockchain, onde as transaccións son verificadas por múltiples nodos e rexistradas nun libro maior inmutable replicado en todos os nodos da rede. Este proceso asegura que unha unidade de criptomoeda non poida ser duplicada e gastada novamente, mantendo a integridade e a confianza no sistema.

Ao longo dos anos, Bitcoin gañou popularidade e aceptación, atraendo tanto a investidores como a tecnólogos. Posteriormente, xurdiron outras criptomoedas como Litecoin (Reed, 2017), creada en 2011 como unha bifurcación de Bitcoin, que buscaban mellorar ou diversificar as características de Bitcoin. En 2015, a introdución de Ethereum (Buterin, 2016) revolucionou o mundo das criptomoedas ao permitir a creación de contratos intelixentes, expandindo enormemente as posibilidades de aplicacións descentralizadas, entre elas a posibilidade de crear de maneira sinxela tokens funxibles (ver apartado 3.3). Desde entón, desenvolvéronse miles de tokens funxibles diferentes que funcionan como criptomoedas, cada un con diferentes obxectivos e características, impulsando a evolución do ecosistema financeiro dixital e a tecnoloxía blockchain. A pesar das fluctuacións no mercado e os desafíos regulatorios, as criptomoedas continúan gañando adopción e recoñecemento como unha innovación disruptiva no mundo financeiro.

De feito, as criptomoedas representan unha revolución tanto no ámbito financeiro como no tecnolóxico. No contexto financeiro, o criptodiñeiro ofrece unha alternativa de diñeiro dixital descentralizado ao diñeiro tradicional emitido polos gobernos. Esta descentralización elimina a necesidade dos intermediarios tradicionais que operan nos mercados financeiros, e permite transaccións máis rápidas e económicas. Ademais, as criptomoedas facilitan a inclusión financeira das persoas que non teñen presenza no sistema bancario tradicional, así como a seguridade e a transparencia das transaccións a través dun rexistro inmutable e distribuído. Este enfoque innovador ten xerado novos modelos de negocio e aplicacións que estudaremos noutros capítulos, como os contratos intelixentes (capítulo 4) e as finanzas descentralizadas (DeFi, capítulo 8), que están a axudar a redefinir a maneira en que se realizan e xestionan as transaccións financeiras. A capacidade de realizar transaccións seguras e verificables sen depender de terceiros, xunto coa potencial redución dos custos e o incremento da velocidade das transaccións, son bos indicadores da importancia e o impacto significativo das criptomoedas no mundo moderno.

3.2 Algunhas criptomoedas populares

O cadro 3.1 recolle algunhas das criptomoedas máis populares. Como apuntabamos antes, Bitcoin (cuxo símbolo é BTC) foi a primeira criptomoeda en popularizarse. A oferta de Bitcoin está limitada a 21 millóns de unidades, o que introduce un elemento de escaseza similar ao do ouro. Esta criptomoeda foi pioneira no espazo das finanzas dixitais e impulsou o desenvolvemento dou-

tras criptomoedas e tecnoloxías relacionadas. Ademais, Bitcoin foi obxecto de especulación, e o seu valor experimentou significativas fluctuaciónes ao longo dos anos, atraendo tanto a investidores como a críticos. A súa adopción creceu globalmente, con cada vez máis comerciantes e plataformas aceptándoa como método de pago.

Cadro 3.1. Algunhas das criptomoedas máis populares (RI = retorno anual media do investimento. USDC é en realidade un token funxible (ver apartado 3.3) que funciona como unha stablecoin. Fonte: Yahoo Finanzas).

Criptomoeda	Símbolo	Ano	Capitalización (mill. €)	RI
Bitcoin	BTC	2009	>1.000.000	126 %
Litecoin	LTC	2011	> 4.000	-11 %
Dogecoin	DOGE	2013	> 18.000	62 %
Ether	ETH	2015	> 400.000	79 %
Solana	SOL	2011	> 84.500	641 %
Tether	USDT	2014	> 114.000	0 %
Binance Coin	NBN	2017	> 83.000	135 %
XRP	XRP	2017	> 35.000	-11 %
US Dollar Coin	USDC	2018	> 33.000	0 %

O prezo do Bitcoin disparouse á vez que se convertía nun activo popular, mantendo tradicionalmente unha alta volatilidade (ver figura3.1). En maio de 2016, podía-se comprar un Bitcoin por uns 500 euros. O 30 de xullo de 2024, o prezo dun Bitcoin roldaba os 63.000 euros, cun crecemento por encima do 12.500 %. Aproximadamente cada catro anos (cada 210.000 bloques minados na cadea de bloques de Bitcoin) redúcese á metade o prezo pagado aos mineiros por minar novos Bitcoins, nun evento chamado *halving*. Este mecanismo é unha parte integral do deseño de Bitcoin e ten como obxectivo controlar a oferta total de Bitcoins, limitándoa a 21 millóns de unidades. O impacto do halving no prezo de Bitcoin adoita ser importante. En teoría, ao reducirse a recompensa por bloque minado, a oferta de novos Bitcoins desacelérase, o cal, combinado cunha demanda constante ou crecente, adoita conducir a un aumento no prezo. Historicamente, todos os halvings, realizados en 2012, 2016, 2020 e 2024, foron asociados a aumentos no prezo do Bitcoin nos meses e anos seguintes. En calquera caso, o mercado de criptomoedas depende de multitude de factores, e aínda que o halving é un evento importante, non é o único determinante do prezo.

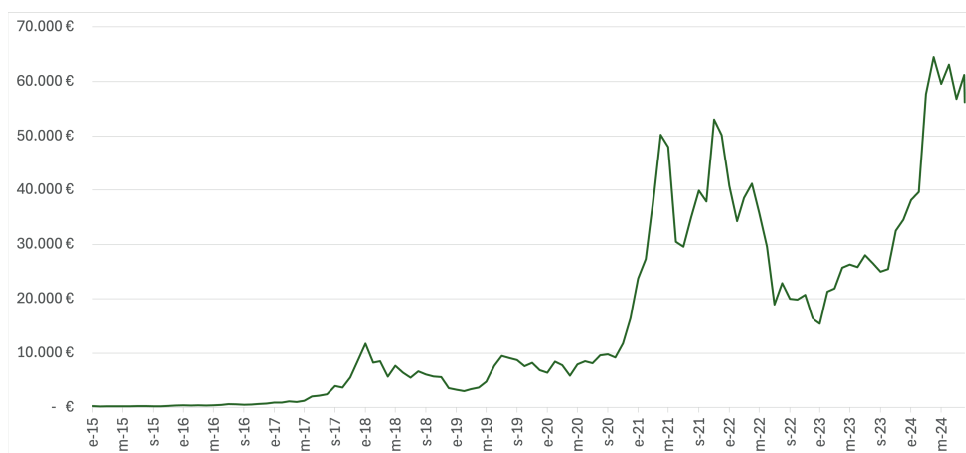


Figura 3.1. Evolución do prezo do Bitcoin en euros. Pódese observar o aumento de prezo despois dos halving de 2016, 2020 e 2024 (fonte: Yahoo Finanzas).

Litecoin (LTC) é unha criptomoeda creada a partir de Bitcoin por Charlie Le, un ex enxeñeiro de Google, en outubro de 2011. Está deseñada para facilitar transaccións rápidas e eficientes, e considérase a miúdo como a *prata* en comparación co *ouro* que sería Bitcoin. Do mesmo xeito que Bitcoin e a maioría das criptomoedas, Litecoin é unha criptomoeda descentralizada de código aberto baseada nunha rede peer-to-peer, o que significa que as transaccións compoñen pouco son controladas por ningunha autoridade central. O mecanismo de consenso utilizado por Litecoin é a proba de traballo, similar ao de Bitcoin. Con todo, a diferenza de Bitcoin, que utiliza o algoritmo de hashing SHA-256 (ver apéndiceB), Litecoin emprega o algoritmo de hashing Scrypt. O uso de Scrypt permite que o proceso de minado sexa máis accesible a unha maior variedade de mineiros, xa que é menos susceptible á especialización do hardware en comparación con SHA-256.

Desde o seu lanzamento, o valor de Litecoin experimentou fluctuacións significativas, reflectindo tanto as tendencias xerais do mercado de criptomoedas como eventos específicos relacionados con Litecoin. Inicialmente, Litecoin tivo un valor relativamente baixo, pero a súa popularidade creceu gradualmente a medida que crecía o interese polas criptomoedas. En maio de 2021, Litecoin alcanzou o seu máximo histórico, superando os 325 euros por cada LTC.

Dogecoin (DOGE, Chohan, 2021) é unha mememoeda, unha criptomoe-da que naceu como un chiste pero rapidamente gañou popularidade e unha comunidade fiel. Creada en 2013 polos enxeñeiros de software Billy Markus



Figura 3.2. Can de raza Shiba Inu do meme Doge (fonte: Wikimedia Commons).

e Jackson Palmer, baseouse no popular meme da internet *Doge*, que representa a un can de raza Shiba Inu (ver figura 3.2) con frases graciosas en texto multicolor. A diferenza doutras criptomoedas que buscaban revolucionar o sistema financeiro, Dogecoin presentouse como unha alternativa divertida e accesible. O seu lanzamento atraeu rapidamente a unha comunidade de entusiastas en redes sociais, promovendo o uso de Dogecoin para propósitos como propinas en liña e microtransacciones. Un momento significativo na historia de Dogecoin foi a súa implicación en varias campañas benéficas e eventos patrocinados pola súa comunidade de fieis. En 2014, recadáronse fondos para enviar ao equipo de bobsleigh xamaicano aos Xogos Olímpicos de Inverno en Sochi. Tamén financiáronse proxectos como a construción de pozos de auga en Quenia e o patrocinio de pilotos de NASCAR.

A popularidade de Dogecoin recibiu un gran impulso en 2021, cando persoeiros públicos como Elon Musk e outras celebridades comezaron a tuitear sobre ela, o que levou a un aumento drástico no seu valor e na súa base de usuarios. Aínda que como no caso de moitas criptomoedas o seu prezo foi tradicionalmente volátil, a súa comunidade segue sendo un dos seus piares máis fortes, considerándose unha criptomoeda relevante e amplamente recoñecida no ecosistema cripto.

Ethereum é unha das cadeas de bloques favoritas dos desenvolvedores de software polo potencial que ofrecen os seus contratos intelixentes para despregar aplicacións descentralizadas. Dispón da súa propia criptomoeda, denominada Ether (ETH) que se utiliza como medio de intercambio para os servizos e aplicacións que se ofrecen sobre a devandita cadea, por exemplo cubrir os custos de despregar e executar contratos intelixentes, crear tokens non funxibles NFT, e mesmo crear novas criptomoedas sobre dita cadea de bloques. Á parte da súa utilización como medio de pago no ecosistema Ethereum, o Ether tamén se converteu en reserva de valor fóra da cadea, experimentando un enorme crecemento. Desde abril de 2016 ata finais de xullo de 2024, o seu prezo pasou duns 10 euros por cada Ether a preto de 3.500 euros, aumentando o seu valor en máis dun 30.000 %.

A criptomoeda Solana (SOL, Pierro y Tonelli, 2022) é a criptomoeda da cadea de bloques Solana, que á súa vez se desenvolveu co obxectivo de potenciar a popularización das finanzas descentralizadas (DeFi), as aplicacións descentralizadas (DApps) e os contratos intelixentes, Solana funciona cun mecanismo de consenso propio que é un híbrido da proba de participación e a proba de historia (ver capítulo 2), que permite procesar as transaccións dunha forma rápida e segura. Cando se lanzou en 2020, o prezo de SOL estaba na contorna de 0,75 euros por cada SOL. A finais de xullo de 2024, o seu prezo roldaba os 185 euros, cunha ganancia de máis do 23.000 %.

A diferenza doutras criptomoedas, Tether (USDT, Bianchi, Rossini y Iacopini, 2020) é un exemplo do que se coñece como *stablecoin*, o que significa que está apoiada por moedas fiduciarias como o dólar estadounidense e o euro (neste caso o dólar estadounidense), e que hipoteticamente mantén un valor igual á denominación á que está asociada (Sidorenko, 2020). En teoría, o valor de Tether suponse que é máis consistente que outras criptomoedas ou tokens, polo que adoita ser atractivo para investidores que desconfían da extrema volatilidade doutras criptomoedas como o Bitcoin ou o Ether. Ao ofrecer unha maior estabilidade no seu valor, as stablecoins permiten realizar transaccións diarias e manter valor nunha contorna menos volátil.

Existen catro tipos de stablecoins, en función do mecanismo que utilizan para manter a súa estabilidade:

1. Stablecoins avaladas con moedas fiduciarias, apoiadas por reservas de moedas fiduciarias, mantidas nunha proporción de 1:1 nunha conta bancaria. Outro exemplo popular ademais de Tether sería o USD Coin (USDC).

2. Stablecoins avaladas con criptomonedas, apoiadas por outras criptomonedas que se utilizan como aval. Para mitigar a volatilidade das criptomonedas subxacentes, adoitan estar sobreavaladas. Un exemplo é DAI, un token que está avalado por Ether (ETH) e outras criptomonedas na plataforma MakerDAO.
3. Stablecoins algorítmicas, que non están indexadas a activos tradicionais, senón que utilizan algoritmos e contratos intelixentes para controlar a oferta e demanda e manter un valor estable. Exemplos deste tipo serían Ampleforth (AMPL) e TerraUSD (UST).
4. Stablecoins avaladas con bens físicos como o ouro. Un exemplo é PAX Gold (PAXG), que está respaldado por lingotes de ouro.

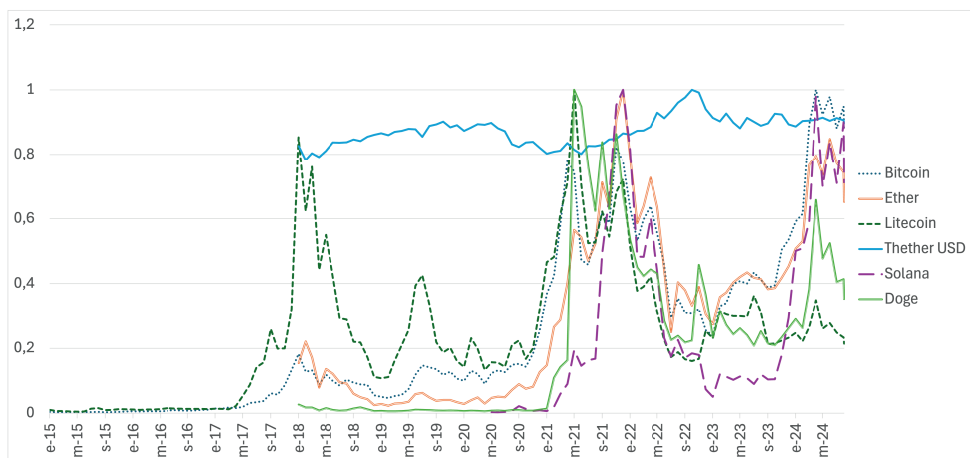


Figura 3.3. Evolución do prezo das criptomonedas nomeadas neste capítulo. No eixo vertical represéntase o valor de cada criptomoneda ao longo do tempo en euros, como unha porcentaxe do seu valor máximo. Ao ser unha stablecoin referenciada ao dólar estadounidense, a gráfica de Tether ten unha forma similar á gráfica da evolución da paridade dólar/euro (fonte: Yahoo Finanzas).

A figura 3.3 ilustra a evolución do prezo das criptomonedas descritas neste capítulo (Bitcoin, Litecoin, Dogecoin, Ether, Tether e Solana). Podemos observar unha gran volatilidade en xeral, con picos a principios de 2018, 2021 e 2022. En 2017 e principios de 2018, houbo unha explosión de ofertas iniciais de moedas (ICO, Initial Coin Offer), onde moitas startups e proxectos baseados en blockchain lanzaron as súas propias criptomonedas para recadar fondos. Isto atraeu a moitos investidores que esperaban obter altos rendementos, elevando os prezos das criptomonedas, que ademais recibiron unha cobertura mediática sen precedentes, reforzándose a percepción positiva sobre estes activos e a expectativa de que os prezos seguirían subindo.

A finais de 2020 e principios de 2021, as grandes institucións financeiras e empresas comezaron a investir en criptomoedas, o que proporcionou lexitimidade a este mercado e atraeu máis capital ao mercado. Estas decisións, xunto cos paquetes de estímulo económicos debido á pandemia de COVID-19 e a explosión de proxectos e plataformas DeFi que proporcionaron liquidez adicional, tiveron un impacto moi importante na demanda de criptomoedas e por tanto nos prezos. En 2022 prodúcese o pico de interese nos tokens non funxibles (NFT) e o mercado de arte dixital, o que á súa vez impulsou a demanda de criptomoedas, especialmente aquelas utilizadas para comprar NFT.

No momento de escribir este manual (verán de 2024) estamos a asistir a outro momento álxido do mercado de criptomoedas.

3.3 Criptomoedas e tokenización

No eido das cadeas de bloques, a tokenización refírese ao proceso de converter un activo físico ou dixital nun token dixital que se pode rexistrar, xestionar e transferir nunha blockchain. Estes tokens representan un dereito ou propiedade sobre o activo subxacente e poden incluír unha ampla gama de activos, como bens raíces, accións, obras de arte, dereitos de propiedade intelectual, entre outros. A tokenización permite fraccionar os activos, o que facilita a posibilidade de que varias persoas posúan partes dun mesmo ben. Ademais, a tokenización pode aumentar a liquidez de activos tradicionalmente carentes de liquidez, xa que os tokens poden comercializarse e intercambiarse facilmente en mercados dixitais. En definitiva, a tokenización permite transformar activos do mundo real en versións dixitais negociables dentro do ecosistema blockchain, abrindo novas oportunidades de investimento e democratizando o acceso a estes activos.

Podemos dicir que as criptomoedas son un tipo de tokens funxibles. Dicimos que un token é funxible porque cada unidade dese token é idéntica e completamente intercambiable con outra unidade do mesmo tipo. En termos económicos, funxibilidade significa que un ben ou activo ten o mesmo valor e pode ser substituído por outro do seu mesma clase sen perder valor nin cambiar a súa natureza. Por exemplo, unha criptomoeda como Bitcoin ou un token ERC20 dos que veremos no capítulo 4 é funxible porque unha unidade de BTC é exactamente igual a calquera outra unidade de BTC, e ambos poden intercambiarse sen ningunha distinción. Do mesmo xeito que un billete de dez euros pode substituírse por outro billete de dez euros sen perder o seu valor, os

tokens funxibles compórtanse desta maneira dentro dunha blockchain, sendo facilmente divisibles, homoxéneos e replicables en valor.

En cambio, os tokens non funxibles como os que veremos no capítulo 7 son únicos e non intercambiabes entre si, xa que cada un representa un activo diferente.

A principal diferenza entre unha criptomoeda e un token funxible radica no seu uso e función dentro do ecosistema blockchain. Unha criptomoeda é unha moeda dixital nativa dunha blockchain, deseñada principalmente como medio de intercambio, reserva de valor ou unidade de conta. Exemplos de criptomoedas inclúen Bitcoin e Ether, que teñen as súas propias redes blockchain descentralizadas, e a súa función primaria é actuar como diñeiro dixital. As súas características (cómo se xera nova criptomoeda, cal será o seu subministro máximo, etc.) forman parte do propio deseño da blockchain que as soporta e dos seus procedimentos de gobernanza e consenso.

Doutra banda, un token funxible é unha representación dixital de activos que se crean dentro dunha plataforma blockchain existente, como a cadea Ethereum. Os tokens funxibles, como os tokens ERC20 de Ethereum, son intercambiabes e teñen o mesmo valor entre si. Aínda que poden utilizarse como moedas, adoitan representar outros activos, dereitos ou utilidades dentro dunha plataforma ou aplicación específica. As súas características defínense por medio do código dun contrato intelixente. Por exemplo, para crear un token funxible que siga o estándar ERC20 de Ethereum, teremos que asignarlle un valor a un conxunto de variables de estado no contrato que definen o seu nome, o seu símbolo ou o seu subministro máximo. No apartado 4.4 do capítulo 4 veremos un exemplo de creación dun token funxible denominado *PATACON*.

3.4 Creación de criptomoeda

O proceso de creación dunha criptomoeda implica varios pasos técnicos e conceptuais que permiten establecer un novo activo dixital baseado na tecnoloxía de cadea de bloques. O primeiro paso consiste na conceptualización e o deseño das súas características principais, é dicir, definir o propósito da criptomoeda, a súa subministración total e as regras que gobernarán a súa produción e o seu uso. No caso de Bitcoin, Satoshi Nakamoto, o individuo ou grupo responsable da súa creación, deseñou Bitcoin como unha moeda dixital descentralizada cunha subministración máxima de 21 millóns de unidades.

Unha vez definido o deseño, o seguinte paso é desenvolver o software que permitirá o funcionamento da rede de blockchain que se utilizará como libro maior para levar un rexistro dos depósitos e das transaccións con esa criptomoeda. No caso de Bitcoin, Nakamoto lanzou o software Bitcoin en 2009, o cal implementa todas as regras que gobernan dita criptomoeda, incluíndo o mecanismo de consenso baseado na proba de traballo e a estrutura dos bloques.

A figura 3.4 ilustra o minado ou forxa de bloques, un proceso fundamental na creación de novas unidades de calquera criptomoeda. Como explicamos no capítulo 2, este proceso utilízase para rexistrar as transaccións levadas a cabo na cadea de bloques, que no caso das criptomoedas consisten en transaccións de determinada cantidade de criptomoeda desde unha dirección de orixe a unha dirección de destino. Cada vez que se acepta un novo bloque na cadea, o mineiro ou forxador recibe unha recompensa en forma de novas criptomoedas recentemente creadas segundo o protocolo establecido na definición do mecanismo de consenso. No caso de Bitcoin, os mineiros utilizan potentes computadoras para resolver complexos problemas matemáticos, os cales verifican e rexistran as transaccións na blockchain de acordo co mecanismo de consenso de proba de traballo. Cada vez que se resolve un problema e se crea un novo bloque, o mineiro recibe unha recompensa en forma de novos bitcoins. Esta é a forma en que se crean novos bitcoins e distribúense no ecosistema.

Despois da creación inicial e o minado dos primeiros bloques, a criptomoeda distribúese a través de varios medios, incluíndo as plataformas de intercambio de criptomoedas ou exchanges, pagos por bens e servizos, e outros métodos de transferencia. A adopción e o uso da criptomoeda crecen a medida que máis persoas e empresas comezan a aceptala e a participar nos intercambios. Unha criptomoeda require mantemento e actualización constante para mellorar a seguridade, escalar a rede, e engadir novas funcionalidades. No caso de Bitcoin, isto realízase a través de propostas de mellora de Bitcoin (BIP) e a colaboración da comunidade de desenvolvedores.

Como vimos no apartado anterior, tamén é posible utilizar redes blockchain xa existentes, e despregar tokens funxibles que funcionan na práctica como criptomoedas utilizando contratos intelixentes nunha blockchain. Por exemplo, os tokens funxibles USD Coin, Chainlink, Dai, Uniswap, Aave, Shiba Inu (unha criptomoeda meme inspirada en Dogecoin), Wrapped Bitcoin (unha versión tokenizada de Bitcoin), Basic Attention Token (utilizado no ecosistema do navegador Brave para recompensar a atención dos usuarios) ou Decentraland

(moeda utilizada no metaverso Decentraland) foron creados mediante contratos intelixentes sobre a blockchain Ethereum. No caso dos tokens funxibles creados con contratos intelixentes, é o propio contrato intelixente o que define a cantidade máxima de criptomoeda, o seu nome, o seu símbolo, o seu prezo inicial na criptomoeda da blockchain base (por exemplo en Ether no caso das criptomoedas creadas sobre Ethereum) e o resto das súas características. Unha vez creado, o contrato intelixente convértese o instrumento para distribuír o token e controlar o seu fluxo.

Para adquirir criptomoedas ou tokens funxibles no mercado, calquera persoa pode rexistrarse nun portal de criptomoedas como Binance, Coinbase ou Kraken, portais que temén coñécense como *exchanges*. Trátase de plataformas en liña onde se poden comprar, vender e intercambiar criptomoedas. Coa paulatina regulación destes mercados, para rexistrarse xeralmente requírese proporcionar información persoal e verificar a identidade a través de documentos oficiais.

Unha vez que nos rexistramos, debemos depositar fondos na nosa conta no portal. Para iso utilizaremos calquera medio de pago dispoñible para realizar pagos na internet, como transferencias bancarias, tarxetas de crédito ou débito, Bizum, PayPal etc.. Unha vez que os nosos fondos están depositados na nosa conta, podemos comprar criptomoedas como compraríamos calquera ben ou servizo en liña, seleccionando a criptomoeda desexada e a cantidade que desexamos adquirir. As transaccións realízanse a prezos de mercado ou mediante ordes limitadas onde o usuario establece o prezo que está disposto a pagar. A figura 3.5 ilustra este proceso.

Para transferir criptomoedas dunha persoa a outra utilizaremos polo xeral unha carteira ou *wallet* de criptomoedas. Trátase de dispositivos ou aplicacións informáticas capaces de interactuar coas cadeas de bloques que dan soporte a cada criptomoeda para solicitar transaccións e capturar o resultado das mesmas. Existen varios tipos de carteiras, como as carteiras en liña, xeralmente accesibles como extensións dos navegadores máis populares (*hot wallets*); carteiras físicas (*cold wallets*), que son dispositivos hardware criptográficos que almacenan a información necesaria para interactuar con cada blockchain (direccións, claves privadas, saldos etc.), e carteiras de software, que non son máis que aplicacións informáticas que implementan as funcións dunha carteira.

As carteiras provén unha interface para introducir a dirección do destinatario e a cantidade a transferir, almacenan as claves privadas necesarias para acceder e xestionar as criptomoedas e xestionan o proceso de firma da

transacción coa clave privada. As claves privadas deben manterse seguras e nunca compartirse, xa que quen teña acceso á clave privada terá control total sobre os fondos asociados. Non podemos esquecer que unha das características básicas do mundo blockchain é a inexistencia dunha entidade central de control ou regulación. Se extraviamos as nosas claves de acceso á cadea de bloques subxacente, non poderemos recuperar de ningunha maneira os fondos que puidésemos ter nela.

A persoa que quere recibir cantidades dunha criptomoeda debe rexistrar unha dirección da cadea de bloques que dá soporte á devandita criptomoeda na súa carteira. Esta dirección é unha cadea única de caracteres alfanuméricos que actúa como un número de conta bancaria no mundo das criptomoedas. A persoa que transfire a criptomoeda debe ingresar a dirección do destinatario na súa carteira e especificar a cantidade para transferir. Despois, confirma a transacción, xeralmente ingresando unha clave privada asociada á súa propia dirección ou utilizando algún método de autenticación. A transacción solicítase desde a carteira á rede blockchain correspondente, onde é verificada e engadida a un bloque da cadea. Este proceso pode levar desde uns poucos minutos ata varias horas, dependendo da conxestión da rede e a criptomoeda en cuestión.

No ámbito de blockchain, a custodia refírese ao control e almacenamento seguro de criptomoedas, tokens funxibles e outros activos dixitais en nome dun usuario ou institución. Os servizos de custodia permiten aos propietarios de criptomoedas delegar a responsabilidade de protexer as súas claves privadas nunha entidade de terceiros, como os exchanges dos que falábamos antes ou provedores especializados en custodia. Existen dous tipos principais de custodia: custodia en frío, que implica o almacenamento dos activos en dispositivos sen conexión a Internet para minimizar o risco de ataques informáticos, e custodia en quente, que permite o acceso inmediato aos activos, pero cun maior risco de exposición a vulnerabilidades en liña. Os custodios proporcionan unha capa de seguridade adicional ao ofrecer protección contra roubos e fraudes, o cal é fundamental para institucións financeiras e grandes investidores que xestionan grandes volumes de criptomoeda.

3.5 Aplicacións e casos de uso

As criptomoedas, e en xeral os tokens funxibles, utilízanse cada vez máis como instrumentos para realizar pagos e transferencias transfronteirizas debido á súa capacidade de operar sen intermediarios tradicionais como bancos

e entidades financeiras. Aínda que a rede pode estar conxestionada en momentos puntuais, estas transaccións lévanse a cabo dunha maneira máis rápida e eficiente que as transferencias tradicionais das entidades financeiras, reducindo significativamente os custos e os tempos de procesamento. As criptomoedas permiten enviar e recibir fondos a nivel mundial case instantaneamente, sen importar as fronteiras xeográficas, o cal é especialmente beneficioso para persoas e empresas que necesitan realizar pagos internacionais frecuentes. Ademais, a natureza descentralizada e a tecnoloxía blockchain subxacente aseguran un alto nivel de transparencia e seguridade nas transaccións, minimizando o risco de fraude e ofrecendo unha maior confianza nas operacións financeiras internacionais.

Como veremos no capítulo 4, as criptomoedas tamén desempeñan un papel fundamental na cobertura dos custos operativos dos contratos intelixentes, que á súa vez se utilizan para facer realidade as aplicacións descentralizadas ou dApps, ao proporcionar un medio de pago (a tarifa en gas ou *gas fees*) para executar estas operacións na blockchain. No caso de Ethereum, por exemplo, os usuarios deben pagar tarifas en Ether para desprezar contratos intelixentes, interactuar con eles ou realizar transaccións. Estas tarifas compensan aos mineiros ou validadores polos recursos computacionais e o espazo de almacenamento que utilizan ao procesar e verificar as transaccións e operacións de contratos intelixentes. Este modelo asegura que a rede blockchain manteña a súa seguridade e funcionalidade, incentivando a participación activa e continua dos nodos que sosteñen a rede. Desta maneira, as criptomoedas non só facilitan a creación e execución de contratos intelixentes, senón que tamén aseguran a sustentabilidade e eficiencia do ecosistema blockchain no seu conxunto.

As criptomoedas son tamén un elemento esencial nas finanzas descentralizadas ou DeFi que veremos en detalle no capítulo 8, ao proporcionar o medio de intercambio e o activo subxacente necesario para operar dentro deste ecosistema financeiro emerxente. As plataformas DeFi utilizan contratos intelixentes nunha blockchain para crear servizos financeiros que non dependen de intermediarios tradicionais. Neste contexto, as criptomoedas permiten aos usuarios participar nunha variedade de actividades financeiras, incluíndo préstamos, depósitos de aforro a prazo, comercio e seguros, de maneira completamente autónoma e descentralizada. Ao eliminar intermediarios, as DeFi buscan democratizar o acceso aos servizos financeiros, reducir custos e aumentar a transparencia e a seguridade nas transaccións. Ademais, as criptomoedas poden ser utilizadas como garantía en préstamos, facilitando a creación de mercados de préstamos entre iguais (*peer-to-peer*), e facilitan a

liquidez nos mercados descentralizados, así como o intercambio de valores de forma máis eficiente e sen restricións xeográficas.

Por outra banda, como comentamos antes, as criptomoedas e os tokens funxibles desempeñan un papel fundamental na tokenización de activos e na creación de tokens non funxibles (NFTs, capítulo 7), ao proporcionar a infraestrutura e os fondos necesarios para estas transaccións. Os NFTs son unha forma especial de tokenización que se utiliza para crear tokens únicos e indivisibles que representan a propiedade dun activo. As criptomoedas facilitan estas operacións ao actuar como a moeda de intercambio nestes mercados, definindo un novo paradigma de propiedade e transacción no mundo dixital, permitindo a creación e o intercambio de valor de formas innovadoras e seguras.

Ademais dos usos mencionados anteriormente, as criptomoedas e os tokens teñen unha ampla variedade de aplicacións adicionais:

- Ao permitir enviar diñeiro a través de fronteiras de maneira rápida, son especialmente adecuadas no caso das remesas internacionais, especialmente no caso de remesas de persoas migrantes en rexións con sistemas bancarios ineficientes ou custosos.
- Algunhas criptomoedas, como Monero e Zcash, céntranse na privacidade, permitindo transaccións que son difíciles de rastrexar, o que é útil para protexer a privacidade financeira das persoas.
- Permiten realizar doazóns de maneira transparente e directa a organizacións benéficas, sen intermediarios e coa posibilidade de rastrexar o uso dos fondos, xa que toda transacción queda rexistrada de maneira inmutable na cadea de bloques.
- Facilitan os micropagos debido ás súas baixas comisións, o que é útil para pagar por contido dixital, xogos, servizos de streaming e outros produtos e servizos que requiren pagos pequenos e frecuentes.
- As ofertas iniciais de criptomoeda (ICO) permiten ás startups e iniciativas de crowdfunding recadar fondos directamente dos investidores sen necesidade de intermediarios, facilitando o acceso ao capital.
- Poden servir como unha forma de investimento e aforro, similar a como se utilizan as accións e bonos. Pódense comprar e manter criptomoedas coa esperanza de que o seu valor aumente no futuro.

- Permiten aos freelancers e nómadas dixitais recibir pagos de maneira rápida e segura, sen depender de plataformas de pago tradicionais que poden cobrar altas comisións.
- No mundo dos videoxogos e a realidade virtual, utilízanse para comprar bens virtuais, participar en economías de xogo e realizar transaccións dentro do xogo.
- No ámbito do e-commerce, son aceptadas por un número crecente de comerciantes e plataformas de comercio electrónico, permitindo aos usuarios comprar bens e servizos en liña de maneira directa.
- No Internet das Cousas (IoT), permiten o intercambio de valor entre dispositivos de maneira autónoma e segura, facilitando novos modelos de negocio e servizos automatizados.

3.6 Desafíos e implicacións económicas e legais

Como comentamos nos parágrafos anteriores, a volatilidade é unha característica relevante das criptomoedas, entendida como a amplitude das fluctuacións do seu prezo en períodos curtos de tempo. A diferenza das moedas fiduciarias tradicionais, cuxo valor tende a ser máis estable ao dispoñer de instrumentos de control como os gobernos ou os bancos centrais, as criptomoedas poden experimentar cambios drásticos no seu valor debido a varios factores, como a especulación, os cambios na percepción do seu valor por parte dos investidores, eventos xeopolíticos, accións dos gobernos ou as entidades financeiras tradicionais e os avances tecnolóxicos. Esta alta volatilidade pode ser tanto unha oportunidade como un risco. Para os investidores, ofrece a posibilidade de obter rendementos significativos en curtos períodos, pero tamén implica un alto risco de perdas substanciais. A volatilidade extrema pode desalentar o uso de criptomoedas como medio de intercambio e reserva de valor, limitando a súa adopción xeneralizada.

En concreto, a especulación no mercado de criptomoedas é un fenómeno común e refírese á compra e venda destes activos co obxectivo de obter ganancias a partir das fluctuacións de prezos. Os especuladores xogan un papel significativo no mercado de criptomoedas, xa que a súa actividade contribúe á volatilidade. A especulación pode estar motivada por varios factores, como as expectativas de novos desenvolvementos tecnolóxicos que teñan un impacto drástico nos prezos, as perspectivas de cambios na adopción institucional ou de cambios regulatorios, ou simplemente o efecto de seguir a tendencia en

resposta ás variacións de prezos. Aínda que a especulación pode proporcionar liquidez ao mercado, tamén creou no pasado burbullas de prezos que, ao estalar, causaron perdas significativas.

En termos legais, a alta volatilidade e a especulación desafían aos reguladores para atopar instrumentos adecuados para a protección dos investidores nunha contorna financeira descentralizada. As regulacións deben abordar a transparencia, a divulgación de información e a protección contra fraudes e manipulacións do mercado. A natureza global das criptomoedas engade unha capa de complexidade, xa que os esforzos regulatorios débense coordinar a nivel internacional para ser efectivos. De feito, os esforzos regulatorios internacionais no ámbito das criptomoedas intensificáronse en resposta ao crecemento exponencial e á integración destes activos no sistema financeiro global. Organismos como o Consello de Estabilidade Financeira incluíron no seu programa de traballo desenvolver un marco de normas globais para a regulación dos cryptoactivos, co fin de xestionar os riscos para a estabilidade financeira e asegurar unha aplicación coherente en distintas xurisdicións. Estes esforzos buscan equilibrar a innovación coa protección do consumidor e a integridade do mercado financeiro.

Na actualidade, a normativa no ámbito das criptomoedas varía significativamente entre países, reflectindo a diversidade de enfoques cara a este emerxente sector financeiro. Por exemplo, en Canadá, as criptomoedas non se recoñecen como un medio de pago legal, pero as transaccións de criptomoedas son legais estando os exchanges de criptomoedas suxeitos a regulacións financeiras específicas. En Estados Unidos, as criptomoedas son legais e considéranse unha moeda convertible libremente, coas empresas de servizos relacionados con criptomoedas suxeitas aos requisitos da Lei de Secreto Bancario. O Reino Unido ten intención de converterse nun centro global de criptomoedas, con lexislación recente que outorga ás stablecoins o status de medio de pago e permite a integración da cadea de bloques nos mercados financeiros. Xapón, pola súa banda, considera ás criptomoedas como propiedade ou mercadoría, pero non como un medio de pago, e aprobou leis que permiten ás compañías de capital de risco e aos fondos de investimento posuír criptomoedas. Estes exemplos ilustran como os diferentes marcos regulatorios poden influír na adopción e o uso de criptomoedas en todo o mundo.

A sustentabilidade e o impacto ambiental das criptomoedas son outro desafío importante. O proceso de minería de criptomoedas, especialmente para aquelas que utilizan o mecanismo de consenso de proba de traballo, implica un

consumo enerxético considerable debido á necesidade de resolver problemas matemáticos complexos para validar transaccións e crear novos bloques na cadea. Isto levou a un aumento nas emisións de gases de efecto invernadoiro, xa que moitos mineiros utilizan electricidade xerada a partir de fontes non renovables. Con todo, a industria está a responder co desenvolvemento de criptomoedas ecolóxicas que buscan minimizar a pegada de carbono mediante a eficiencia enerxética e a redución do impacto ambiental. Estas criptomoedas ecolóxicas manteñen a integridade da blockchain mentres buscan ser máis sostibles nun contexto onde a sustentabilidade está a converterse nun criterio esencial para o futuro da tecnoloxía blockchain, co obxectivo de harmonizar o crecemento dixital coa responsabilidade ambiental.

Para mellorar a sustentabilidade das criptomoedas pódense adoptar varias estratexias. Unha das máis importantes é a transición cara a fontes de enerxía renovable, como a solar, eólica ou hidroeléctrica, que reducen significativamente a pegada de carbono da minería de criptomoedas. Ademais, a optimización de algoritmos e o desenvolvemento de hardware máis eficiente melloran a eficiencia enerxética dos equipos de minería, diminuindo así o consumo de enerxía necesario para manter a rede blockchain. Outra estratexia relevante é a educación e a conciencia ambiental dentro da comunidade criptográfica, promovendo prácticas máis responsables e éticas. Finalmente, proxectos como Ethereum teñen migrado ou están migrando a mecanismos de consenso máis sostibles como a proba de participación, o que chega a reducir o consumo total de enerxía nun 99.95 %, demostrando que é posible unha criptoconomía máis verde e sostible (Lin, 2023).

Por outra banda, as criptomoedas están a emerxer como un elemento moi relevante na evolución dos delitos informáticos e na internacionalización das bandas criminais. A natureza descentralizada e o anonimato potencial que ofrecen as criptomoedas facilitan diversas actividades ilícitas, incluíndo o lavado de diñeiro e o financiamento do terrorismo. Segundo Europol, é esencial fortalecer a cooperación internacional e os mecanismos de seguimento para combater estes delitos, xa que se identificou un aumento no uso de criptomoedas para o lavado de diñeiro, especialmente en relación co tráfico de drogas e os ataques de ransomware, e as autoridades están a responder cun incremento nas incautacións de criptomoedas relacionadas con actividades criminais.

De todos os xeitos, a mesma tecnoloxía blockchain que permite estas actividades tamén ofrece instrumentos para rastrexar transaccións ilícitas e obter evidencia, mesmo anos despois de cometido o delito. O rastrexo de

transaccións de criptomoedas realízase a través de exploradores de blockchain, que son ferramentas en liña públicas que permiten visualizar todas as transaccións realizadas nunha rede blockchain concreta. Cada transacción rexistrada na cadea de bloques inclúe detalles como a cantidade transferida, unha marca de tempo e as direccións do remitente e do receptor. Calquera persoa pode ingresar a dirección dunha carteira na barra de procura dun explorador de blockchain para ver o historial de transaccións asociado a esa carteira. Ademais, existen servizos e ferramentas especializadas que poden analizar e seguir o fluxo de fondos entre diferentes carteiras, o que é especialmente útil para investigacións financeiras e a loita contra actividades ilícitas. Aínda que as criptomoedas poden ofrecer certo grao de anonimato (non existindo un rexistro da relación entre unha dirección na blockchain e a persoa concreta que coñece a clave privada que permite usala para transferir fondos), a tecnoloxía blockchain proporciona de maneira nativa unha forma transparente e auditable de seguir as transaccións, o que axuda ás autoridades para rastrexar operacións sospeitosas e combater o crime organizado.

En definitiva, a protección da privacidade no rastrexo de transaccións de criptomoedas é un tema complexo que implica un equilibrio entre a seguridade e os dereitos individuais. Aínda que as transaccións na blockchain son públicas, a identidade dos usuarios mantense protexida ao non existir unha relación trazable entre as direccións na cadea e a identidade no mundo real. Ademais, para unha maior privacidade, existen criptomoedas como Monero e Zcash que ofrecen características avanzadas de anonimato, como direccións ocultas e probas de coñecemento cero, e técnicas como as transaccións confidenciais e iniciativas como CoinJoin axudan a ocultar a cantidade transferida e as partes involucradas nunha transacción. A pesar destas medidas, a normativa pode requirir a identificación das persoas ou entidades participantes nunha transacción en certos casos, como parte das medidas contra o lavado de diñeiro e o financiamento do terrorismo. Por tanto, a privacidade no uso de criptomoedas depende tanto da tecnoloxía utilizada como das leis aplicables en cada xurisdición.

3.7 O futuro das criptomoedas

Unha das tendencias máis significativas no ámbito das criptomoedas é a crecente adopción por parte de institucións financeiras e empresas. Nos últimos anos, grandes multinacionais como Tesla ou PayPal comezaron a incorporar criptomoedas nos seus modelos de negocio, xa sexa aceptándoas

como método de pago ou investindo directamente nelas. Esta tendencia suxire un futuro onde as criptomoedas poderían empezar a formar parte do sistema financeiro global, non só como un activo de investimento ou especulación, senón tamén como un medio de intercambio aceptado universalmente. A adopción institucional tamén contribuirá a proporcionar estabilidade ao mercado de criptomoedas, reducindo a súa volatilidade e aumentando a confianza do público nestes activos dixitais.

Outro elemento fundamental no futuro das criptomoedas é o crecemento do ecosistema de finanzas descentralizadas ou DeFi, popularizándose a posibilidade de realizar calquera tipo de operación financeira (e.g., préstamos, compravendas, seguros etc.) sen a participación das entidades financeiras ou aseguradoras tradicionais. Este ecosistema, construído sobre plataformas blockchain, está a evolucionar moi rápido debido á súa capacidade para ofrecer servizos financeiros máis accesibles, transparentes e menos custosos. A medida que a tecnoloxía madure e as solucións DeFi sexan máis seguras e fáciles de usar, é bastante probable que a súa adopción continúe crecendo, desafiando os modelos financeiros tradicionais e expandindo o uso das criptomoedas.

Como consecuencia da popularización das criptomoedas, estamos a asistir a unha intensificación nos esforzos regulatorios a nivel global. Países como Estados Unidos ou China, ademais da Unión Europea, están a desenvolver marcos regulatorios para manexar os riscos asociados coas criptomoedas, como o lavado de diñeiro, a evasión fiscal, e a volatilidade dos mercados. Quizá a súa regulación sexa o maior reto de todos, xa que as criptomoedas operan nunha contorna global e descentralizada que transcende as fronteiras nacionais. O futuro do mercado de criptomoedas dependerá en gran medida de como os gobernos e as institucións internacionais aborden estes desafíos regulatorios, equilibrando a protección do consumidor coa innovación tecnolóxica.

A aparición de criptomoedas cuxo valor está vinculado a activos máis estables como o dólar estadounidense ou o euro é outra tendencia relevante. As stablecoins están deseñadas para mitigar a volatilidade inherente ás criptomoedas tradicionais, o que as fai máis atractivas para transaccións cotiás e como reserva de valor. Ademais, varios bancos centrais ao redor do mundo están a explorar a posibilidade de emitir as súas propias moedas dixitais, coñecidas como CBDCs (do inglés *Central Bank Dixital Currencies* ou moedas dixitais emitidas por bancos centrais). Estas iniciativas poderían transformar radicalmente o sistema financeiro global, ofrecendo unha forma de diñeiro totalmente dixital e centralizada, no sentido de que está apoiado

pola autoridade dos bancos centrais, pero coas vantaxes tecnolóxicas das criptomoedas, como a súa transparencia ou a súa garantía de permanencia debido á inmutabilidade das transaccións nunha cadea de bloques.

En calquera caso, a innovación tecnolóxica continúa sendo o motor da evolución das criptomoedas. Os desenvolvedores están a traballar en novas solucións para mellorar a escalabilidade e a eficiencia das redes blockchain, como as solucións de segunda capa (layer 2) e as redes baseadas no mecanismo de proba de participación. Estas innovacións son moi importantes para conseguir que as criptomoedas sexan escalables para poder xestionar un volume maior de transaccións sen comprometer a seguridade ou a descentralización. O éxito destas melloras tecnolóxicas determinará en gran medida a capacidade das criptomoedas para converterse nunha alternativa viable aos sistemas financeiros tradicionais.

En definitiva, as criptomoedas teñen o potencial de desintermediar moitas das funcións tradicionais dos sistemas financeiros, facer os mercados máis inclusivos e transparentes, e estimular a innovación a través da dixitalización de activos e procesos. Isto podería levar a unha economía global máis conectada, eficiente e equitativa.

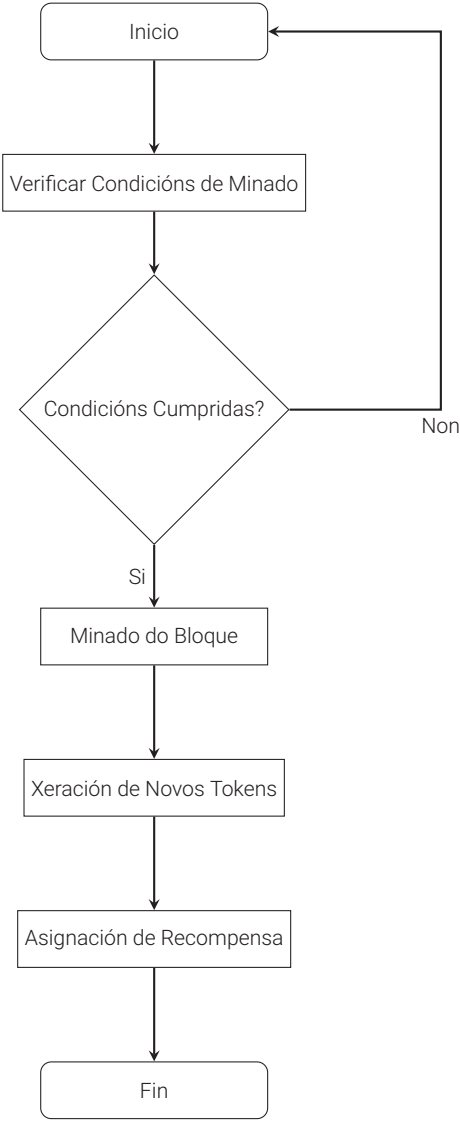


Figura 3.4. Xeración de novas unidades dunha criptomoeda.

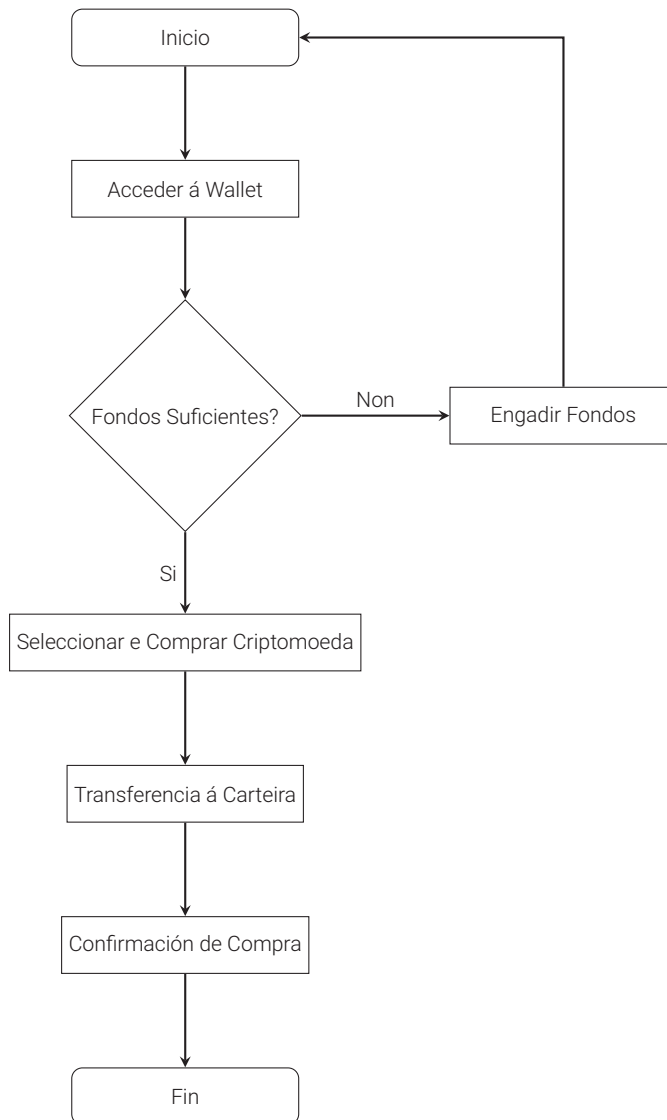


Figura 3.5. Compra de criptomoeda utilizando unha carteira.

Capítulo 4

Contratos intelixentes

En xeral, os contratos intelixentes defínense como contratos dixitais autoexecutables nos que os termos do acordo entre as partes asinantes do contrato exprésanse directamente en liñas de código software. Dende o punto de vista da súa realización práctica, trátase de software que se executa nos ordenadores que forman a rede blockchain. Noutras palabras, un contrato intelixente é un programa informático que se pode activar de maneira automática, que representa mediante código as cláusulas e condicións dun acordo entre dúas ou máis partes. No ámbito das cadeas de bloques, materialízase nun programa informático almacenado nun nodo dunha rede blockchain, que se executa na devandita rede, e que pode programarse para activar automaticamente accións, transaccións ou outros eventos en función de condicións predefinidas.

4.1 Para que se utilizan os contratos intelixentes

Os contratos intelixentes poden utilizarse para unha ampla gama de aplicacións, por exemplo as transaccións financeiras, a xestión de cadeas de subministración, a creación e transferencia de activos dixitais ou os sistemas de votación. Están deseñados para ser transparentes e seguros, e poden axudar a reducir os custos das transaccións, mellorar a eficiencia e aumentar a transparencia e a confianza entre os diferentes actores que participan nunha blockchain.

Os contratos intelixentes, ao automatizar acordos e transaccións sen intermediarios, permiten programar regras e condicións que se executan automaticamente cando se cumpren certos criterios. Estas ferramentas son polo tanto moi útiles na construción de solucións descentralizadas que operan de

forma eficiente, transparente e segura, eliminando a necesidade de confiar en terceiros para validar e executar contratos. Un exemplo avanzado da utilización de contratos intelixentes son as organizacións autónomas descentralizadas (Decentralized Autonomous Organizations, DAO). Unha DAO é unha organización xestionada de forma descentralizada mediante regras predefinidas en contratos intelixentes que se executan nunha blockchain. Estas regras son inmutables e garanten que as decisións se toman de forma colectiva polos membros da organización, quen posúe tokens ou criptomoeda que lles outorgan dereito a voto. As DAO operan sen a necesidade dunha entidade centralizada ou dun intermediario, permitindo que os membros xestionen recursos e tomen decisións de maneira autónoma e transparente. Este modelo descentralizado permite aos membros propoñer, votar e executar accións directamente a través da blockchain, o que fai ás DAOs resistentes á censura e altamente transparentes, á vez que reduce os custos de xestión e os custos operativos.

Os contratos intelixentes son o cerne das aplicacións descentralizadas, un paradigma de computación e provisión de servizos online baseado nas cadeas de bloques.

Unha dApp (*decentralized Application*, aplicación descentralizada) é unha aplicación que se executa nunha rede blockchain e está deseñada para funcionar de forma descentralizada e autónoma, sen necesidade de intermediarios nin servidores centrais, o que reduce o risco de fraude ou manipulación. Nunha dApp, a lóxica de servidor da aplicación impleméntase como un contrato intelixente ou un conxunto de contratos intelixentes despregados nunha rede blockchain. Os usuarios interactúan coa dApp a través dunha interface de usuario, habitualmente unha aplicación web ou unha aplicación móbil (ver figura 4.1).

As dApps representan unha novo paradigma para construír e despregar aplicacións descentralizadas que explotan as vantaxes das redes blockchain para obter unha maior seguridade, transparencia e fiabilidade, en comparación coas aplicacións centralizadas tradicionais. As dApps adoitan ser de código aberto, o que significa que o código está dispoñible publicamente para que calquera poida velo, usalo e modificalo. Ademais, as dApps non están controladas por ningunha autoridade central, o que as fai resistentes á censura e a manipulación. Dada a súa natureza, os requisitos das dApps materialízanse mediante contratos intelixentes, que son visibles e transparentes. As dApps están protexidas pola tecnoloxía blockchain subxacente, que utiliza criptografía para protexer a integridade dos datos e evitar accesos non autorizados.

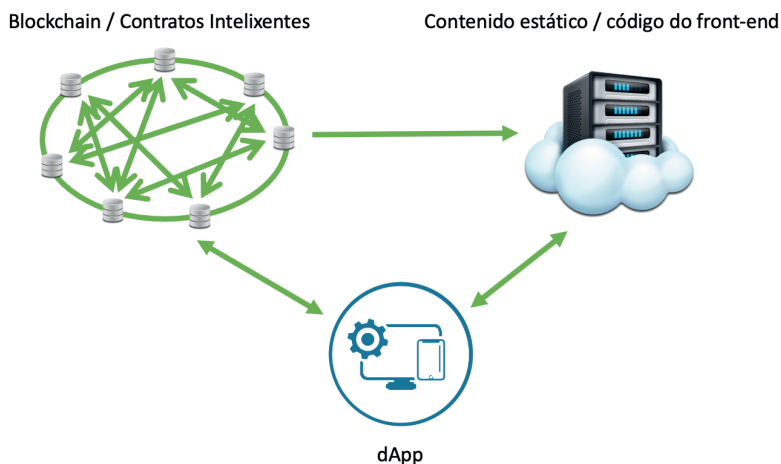


Figura 4.1. Estructura típica dunha dApp. O contido estático e o código da aplicación (p. ex., app móbil ou código JavaScript do cliente) tómasse dun servidor de contidos, mentres que a lóxica impleméntase mediante a orquestración dunha serie de contratos intelixentes nunha blockchain.

Como exemplo de dApp, podemos deseñar unha plataforma de redes sociais completamente descentralizada e que utilice como soporte para a xestión dos datos dos usuarios unha cadea de bloques. Esta rede social aproveitaría as vantaxes da tecnoloxía blockchain para proporcionar unha plataforma segura, transparente e descentralizada para que os usuarios interactúen e compartan contidos. A plataforma permitiría aos usuarios crear perfís, publicar contidos e interactuar con outros usuarios, como calquera outra rede social. Con todo, en lugar de estar controlada por unha autoridade central, a plataforma funcionaría almacenando a información nunha blockchain, o que á súa vez garantiría a seguridade, a transparencia e a inmutabilidade. Cando un usuario crea un perfil na nosa plataforma descentralizada de redes sociais, a súa información pódese guardar na cadea de bloques, o que garante que a información está segura e non pode ser manipulada por ninguén. Cando un usuario crea unha publicación ou interactúa con outro usuario, esta información tamén se almacena na blockchain, o que proporciona un rexistro transparente e inmutable de toda a actividade na plataforma. Dado que a plataforma está descentralizada, non existe unha autoridade central que a controle ou poida censurar contidos. No seu lugar, a plataforma funcionaría mediante un mecanismo de consenso, no que os usuarios deciden colectivamente que contidos son válidos e cales non. Isto garante que a plataforma sexa realmente aberta e accesible a todos os usuarios, independentemente da súa localización xeográfica ou as súas opinións políticas.

As aplicacións no ámbito das finanzas descentralizadas (DeFi, (*Decentralized Finances*)) tamén son bos exemplos de dApps. Trátase dun sector en rápido crecemento que aproveita a tecnoloxía blockchain para ofrecer servizos financeiros sen necesidade de intermediarios tradicionais como os bancos. Podemos imaxinar unha plataforma de préstamos construída sobre unha rede blockchain que opera de forma descentralizada sen o control das entidades financeiras. A plataforma permitiría aos usuarios prestar e tomar prestados activos dixitais como criptomoedas, stablecoins, ou tokens funxibles ou non funxibles como os descritos no capítulo 7. Como vimos no capítulo 3, as stablecoins ou criptomoedas estables son tokens indexados a unha moeda tradicional como o euro ou o dólar, e por tanto estables con relación a estas últimas.

Cando un usuario desexa prestar activos dixitais na plataforma, deposita os seus activos nun contrato intelixente na rede blockchain. O contrato intelixente emparella automaticamente a prestamistas con prestatarios en función dos tipos de interese e as condicións de préstamo desexados. Unha vez concedido o préstamo, o prestatario recibe os activos dixitais e o prestamista recibe os intereses do seu depósito. Dado que a plataforma de préstamos baséase nunha rede blockchain, é transparente e segura. Todas as transaccións e actividades de préstamo rexístranse na cadea de bloques, proporcionando un rexistro transparente e inmutable de todas as actividades de préstamo. Isto garante que a plataforma de préstamos sexa fiable e funcione de forma segura e transparente. Ademais, a plataforma de préstamos opera sen necesidade de intermediarios, o que pode reducir os custos de transacción e proporcionar unha maior accesibilidade aos servizos financeiros. Isto pode ser especialmente beneficioso para particulares e empresas de países en desenvolvemento ou para quen non teñen acceso aos servizos bancarios tradicionais.

De todos os xeitos, as dApps presentan importantes desafíos en termos de privacidade debido á natureza transparente e inmutable das blockchains nas que se constrúen. Nunha blockchain pública, todas as transaccións e datos almacenados son visibles para todos os participantes da rede, o que contrasta co enfoque tradicional das aplicacións centralizadas, onde os datos están almacenados en servidores privados e poden ser controlados ou eliminados por unha entidade central. Esta transparencia inherente ás tecnoloxías descentralizadas pode poñer en risco a privacidade dos usuarios, especialmente cando se manexan datos sensibles como información persoal ou financeira.

Por exemplo, na rede social construída sobre unha blockchain que acabamos de describir, a información persoal dos usuarios (nomes, datas de

nacemento, localizacións, etc.) podería quedar exposta publicamente ou ser accesible para calquera usuario da rede. Aínda que os datos se poden almacenar de forma cifrada, a relación entre as transaccións ou interaccións dos usuarios pode revelar patróns de comportamento, conexións persoais, ou preferencias que comprometan a súa privacidade. Ademais, unha vez que a información se almacena na blockchain, non pode ser alterada nin eliminada, o que dificulta a protección da privacidade e dos datos de carácter persoal a longo prazo. Estes retos obrigan aos desenvolvedores de dApps a buscar solucións avanzadas, como o uso de técnicas criptográficas, redes de privacidade de segunda capa (layer 2) ou enfoques innovadores como as probas de coñecemento cero, para protexer adecuadamente a información dos usuarios mentres se aproveitan os beneficios da tecnoloxía blockchain.

Nos parágrafos seguintes explicamos en liñas xerais como abordar o desenvolvemento de contratos intelixentes e dApps. A continuación, describimos os aspectos relacionados coa seguridade máis relevantes no desenvolvemento de contratos intelixentes.

4.2 Construindo contratos intelixentes e dApps

Os contratos intelixentes constrúense utilizando linguaxes de programación específicos como Solidity, Vyper ou Scilla, ou linguaxes de propósito xeral como Rust, Go ou JavaScript. Cada blockchain que soporta contratos intelixentes promove a súa propia linguaxe de preferencia e mesmo o seu propio modelo de despregamento. Solidity (Chittoda, 2019a) é a linguaxe de programación máis utilizada con diferenza para desenvolver contratos intelixentes, e Ethereum a blockchain máis popular para despregalos. Solidity é similar a JavaScript e está deseñado para ser fácil de aprender e usar. Vyper (Mahmoodi, 2020) é unha linguaxe máis recente, tamén de Ethereum, deseñada para ser máis segura e fácil de auditar que Solidity. Rust (Klabnik y Nichols, 2019) é unha linguaxe de programación de sistemas xa consolidada que está a gañar popularidade para desenvolver contratos intelixentes na blockchain Polkadot. Unha versión modificada da linguaxe Go é a linguaxe de Hyperledger Fabric para desenvolver os seus contratos intelixentes, chamados *chaincodes* (Abdelhady, 2019). Scilla é a linguaxe da cadea de bloques Zilliqa para os seus contratos intelixentes (Zilliqa Team, 2019). É unha linguaxe deseñada para ser máis segura e máis fácil de auditar que Solidity, e inclúe características como a análise estática e a verificación formal.

Os contratos intelixentes, unha vez programados e compilados almacénanse nunha rede blockchain mediante unha transacción específica, o que os fai inmutables e a proba de manipulacións. O código dos contratos intelixentes permite realizar transaccións automatizadas e sen supervisión entre as partes, eliminando a necesidade de intermediarios, como as entidades financeiras ou os reguladores.

En xeral, crear un contrato intelixente implica identificar o problema, elixir unha plataforma blockchain, escribir o código, probar e despregar o contrato e interactuar con el na rede blockchain.

En primeiro lugar trataríase de identificar e caracterizar o problema ou tarefa que o contrato intelixente pretende resolver, é dicir, determinar as regras e condicións que rexerán a execución do contrato. Por exemplo, no contexto da xestión de cadeas de subministración, o noso reto pode consistir en automatizar o seguimento e a verificación das mercadorías a medida que avanzan pola cadea. Podería programarse un contrato intelixente para liberar automaticamente o pago a un provedor unha vez que o comprador reciba e verifique un envío de mercadorías. Unha vez que o contrato coñeza que a entrega da mercadoría foi verificada, o contrato pode realizar as transaccións necesarias para transferir criptomoedas da carteira do comprador ao vendedor.

Unha vez que completamos a fase anterior de caracterización do problema e captura de requisitos, é necesario elixir unha plataforma de cadea de bloques que admita contratos intelixentes, tentando que sexa a máis axeitada ás características do noso caso de uso. Seleccionar a cadea de bloques adecuada para despregar os nosos contratos intelixentes pode ser unha decisión complexa, xa que as distintas cadeas de bloques teñen características e capacidades diferentes que poden ser máis acaídas para determinadas aplicacións. A blockchain elixida debe ter certa reputación de fiabilidade e seguridade para garantir a integridade dos contratos intelixentes e evitar accesos ou modificacións non autorizadas. Debe ter ademais unha estrutura de goberno clara e cumprir con toda a normativa e os estándares aplicables. Por outra banda, debe ser capaz de manexar o número de transaccións e execucións de contratos intelixentes suficiente para a nosa aplicación. A interoperabilidade é outro criterio importante, xa que a cadea de bloques debe ser capaz de integrarse cos sistemas e plataformas requiridos polo noso caso de uso (p, ex., o sistema loxístico existente nunha cadea de subministración ou determinados oráculos dos descritos no capítulo 6). Finalmente, o custo de despregar e executar contratos intelixentes na blockchain debe ser razoable e competitivo en comparación con outras opcións.

Hai diversas formas de interaccionar cunha blockchain. O máis habitual é que non dispoñamos dun nodo da rede blockchain propio, polo que teremos que valernos dalgunha plataforma que nos ofrezca unha interface do programador (API) que nos permita comunicarnos coa cadea elixida sen ter que executar os nosos propios nodos. Estas plataformas tamén adoitan contar con ferramentas de desenvolvemento para o seguimento e análise dos nosos contratos intelixentes.

O seguinte paso consiste en configurar unha contorna de desenvolvemento. Teremos que instalar o software necesario e a configuración dunha carteira para probas e despregamento. Como acabamos de indicar, existen plataformas que proporcionar ferramentas de desenvolvemento sólidas e unha comunidade activa para dar soporte ao desenvolvemento e ao despregamento de contratos intelixentes. Por exemplo, Remix (Towaha, 2019) é un IDE (Contorna de Desenvolvemento Integrada) baseada en web para escribir, probar e despregar contratos intelixentes na rede Ethereum, utilizando calquera navegador. Proporciona unha interface fácil de usar para compilar e despregar contratos intelixentes, e tamén inclúe un marco de probas integrado. Truffle Suite (Chittoda, 2019b), lamentablemente en período de desmantelamento nestes momentos aínda que todavía é posible obter as aplicacións, era outro marco de desenvolvemento para Ethereum que incluía ferramentas para compilar, probar e despregar contratos intelixentes. Ganache (Singh, 2020) era unha blockchain persoal para o desenvolvemento de Ethereum que permitía aos desenvolvedores probar os seus contratos intelixentes nunha contorna local. Hardhat (Infante, 2021) é unha contorna de desenvolvemento para construír e despregar contratos intelixentes en Ethereum, Binance Smart Chain e outras plataformas blockchain. Inclúe unha interface de liña de comandos para xestionar contratos, así como un marco de probas integrado e soporte para o despregamento en redes de proba e redes principais.

Unha vez elixido a contorna de desenvolvemento, estamos en condicións de escribir o código do noso smart contract utilizando unha das linguaxes de programación soportadas pola cadea de bloques. Definiremos as funcións, as interfaces, as estruturas de datos e a lóxica que rexen o comportamento do contrato, de acordo cos requisitos identificados na fase de análise do problema. Para probar e validar o contrato desenvolvido, o máis habitual é utilizar unha blockchain de proba ou *testnet* para asegurarse de que funciona segundo o previsto. Para elo usaremos unha ferramenta como Remix para despregar e interactuar co contrato. As probas nunha rede de proba garanten que o contrato intelixente sexa seguro e estea libre de vulnerabilidades antes de despregalo na rede principal ou *mainnet*. Isto axuda a minimizar o risco de

perdas económicas ou de expoñer datos sensibles debido a problemas de seguridade.

As probas nunha testnet permiten ademais aos desenvolvedores verificar que o contrato intelixente funciona correctamente e cumpre os requisitos do caso de uso previsto, e permite recibir comentarios da comunidade e identificar posibles problemas ou áreas de mellora antes de despregar o contrato na rede principal. Isto axuda a garantir que o contrato funcionará como se espera na rede principal. Por outra banda, probar nunha testnet pode aforrar custos ao evitar as tarifas asociadas ao despregamento e a execución de contratos intelixentes na rede principal.

A modo de exemplo, imos ver como despregar un contrato intelixente sinxelo na testnet Sepolia utilizando o entorno de programación de contratos intelixentes Hardhat. Trátase dun contrato intelixente que simplemente ofrece unha mensaxe almacenada cando se consulta, e permite actualizar devandita mensaxe. O seu código en Solidity sería o seguinte:

```
1 // SPDX-License-Identifier: MIT  hello.sol
2 // Sinxelo contrato intelixente que difunde unha mensaxe e
3 // permite a súa actualización.
4 //
5 // Especifica a versión de Solidity, utilizando o versionado
6 // semántico.
7 // Máis información:
8 // https://docs.soliditylang.org/en/latest/layout-of-source-files.html
9 pragma solidity ^0.8.18;
10
11 // Define un contrato chamado 'BroadcastMessage'.
12 // Un contrato é unha colección de funcións e datos (o seu estado).
13 // Unha vez despregado, o contrato residirá nunha dirección
14 // específica dunha blockchain. No noso caso, nunha testnet de
15 // Ethereum chamada Sepolia.
16 // Máis información:
17 // https://docs.soliditylang.org/en/latest/structure-of-a-contract.html
18 contract BroadcastMessage {
19
20     // Declara unha variable de estado 'mensaxe' de tipo 'string'.
21     // As variables de estado son atributos cuxos valores se gardan
22     // permanentemente no almacenamento do contrato.
23     // A palabra chave 'public' fai que as variables sexan
24     // accesibles desde fóra dun contrato e crea unha función que
25     // outros contratos ou clientes poden chamar para acceder ao valor.
26     string public mensaxe;
27
28     // Similar a moitas linguaxes orientadas a obxectos baseados en
```

```
29 | // clases, un construtor é unha función especial que só se
30 | // executa ao crear o contrato.
31 | // Os construtores utilízanse para inicializar os datos do
32 | // contrato, no noso caso o valor inicial da mensaxe.
33 | // Más información:
34 | // https://docs.soliditylang.org/en/latest/contracts.html
35 | constructor(string memory initMensaxe) {
36 |
37 |     // Acepta un argumento 'initMensaxe' que se usa para o valor
38 |     // inicial do atributo 'mensaxe' do contrato.
39 |     mensaxe = initMensaxe;
40 | }
41 |
42 | // Finalmente, o contrato intelixente ofrece unha función pública
43 | // que ten como parámetro unha cadea de caracteres
44 | // e actualiza o atributo 'mensaxe'.
45 | function actualiza(string memory novaMensaxe) public {
46 |     mensaxe = novaMensaxe;
47 | }
48 | }
```

Unha vez que temos o código fonte do contrato, o compilamos co compilador de Solidity. Ao compilar o código do contrato, obtemos dous produtos:

- O bytecode, ou código en formato binario que é executado pola Máquina Virtual de Ethereum (EVM) aloxada nun dos nodos da blockchain. Este bytecode é o que se desprega na blockchain cando se publica o contrato. A EVM le este bytecode e execútao para procesar as transaccións asociadas ao contrato intelixente.
- A Application Binary Interface (). A ABI dun contrato intelixente é unha representación estandarizada que define como se pode interactuar co contrato na blockchain. Describe as funcións do contrato, os seus parámetros de entrada, os tipos de retorno e os eventos que pode emitir. Esta interface é fundamental porque permite ás aplicacións externas (como as dApps ou as carteiras) e a outros contratos comunicarse correctamente co contrato intelixente sen necesidade de coñecer o seu código fonte. A ABI traduce as funcións do contrato a un formato que a EVM e as ferramentas que interactúan coa blockchain poden entender, como Web3.js ou Ethers.js.

Para despregar o contrato intelixente na testnet Sepolia, necesitamos configurar Hardhat cos detalles sobre a rede necesarios para o devandito despre-

gamento. Esa información recollémola nun ficheiro en JavaScript denominado `hardhat.config.js`:

```

1  //  hc-hello.js
2  // Archivo de configuración de Hardhat para o noso proxecto de
3  // contrato intelixente sinxelo.
4  //
5  // Para despregar un contrato intelixente escrito en Solidity,
6  // necesitamos este arquivo e o arquivo fonte (.sol) do
7  // contrato intelixente.
8  //
9  // Módulos de Node.js utilizados:
10 // - dotenv para dispoñer do arquivo .env para proporcionar
11 //   variables de configuración (liñas VARIABLE = "valor").
12 // - hardhat-toolbox é un plugin que agrupa paquetes e plugins de
13 //   Hardhat de uso común. Moi recomendado para empezar a
14 //   desenvolver con Hardhat.
15 // Entre outras cousas, permite despregar e interactuar con
16 // contratos intelixentes utilizando a biblioteca ethers.js e
17 // o plugin hardhat-ethers.
18 require('dotenv').config();
19 require("@nomicfoundation/hardhat-toolbox");
20
21 // URL da API de Alchemy que utilizaremos para acceder á rede
22 // Sepolia e clave privada da carteira de criptomoedas
23 // que utilizaremos para cubrir os gastos do despregamento.
24 // Tomamos esta información do ficheiro de configuración .env.
25 // O ficheiro .env contén dúas liñas coa forma:
26 //
27 // API_URL = "https://XXXXXXX"
28 // PRIVATE_KEY = "XXXXXXXXXX"
29 const { API_URL, PRIVATE_KEY } = process.env;
30
31 /**
32  * @type import('hardhat/config').HardhatUserConfig
33  */
34
35 module.exports = {
36   solidity: "0.8.18",           // versión mínima de Solidity
37   defaultNetwork: "sepolia",    // rede utilizada (testnet Sepolia)
38   networks: {
39     hardhat: {},
40     sepolia: {                  // datos de conexión a Sepolia:
41       url: API_URL,             // URL da API e clave da cartera
42       accounts: ['0x${PRIVATE_KEY}']
43     }
44   }
45 };

```

Previamente, como non dispoñemos do noso nodo propio na rede Sepolia, obtivemos un punto de acceso á devandita rede a través dunha plataforma de desenvolvemento de blockchain, no noso caso a plataforma Alchemy. Este trámite require que nos rexistremos en Alchemy pero é de balde. O punto de acceso é unha URL que podemos utilizar desde o noso código para interaccionar con Sepolia.

Podemos utilizar o código seguinte en JavaScript para despregar o contrato:

```
1 //  desp-hello.js
2 // Script de despregamento para o contrato intelixente do
3 // exemplo.
4 //
5 // Comando de despregamento:
6 //
7 //     npx hardhat run scripts/deploy.js --network sepolia
8 //
9
10 const { ethers } = require("hardhat");
11
12 async function main() {
13     // Unha ContractFactory é unha abstracción usada para despregar
14     // novos contratos intelixentes, así que broadcastMessage
15     // é unha fábrica para instancias do noso contrato
16     // BroadcastMessage. Cando se usa ContractFactory e Contract
17     // do plugin toolbox de Hardhat, as instancias conéctanse
18     // por defecto ao primeiro asinante (o propietario).
19     // ethers.js (que ofrece o obxecto ethers) é unha librería que
20     // se carga co toolbox de Hardhat.
21     const broadcastMessage =
22         await ethers.getContractFactory("BroadcastMessage");
23
24     // Iniciamos o despregamento. deploy devolve unha promesa que
25     // se resolve a un obxecto Contract (bcm). Este obxecto terá
26     // un método para cada unha das funcións e atributos
27     // visibles do noso contrato intelixente (no noso caso,
28     // o atributo mensaxe e o método actualiza()).
29     // Pasamos a deploy os argumentos do construtor utilizado
30     // (no noso caso, a mensaxe inicial)
31     const bcm = await broadcastMessage.deploy("Mensaxe inicial");
32     console.log("Contrato despregado na dirección:", bcm.target);
33 }
34
35 // Chamada a main() para despregar. Se todo vai ben, saímos.
36 // Se non, sacamos a mensaxe de erro por consola.
37 main()
38     .then(() => process.exit(0))
```

```

38 |     .catch(error => {
39 |         console.error(error);
40 |         process.exit(1);
41 |     });

```

Para terminar con este exemplo, presentamos a continuación un sinxelo programa en JavaScript para interaccionar co noso contrato a través da súa ABI.

```

1 | // Interacción co contrato intelixente creado  int-hello.js
2 | // previamente.
3 | //
4 | // Pide ao usuario que introduza unha mensaxe e actualízaa
5 | // no contrato intelixente BroadcastMessage. Finalmente,
6 | // obtén do contrato intelixente a (nova) mensaxe
7 | // e sácaa por pantalla.
8 | //
9 | // As variables de contorna obtéñense do arquivo .env
10 |
11 | // Dirección do noso contrato intelixente. Obtense ao
12 | // despregalo.
13 | const CONTRACT_ADDRESS = process.env.CONTRACT_ADDRESS;
14 |
15 | // Cargamos a ABI do noso contacto (Application Binary Interface)
16 | // para interactuar con el. A ABI tamén se crea ao despregar o
17 | // contrato.
18 | const contract =
19 |     require("../artifacts/contracts/BroadcastMessage.sol/" +
20 |         "BroadcastMessage.json");
21 |
22 | // Asinante
23 | // Unha conta que ten a capacidade de asinar transaccións
24 | // No noso caso, unha carteira creada con MetaMask
25 | // con fondos da testnet Sepolia suficientes para
26 | // cubrir os gastos da interacción.
27 | const ownerAddress = process.env.PUBLIC_KEY;
28 | const signer = ethers.provider.getSigner(ownerAddress);
29 |
30 | // Contrato
31 | // Un obxecto ethers.js que representa o contrato específico
32 | // despregado na cadea Sepolia (BroadcastMessage)
33 | // Para crear o obxecto, necesitamos a dirección do contrato, o
34 | // ABI e a conta.
35 | const bcContract =
36 |     new ethers.Contract(CONTRACT_ADDRESS, contract.abi, signer);
37 |
38 | // Agora imos ler a mensaxe almacenada no noso contrato

```

```
39 // intelixente e imprimila na consola. Tamén a actualizaremos.
40 // En JavaScript, usamos funcións asíncronas para interactuar con
41 // blockchains.
42
43 console.log("");
44
45 async function main() {
46
47     // En primeiro lugar, imos obter o valor do atributo 'mensaxe'
48
49     const mensaxe = await bcContract.mensaxe();
50     console.log("A mensaxe actual é: " + mensaxe);
51
52     // Agora actualicemos a mensaxe usando o método proporcionado
53     // polo contrato intelixente (actualiza).
54     // Usamos o módulo Node.js prompt-sync para pedir texto pola
55     // consola.
56     var prompt = require('prompt-sync')();
57
58     // Pedimos unha nova mensaxe por consola e
59     // actualizámola no contrato intelixente
60     var novaMensaxe = prompt('Nova mensaxe: ');
61     console.log('Actualizando mensae a ' + novaMensaxe + '...');
62     const tx = await bcContract.actualiza(novaMensaxe);
63
64     // .wait() sobre o obxecto de transacción devolto asegura
65     // que o noso programa espere a que a transacción sexa
66     // forxada na blockchain antes de continuar.
67     // Se omitísemos esta liña, é posible que o novo
68     // valor non estea aínda dispoñible ao solicitalo (pode tardar
69     // uns segundos).
70
71     await tx.wait();
72
73     // Obtemos o novo valor do atributo 'mensaxe'
74
75     const mensaxeActualizada = await bcContract.mensaxe();
76     console.log("A nova mensaxe é: " + mensaxeActualizada);
77 }
78
79 // Imos aló
80
81 main();
```

Unha vez que o contrato funciona do modo esperado na testnet, despregariámolo nunha rede pública de blockchain, por exemplo a mainnet de Ethereum ou de Hyperledger Fabric, utilizando unha ferramenta como Remix ou Truffle.

Teremos que pagar a taxa da transacción correspondente na criptomoeda da rede que elixamos para o despregamento. Unha vez despregado o contrato intelixente, pódese interactuar con el enviándolle transaccións desde unha carteira ou dApp. O contrato executarase automaticamente baseándose na súa lóxica programada, e os resultados rexistraranse na blockchain.

Podemos ver unha dApp como a orquestración dun conxunto de contratos intelixentes para obter unha funcionalidade complexa. Tras elixir a plataforma blockchain e a contorna de desenvolvemento, o noso primeiro obxectivo sería deseñar a arquitectura da dApp. Isto implica definir o modelo de datos, crear ou identificar os contratos intelixentes que necesitamos e decidir o deseño da interface e a experiencia do usuario (UI/UX).

Unha vez que identificamos ou desenvolvemos os contratos intelixentes, debemos construír un interface de usuario ou *front-end* para a dApp. Isto implica crear un deseño UI/UX fácil de usar que permita aos usuarios interactuar coa dApp. Hai varias tecnoloxías populares que os desenvolvedores para construír front-ends para aplicacións descentralizadas. JavaScript é con diferenza a opción máis habitual para iso. Por exemplo, React (Chinnathambi, 2018) é unha popular biblioteca de JavaScript para construír interfaces de usuario. Permite aos desenvolvedores crear compoñentes de interface de usuario reutilizables e proporciona unha forma sinxela e eficiente de xestionar o estado da aplicación. Material UI (Schwarzmüller, 2018) é un conxunto de compoñentes de React que implementan as directrices de Material Design de Google. Proporciona unha serie de compoñentes de interface de usuario, como botóns, formularios, cadros de diálogo etc., que se poden personalizar e personalizar facilmente.

Vue.js (Passaglia, 2017) é outro popular framework JavaScript para a construción de interfaces de usuario. Proporciona unha sintaxe sinxela e intuitiva e ofrece funcións como a vinculación bidireccional de datos e os compoñentes reactivos. Angular (Schwarzmüller, 2021) é un completo framework JavaScript moi utilizado para construír aplicacións web complexas. Ofrece unha ampla gama de funcións para personalizar as aplicacións, interactuar con datos noutras plataformas, etc. Web3.js (Adaś, 2021) é unha biblioteca JavaScript que permite aos desenvolvedores interactuar con contratos intelixentes baseados en Ethereum e construír aplicacións descentralizadas. Proporciona unha forma sinxela e eficiente de interactuar coa rede Ethereum e xestionar transaccións e eventos. Bootstrap (Spurlock, 2021) é un popular marco CSS que proporciona unha serie de compoñentes e estilos de interface de usuario preconfigurados. Pódese combinar coas tecnoloxías anteriores para crear deseños de interface

de usuario responsivos e adaptados a dispositivos móbiles para aplicacións dixitais.

As probas son unha parte fundamental do proceso de desenvolvemento de dApps para verificar que funcionan segundo o previsto. Isto implica probar os contratos intelixentes en busca de erros, probar a interface de usuario para validar a súa usabilidade e probar o rendemento xeral da dApp.

Despois de probar a dApp, o paso final é despregala na blockchain e nos servidores elixidos. Despregaremos os contratos intelixentes na rede blockchain, o front-end no servidor ou nas tendas de apps e poremos a dApp ao dispor dos usuarios.

Do mesmo xeito que ocorre cos contratos intelixentes, existen plataformas que axudan a simplificar o proceso de desenvolvemento de dApps permitindo que os desenvolvedores se concentren en crear a funcionalidade principal das súas aplicacións descentralizadas en lugar de preocuparse pola infraestrutura de blockchain subxacente. Tamén proporcionan ferramentas avanzadas de análise e monitorización para axudar aos desenvolvedores a optimizar o rendemento das súas dApps. Por exemplo, OpenZeppelin é unha biblioteca de contratos intelixentes reutilizables para Ethereum que proporciona unha base segura e probada para construír aplicacións descentralizadas. Inclúe unha interfaz de liña de comandos para despregar contratos en redes Ethereum, así como un IDE en liña para probar e despregar contratos. Outro exemplo é Alchemy, quizá a plataforma de desenvolvemento máis popular no momento de publicar este manual (2024). Trátase dunha plataforma que proporciona aos desenvolvedores as ferramentas e a infraestrutura necesarias para crear dApps en varias redes de cadeas de bloques. Ofrece un conxunto de ferramentas para desenvolvedores, unha API, e ferramentas de depuración e análise para crear, probar e escalar aplicacións. Soporta varias redes, como Ethereum, Binance Smart Chain ou Polygon, entre outras.

4.3 O custo de operar en Ethereum: o gas

O gas no contexto dos contratos intelixentes en Ethereum é un concepto fundamental que se refire ao custo computacional de executar operacións na rede Ethereum. Funciona como unha unidade de medida para a cantidade de traballo computacional requirido para realizar unha transacción ou executar un contrato intelixente.

Cada operación na EVM ten un custo de gas asociado. Este sistema ten como obxectivo previr o spam e os ataques de denegación de servizo xa que, ao requirir unha compensación económica pola computación, evítase que a rede sexa asolagada con transaccións innecesarias ou maliciosas. Tamén serve para limitar a complexidade do código dos contratos intelixentes mediante a definición dun límite máximo de gas dispoñible por bloque, o que asegura que os contratos non poidan consumir recursos infinitos.

Ademais, o gas utilízase para compensar aos mineiros/validadores por procesar as transaccións e manter a rede segura mediante a súa conversión a Ether. O custo total dunha transacción calcúlase multiplicando o gas utilizado polo prezo do gas (en gwei, onde $1 \text{ gwei} = 10^{-9} \text{ Ether}$). Os usuarios poden indicar o prezo en gwei que están dispostos a pagar polo gas, o que afecta á probabilidade de que os mineiros decidan meter dita transacción nun bloque, e polo tanto ao tempo necesario para que a transacción será procesada.

Algúns exemplos do custo de gas para operacións comúns en Ethereum son (Wood, 2024):

- Unha operación de suma simple podería custar ao redor de 3 gas.
- Unha operación de multiplicación serían ao redor de 5 gas.
- Acceder a unha variable de estado tería un custo de ao redor de 200 gas.
- Unha transacción básica de criptomoeda, como por exemplo enviar ETH dunha dirección a outra, 21.000 gas.
- Almacenar un valor no almacenamento do contrato: 20.000 gas.
- Crear un novo contrato: a partir de 21.000 gas (ver máis abaixo), máis o custo do código do contrato.
- Chamar a unha función dun contrato que muda o seu estado: varía segundo a complexidade, pero xeralmente comeza ao redor de 21.000 gas máis o custo das operacións específicas.

Estes custos poden variar dependendo das actualizacións da rede e a complexidade específica das operacións. Para operacións máis complexas, como as involucradas en protocolos DeFi (ver capítulo 8), os custos poden ser significativamente máis altos.

O cálculo do custo en Ether para despregar e executar un contrato intelixente en Ethereum implica varios factores. Para calcular o custo de despregamento dun contrato intelixente temos que considerar:

1. O custo en gas do despregamento, que inclúe un custo base (21.000 gas para a transacción) máis o custo de almacenar o código do contrato na blockchain. Cada byte de código custa 200 gas.
2. O custo da execución do construtor do contrato intelixente. Se o contrato ten un método construtor que se executa durante o despregamento do mesmo, débese engadir o gas utilizado para executar o código de ese construtor.
3. O custo de inicialización de variables de estado. Establecer valores iniciais para as variables de estado do contrato tamén consome gas.

Como indicamos antes, podemos propoñer un prezo para o gas en gwei que estamos dispostos a pagar, que será o que paguemos se a nosa transacción acéptase para ser minada e validada. Para o cálculo final necesitamos multiplicar a cantidade de gas total utilizada polo prezo do gas no momento do despregamento, o que nos dará o custo en Ether. Para coñecer o custo en euros, teremos en conta a cotización do Ether nese momento.

Por exemplo, se un contrato utiliza 500.000 gas para despregarse e o prezo do gas é 20 gwei, o prezo do despregamento será de 0,01 Ether, é dicir, uns 20 euros no verán de 2024.

No caso da execución dun contrato intelixente, os elementos para considerar son:

1. O custo en gas da transacción base, 21.000 gas para calquera transacción que interactúe cun contrato.
2. O custo da execución do código, que dependerá da complexidade da función chamada e das operacións realizadas.
3. O custo de almacenamento. Se a función modifica o estado do contrato, por exemplo actualizando as súas variables persistentes, consumirase gas adicional.

Para o cálculo final de executar un contrato intelixente xa despregado seguimos o mesmo procedemento que no caso do seu despregamento. Por

exemplo, se chamar a unha función utiliza 100.000 gas e o prezo do gas no momento de chamar á función é 15 gwei, teremos un custo en Ether de 0.0015 ETH, ou uns 3 euros no verán de 2024.

Estes custos poden variar significativamente dependendo das condicións da rede e a complexidade específica do contrato ou función en cuestión. O prezo do gas fluctúa segundo a conxestión da rede, e en momentos de alta demanda os prezos poden aumentar significativamente. Por outra banda, algunhas operacións, como liberar almacenamento, poden resultar nun reembolso parcial de gas. Ademais, ao solicitar unha transacción débese especificar un límite de gas para a mesma. Se a execución excede este límite, a transacción falla, pero o gas ata ese punto vaise cobrar igual.

No caso das funcións de tipo **pure** ou **view** que non modifican o estado do contrato, e que por tanto non requiren minar ou validar transaccións, non consumiremos gas ao invocalas. Son funcións que só len datos do contrato intelixente pero non realizan cambios nel. Ao non modificar o estado da blockchain nin consumir gas, estas funcións son especialmente útiles en casos onde só se require acceso a datos sen necesidade de realizar transaccións. A súa capacidade para proporcionar información sen custos adicionais nin efectos secundarios convérteas en ferramentas valiosas para os desenvolvedores. Se temos que implementar unha funcionalidade que non require modificacións de estado, debemos por tanto marcala como **pure** ou **view**. Por exemplo, o contrato seguinte ten unha única función **pure**:

```
1 | pragma solidity ^0.8.18;
2 |
3 | contract MathOperations {
4 |     // Función pure que calcula o cadrado dun número.
5 |     function square(uint256 _number) public pure returns (uint256) {
6 |         return _number * _number;
7 |     }
8 | }
```

E o seguinte unha única función **view**:

```
1 | pragma solidity ^0.8.18;
2 |
3 | contract SimpleStorage {
4 |     uint256 public storedData;
5 |
6 |     // Constructor para inicializar un valor almacenado.
7 |     constructor(uint256 _initialData) {
```

```
8         storedData = _initialData;
9     }
10
11     // Función view que devolve o valor almacenado
12     // na variable de estado.
13     function getStoredData() public view returns (uint256) {
14         return storedData;
15     }
16 }
```

A diferenza entre unhas e outras é que as **view** len datos da blockchain aínda que non poden modificalos, mentres que as **pure** simplemente fan cálculos internos sen utilizar información da blockchain e devolven o resultado. Non interactúan co resto do contrato.

4.4 Contratos intelixentes e criptomoedas

Como vimos ao principio deste manual, a primeira aplicación das cadeas de bloques foi a de dar soporte a un sistema de criptomoedas descentralizado. De feito, moitas cadeas de bloques teñen unicamente esa función. No caso de Ethereum, ademais de dar soporte a contratos intelixentes e dApps, ofrece a súa propia criptomoeda, o Ether (ETH). Ademais de poder facer transferencias e adquirir bens ou servizos coa devandita criptomoeda, como con calquera outra, utilizando unha carteira electrónica, tamén é posible utilizar contratos intelixentes para realizar as devanditas transferencias. Do mesmo xeito que unha conta de criptomoeda na rede Ethereum, os contratos intelixentes na devandita rede tamén se identifican cunha dirección. Do mesmo xeito que podemos facer transferencias de criptomoeda desde ou con destino a unha conta de criptomoeda, tamén podemos facer transferencia desde ou cara a un contrato intelixente utilizando a súa dirección pública.

O exemplo que propoñemos a continuación ilustra como realizar estas transferencias. Trátase dun contrato intelixente que permite acceder a un servizo, previo pago dunha taxa en criptomoeda. O contrato se inicializa cun custo de servizo cando se desprega. A dirección que despregou o contrato convértese no dono do contrato, e será a única dirección capaz de recuperar os fondos almacenados no mesmo. Os usuarios poden enviar Ether ao contrato en calquera momento, representando o pago por un servizo. Neste exemplo, poden facelo de dúas maneiras:

- Enviando criptomoeda directamente. A función **receive** é dunha función especial (función *fallback* ou de alternativa) que xestiona as transaccións entrantes de Ether cando se envía Ether directamente ao contrato sen invocar ningunha das súas funcións públicas ou externas. É dicir, se transferimos fondos ao contrato directamente desde unha carteira, poñendo como dirección do destinatario da transferencia a dirección do contrato, executarase esta función. No noso caso, comproba se o Ether enviado é suficiente para cubrir o custo do servizo, e calquera exceso de pago devólvese ao remitente.
- Mediante a invocación dunha función externa ou pública do contrato declarada como *payable* (pagable). No noso exemplo, definimos a función declarada como payable **payForService**. A palabra chave **payable** na declaración da función permite que a función reciba Ether. No noso exemplo, os usuarios deben enviar polo menos o custo do servizo especificado como Ether para pagar polo servizo. Neste caso, quedamos con toda a criptomoeda que nos envíe a dirección que invocou a función, sen devolver calquera exceso.

A variable **totalCollected** leva a conta do total de Ether recadado das direccións que interactúan co contrato. No noso exemplo, para recuperar os fondos almacenados no contrato, a función **redeemCollectedFunds** permite ao propietario do contrato extraer o total de fondos recadados mediante a transferencia dos devanditos fondos á súa dirección. O contrato utiliza eventos para rexistrar os pagos e os reembolsos de fondos.

```
1 | 📄 ServicePayment.sol  
2 | // SPDX-License-Identifier: MIT  
3 | //  
4 | // Exemplo para ilustrar como un contrato intelixente  
5 | // pode implementar un servizo por unha tarifa (a pagar en  
6 | // Ether), e como o provedor do servizo (é dicir, o  
7 | // propietario do contrato no noso caso), pode recuperar  
8 | // os honorarios cobrados.  
9 | //  
10 | // Trátase dun exemplo simplificado. No mundo real  
11 | // é probable que a lóxica de pago e recuperación sexa  
12 | // máis complexa, con medidas adicionais de seguridade  
13 | // e control de acceso.  
14 | pragma solidity ^0.8.18;  
15 |  
16 | contract ServicePaymentContract {  
17 |
```

```
18 // Variables globais para recoller información sobre o
19 // propietario do contrato, a tarifa de servizo e
20 // o importe cobrado ata o momento.
21 address public owner;
22 uint256 public serviceCost;
23 uint256 public totalCollected;
24
25 // O contrato utiliza eventos para rexistrar pagos
26 // e reembolsos de fondos.
27 // Evento para indicar que algunha dirección pagou
28 // a taxa de servizo.
29 event ServicePaid(address indexed payer, uint256 amount);
30
31 // Evento para indicar que o propietario recuperou o
32 // saldo do contrato
33 event EtherRedeemed(uint256 amount);
34
35 // Cando se crea o contrato, se inicializa o seu propietario.
36 // O creador/propietario tamén establece a taxa de servizo
37 // en Wei.
38 constructor(uint256 cost) {
39     owner = msg.sender;
40     serviceCost = cost;
41 }
42
43 // Para accións dispoñibles só para o propietario do
44 // contrato.
45 modifier onlyOwner() {
46     require(msg.sender == owner,
47         "So o propietario pode facer isto.");
48     -;
49 }
50
51 // Os usuarios poden enviar Ether ao contrato, representando o
52 // pago por un servizo. A función de recepción é unha función
53 // especial (función fallback) que xestiona transaccións de
54 // Ether que non se corresponden coa invocación dunha función
55 // "payable" dun contrato (é dicir, mediante o envío de Ether
56 // ao contrato con "transfer", "send" ou "call").
57 // msg.data ten que estar baleiro, pola contra invocárase
58 // fallback() no seu lugar.
59 // Nótese que tanto receive() como fallback() defínense sen a
60 // palabra chave "function".
61 // No noso caso, a función comproba se o Ether enviado é
62 // suficiente para cubrir o custo do servizo.
63 // Calquera pago en exceso devólvese ao remitente.
64 receive() external payable {
65     require(msg.value >= serviceCost,
66         "Pago insuficiente para acceder ao servizo.");
```

```

67     uint256 excessPayment = msg.value - serviceCost;
68     totalCollected += serviceCost;
69
70     // Emite un evento para rexistrar o pago.
71     emit ServicePaid(msg.sender, serviceCost);
72
73     // Devolve calquera exceso de pago ao remitente.
74     // A función msg.sender.call envía o exceso de pago
75     // en wei (1 Ether = 10^18 wei) á dirección que accede
76     // ao servizo e comproba se a transacción realízouse
77     // correctamente.
78     if (excessPayment > 0) {
79         (bool success, ) = msg.sender.call{value: excessPayment}("");
80         require(success, "Erro ao recuperar o exceso de pago.");
81     }
82 }
83
84 // Esta función encapsula o servizo como unha función pagable,
85 // facendo máis intuitivo para os usuarios pagar polo servizo
86 // que simplemente enviando Ether á dirección do contrato.
87 // O propietario tamén pode recuperar os fondos recadados.
88 //
89 // A palabra clave payable na declaración da función
90 // permite que a función reciba Ether. Os usuarios deben enviar
91 // polo menos o serviceCost en Ether para pagar polo servizo.
92 // A variable totalCollected leva a conta do total
93 // de Ether recadado dos usuarios.
94 function offerServiceForAFee() public payable {
95     require(msg.value >= serviceCost,
96         "Pago insuficiente para acceder ao servizo.");
97     totalCollected += msg.value;
98
99     // Emite un evento para rexistrar o pago.
100    emit ServicePaid(msg.sender, msg.value);
101
102    // A implementación do servizo iría aquí.
103 }
104
105 // Función que permite ao titular do contrato rescatar os
106 // fondos. O propietario do contrato chamaría a esta función
107 // para retirar o saldo de Ether do contrato.
108 function redeemCollectedFunds() public onlyOwner {
109     require(totalCollected > 0, "Non hai fondos.");
110     uint256 amountToRedeem = totalCollected;
111     totalCollected = 0;
112
113     // Transfire o saldo do contrato ao propietario.
114     (bool success, ) = owner.call{value: amountToRedeem}("");
115     require(success, "Erro ao recuperar os fondos.");

```

```
116 |  
117 |         // Emite un evento para rexistrar a transacción.  
118 |         emit EtherRedeemed(amountToRedeem);  
119 |     }  
120 | }
```

En xeral, os contratos intelixentes de Ethereum dispoñen de dúas funcións fallback. A **receive** comentada e outra chamada directamente **fallback**. Que se execute unha ou a outra depende de como se interaccione co contrato intelixente (ver Fig. 4.2):

- A función **receive()** execútase cando o contrato recibe Ether sen ningún dato adicional ou chamada a unha das súas funcións públicas ou externas. Esta función pode utilizarse para manexar transferencias de Ether sinxelas asociadas a accións sinxelas, como simplemente almacenar o Ether entrante ou actualizar o estado do contrato.
- A función **fallback()** execútase cando alguén envía Ether á dirección do contrato xunto con algún dato que non coincide coa invocación de ningunha función do contrato. Serve como *caixón de xastre* ou *función alternativa* para transaccións con datos non válidos ou non recoñecidos, ou mesmo para implementar un comportamento personalizado cando se chama a unha función descoñecida.

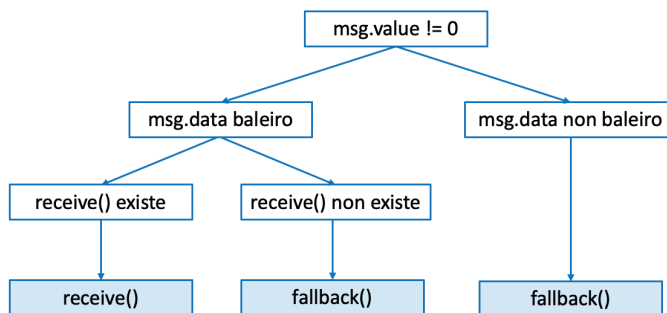


Figura 4.2. Invocación de **fallback()** ou **receive()**.

Na práctica, os desenvolvedores adoitan utilizar a función **receive** para xestionar o Ether enviado directamente á dirección do contrato, que é un caso de uso común para recibir pagos ou doazóns. A función **fallback** utilízase con menos frecuencia e adoita empregarse para fins máis avanzados ou específicos. De todos os xeitos, como a función **fallback** acepta Ether,

esencialmente estende o comportamento de `receive`, xestionando tanto as transferencias de Ether como as transaccións de datos non recoñecidas.

Aparte de realizar transaccións coa criptomoeda nativa de Ethereum, os contratos intelixentes permiten crear outro tipo de tokens funxibles que funcionan de xeito semellante ás criptomoedas. A continuación presentamos un exemplo de cómo podemos utilizar os contratos intelixentes sobre Ethereum para crear o noso propio token, baseado no estándar Ethereum ERC20. O estándar ERC20 (Ethereum Request for Comments número 20) é unha especificación técnica para tokens funxibles en xeral na blockchain de Ethereum. Define un conxunto común de regras e funcións que un token funxible debe implementar para ser compatible co ecosistema máis amplo de Ethereum. Estas regras inclúen métodos para transferir tokens, aprobar gastos por terceiros, ou consultar saldos e subministracións totais, entre outros. Ao adherirse a este estándar, os tokens ERC20 garanten interoperabilidade cunha ampla gama de dApps, carteiras e exchanges de criptomoedas ao comportarse como estas últimas, facilitando a súa integración e uso en todo o ecosistema Ethereum. Esta estandarización foi fundamental para o crecemento do ecosistema de finanzas descentralizadas (DeFi), permitindo unha fácil creación, intercambio e xestión de activos dixitais na rede Ethereum.

Podemos ver a continuación un contrato intelixente para crear un token ERC20 chamado PATACON cuxo símbolo é PAT, coas seguintes características:

- Utiliza a biblioteca OpenZeppelin para implementar a norma ERC20 e a funcionalidade de propiedade do contrato.
- A subministración inicial é de 1 millón de PATACONS.
- Ten unha subministración máxima de 10 millóns de tokens.
- O propietario pode acuñar novos tokens ata alcanzar a subministración máxima.
- Calquera titular de PATACONS pode *queimar* ou destruír os seus propios tokens.

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
5 import "@openzeppelin/contracts/access/Ownable.sol";
```

 patacon.sol

```
6
7 contract PATACON is ERC20, Ownable {
8     // 1 millón de PATACONS, divisibles con 18 decimales
9     // como o propio ETH. A unidade de intercambio
10    // mínima será o "wei" de PATACON (10**-18 PAT)
11    uint256 private constant INITIAL_SUPPLY = 1000000 * 10**18;
12
13    // 10 millóns de PATACONS máximo, subdivisibles en
14    // "weis" de PATACON (pataweis).
15    uint256 public constant MAX_SUPPLY = 10000000 * 10**18;
16
17    // O constructor chama ao constructor do contrato ERC20
18    // que á súa vez crea o token co nome e símbolo
19    // desexados, a despois mina o subministro
20    // inicial da criptomoeda.
21    constructor() ERC20("PATACON", "PAT") {
22        _mint(msg.sender, INITIAL_SUPPLY);
23    }
24
25    // Función para minar máis criptomoeda, ata acabar
26    // o subministro máximo. Só o pode facer o creador da
27    // criptomoeda, quen pode asignarlle os novos PAT
28    // a usuario que queira (pasando a súa dirección
29    // como primeiro parámetro da función)
30    function mint(address to, uint256 amount) public onlyOwner {
31        require(totalSupply() + amount <= MAX_SUPPLY,
32            "Excede o subministro maximo");
33        _mint(to, amount);
34    }
35
36    // Función para destruír criptomoeda. Só se pode destruír
37    // criptomoeda propiedade de quen invoca o contrato
38    // (msg.sender). Utilízase a función _burn que
39    // proporciona o contrato base ERC20.
40    function burn(uint256 amount) public {
41        _burn(msg.sender, amount);
42    }
43
44    // Exemplo de uso da función transfer de ERC20 para
45    // transferir criptomoeda. Neste caso, definimos
46    // unha función especial que só pode executar o creador
47    // da criptomoeda para transferir tokens a un
48    // terceiro. Calquera pode invocar transfer sobre
49    // o seu propio stock de criptomoeda (que recibiu previamente
50    // dalgunha dirección, orixinalmente do creador da
51    // criptomoeda).
52    //
53    // transfer é unha función que proporciona o contrato base
54    // ERC20. Transfire criptomoeda do pool de criptomoeda
```

```
55 | // deste contrato a cantidade desexada (amount) á
56 | // dirección indicada (recipient)
57 | function ownerTransfer(address recipient, uint256 amount)
58 |     public onlyOwner returns (bool) {
59 |         return transfer(recipient, amount);
60 |     }
61 | }
```

O contrato anterior asume que estamos a utilizar a biblioteca OpenZeppelin, que é unha práctica común para implementar tokens ERC20 de maneira segura. O contrato inclúe funcións para acuñar e destruír tokens, o que permite certa flexibilidade na xestión da subministración. Ademáis:

- A función **mint** só pode ser chamada polo propietario do contrato, engadindo unha capa de control. É dicir, o creador do token pode acuñar novas unidades ata completar a subministración definida na súa creación.
- A función **burn** permite a calquera titular de tokens destruír os seus propios tokens, se así o desexan.
- Para transferir PATACONS desde o propietario (owner) a calquera dirección, ou dende calquera dirección que xa dispoña de PATACONS, pódese utilizar a función **transfer** que vén incluída na estándar ERC20. Esta función xa está implementada no contrato base ERC20 de OpenZeppelin que estamos a utilizar, e é recoñecida polas carteiras de criptomoeda habituais.

4.5 Interacción coa blockchain dende un contrato intelixente

Ademáis de poder comunicarse con aplicacións descentralizadas e con outros contratos intelixentes na blockchain, os contratos intelixentes tamén poden interaccionar coa propia cadea de bloques. Para elo, a máquina EVM proporciona tres obxectos que se poden utilizar para obter información da cadea de bloques, chamados **block**, **msg** e **tx**. Estes obxectos proporcionan información sobre a cadea de bloques, sobre a información enviada ao contrato intelixente na transacción activa, e sobre a propia transacción activa. Os atributos máis relevantes destes obxectos aparecen recollidos no cadro 4.1.

Dispoñemos ademais das funcións **blockhash** e **gasleft**:

- **blockhash(uint blockNumber) returns (bytes32)**: devolve o hash do bloque pasado como parámetro, só no caso de que o bloque é un dos 256 bloques máis recentes. Noutro caso devolve 0.
- **gasleft() returns (uint256)**: devolve a cantidade de gas restante na transacción Ethereum actual. Os desenvolvedores poden utilizar esta función para tomar decisións en tempo de execución dentro dos seus contratos intelixentes, dependendo de canto gas quede, como por exemplo realizar certas accións dentro dun contrato só se queda suficiente gas para completalas. Isto asegura que as transaccións non fallen por quedar sen gas antes de tempo, o que pode resultar na perda de Ether e nunha transacción que non se complete cos resultados esperados.

4.6 Aspectos de seguridade

Os contratos intelixentes convertéronse nun obxectivo popular para os atacantes porque poden levar asociados elementos de gran valor e adoitan ser complexos, o que os fai vulnerables a varios tipos de ameazas. Neste apartado describimos os ataques máis relevantes aos contratos intelixentes. O código dos contratos intelixentes pode ser complexo e difícil de analizar, e os atacantes a miúdo poden atopar fallos de programación que lles permitan manipular o estado do contrato ou eludir as medidas de seguridade.

O ataque de reentrada é un dos ataques máis comúns e perigosos contra os contratos intelixentes. Neste tipo de ataque, o atacante explota un fallo no código do contrato que lle permite chamar repetidamente á mesma función antes de que a chamada anterior teña finalizado. Isto pode desencadear, por exemplo, a repetición dunha transferencia de activos varias veces seguidas antes de que a primeira (e supostamente única) delas fose consolidada no libro maior.

Outro ataque común é un dos habituais en enxeñería de software: o desbordamento e subdesbordamento de enteiros. Estes ataques prodúcense cando o código do contrato non valida correctamente os valores de entrada, o que provoca erros matemáticos que poden ser aproveitados polos atacantes para manipular o estado do contrato e conseguir realizar transaccións ilícitas.

Un ataque específico das cadeas de bloques é o chamado *front running*. Prodúcese cando un atacante utiliza o seu coñecemento das transaccións pendentes para colocar a súa propia transacción por diante doutras no con-

Cadro 4.1. Propiedades dos obxectos `block`, `msg` e `tx`.

<code>block.basefee (uint)</code> <code>block.chainid (uint)</code> <code>block.coinbase (address payable)</code> <code>block.difficulty (uint)</code> <code>block.gaslimit (uint)</code> <code>block.number (uint)</code> <code>block.prevrandao (uint)</code> <code>block.timestamp (uint)</code>	Tarifa base do bloque actual. Identificador da cadea actual. Dirección do validador do bloque actual . Dificultade do bloque actual (obsoleto). Compórtase como <code>block.prevrandao</code>. Límite de gas do bloque actual. Número do bloque actual. Número aleatorio proporcionado pola Beacon Chain (ver apéndice D). Timestamp do bloque actual en segundos desde unix epoch (00:00:00 UTC do 1 de xaneiro de 1970).
<code>msg.data (bytes calldata)</code> <code>msg.sender (address)</code> <code>msg.sig (bytes4)</code> <code>msg.value (uint)</code>	Calldata completo da mensaxe actual. Dirección de chamada da mensaxe actual. Catro primeiros bytes da mensaxe (i.e. identificación da función chamada). Cantidade de cripto enviada coa mensaxe en Wei.
<code>tx.gasprice (uint)</code> <code>tx.origin (address)</code>	Prezo da transacción en gas. Orixe da transacción actual.

xunto de transaccións, o que lle permite manipular o estado do contrato e potencialmente conseguir que se comporte de maneira non prevista.

Como todo sistema distribuído, os contratos intelixentes tamén están expostos a ataques de denegación de servizo (DOS), nos que un atacante asolaga o contrato con transaccións ou datos de entrada co fin de sobrecargarlo e facer que deixe de funcionar.

Os contratos intelixentes a miúdo dependen de bibliotecas de terceiros para implementar a funcionalidade desexada, como podemos ver nos exemplos de contratos intelixentes neste manual. Os atacantes poderían crear e distribuír bibliotecas maliciosas deseñadas para explotar vulnerabilidades no código do contrato.

Para mitigar estes e outros posibles ataques, é fundamental que os desenvolvedores revisen e proben coidadosamente o código dos seus contratos intelixentes e apliquen as mellores prácticas de seguridade e xestión de riscos, como a realización de auditorías de código, probas de vulnerabilidades e o uso de prácticas de codificación seguras, como a validación de entradas e o control de acceso. Na mesma liña, é conveniente utilizar aritmética de enteiros segura para evitar vulnerabilidades de desbordamento e subdesbordamento de enteiros. As versións obsoletas de funcións de procesamento de enteiros, en bibliotecas ou contratos externos, son máis vulnerables aos ataques, polo que é mellor evitar o seu uso e utilizar sempre a última versión.

Para evitar os ataques baseados en bibliotecas maliciosas, é importante utilizar sempre que sexa posible bibliotecas de código aberto ben auditadas. Os contratos intelixentes poden ser complexos, e a miúdo é máis seguro utilizar bibliotecas ben probadas, auditadas e amplamente utilizadas para funcións básicas como a criptografía e a serialización de datos. Do mesmo xeito, os contratos intelixentes que dependen de contratos externos son máis vulnerables aos ataques, polo que é mellor minimizar o seu uso e confiar unicamente en contratos externos de confianza e auditados. Tamén é unha boa práctica incluír no código controles de acceso para evitar que usuarios non autorizados accedan a funcións ou datos sensibles, por exemplo un control de acceso baseado en roles e o uso de mecanismos de autenticación adecuados.

Os desenvolvedores de contratos intelixentes deben probar o seu código a fondo antes de despregalo nunha rede activa (mainnet), incluíndo probas unitarias e probas de integración, así como probas baixo diversas condicións de carga, utilizando para iso unha rede de proba de confianza. Deben implemen-

tarse tamén mecanismos sólidos de xestión e rexistro de erros para axudar a identificar e solucionar problemas.

A simplicidade é clave cando se trata do desenvolvemento de contratos intelixentes. Un código complexo pode introducir riscos innecesarios e vulnerabilidades potenciais. Ademais, documentar profusamente o código é esencial para garantir que poida entenderse correctamente e manterse ao longo do tempo.

4.7 Contratos intelixentes e enxeñería do software

Desde o punto de vista da enxeñería do software, desenvolver contratos intelixentes difire significativamente do desenvolvemento de programas para plataformas tradicionais en varios aspectos fundamentais. Unha das diferenzas máis relevantes é que os contratos intelixentes, unha vez despregados nunha blockchain, son inmutables; é dicir, o seu código non pode ser modificado ou actualizado facilmente. Isto contrasta co desenvolvemento tradicional, onde os programas poden ser actualizados, corrixidos ou mellorados regularmente a través de novas versións. Por tanto, os desenvolvedores de contratos intelixentes deben garantir que o código sexa seguro, eficiente e estea libre de erros antes da súa implementación, xa que calquera erro podería ter consecuencias graves, como perdas financeiras ou vulnerabilidades de seguridade irreparables.

Outra diferenza importante é a restrición de recursos na contorna de execución. Os contratos intelixentes execútanse nunha máquina virtual de blockchain, a EVM no caso de Ethereum e das redes baseadas no mesmo software, que ten limitacións de recursos en termos de procesamento, almacenamento e consumo de gas (as tarifas asociadas coa execución de operacións na blockchain das que falábamos no apartado 4.3). Os desenvolvedores deben optimizar o seu código para ser o máis eficiente posible, minimizando o uso de recursos e reducindo custos de execución. A diferenza das plataformas tradicionais, onde os recursos son máis abundantes e os custos de procesamento non están directamente ligados ao uso, no desenvolvemento de contratos intelixentes cada instrución executada ten un custo financeiro asociado, o que engade unha capa adicional de complexidade e require unha maior atención á eficiencia do código.

Para mellorar a eficiencia do código dos contratos intelixentes e limitar o seu custo en gas, existen varias prácticas recomendadas que os desenvolve-

dores poden adoptar. Dado que o gas é o recurso que se consome por cada operación executada na blockchain, optimizar o código pode reducir significativamente o custo de execución. Algúns exemplos de prácticas eficientes serían:

- Evitar os bucles longos. As estruturas de control como bucles (**for**, **while**) poden incrementar rapidamente o custo en gas, especialmente se o número de iteracións é grande ou descoñecido. En lugar de usar bucles, é preferible limitar as operacións que dependen de datos externos, utilizando funcións máis pequenas e simplificadas ou dividindo as operacións en múltiples transaccións.
- Utilizar tipos de datos máis pequenos. Elixir tipos de datos que requiran menos almacenamento cando sexa posible, como **uint8** ou **fuint16** en lugar de **fuint256** que require 32 bytes, pode reducir o uso de almacenamento e o custo de gas. Tamén é útil agrupar variables de tamaño similar para que se almacenen de maneira máis eficiente na memoria da EVM.
- Minimizar o uso de almacenamento na blockchain. O almacenamento de datos na blockchain é unha das operacións máis custosas en termos de gas. En lugar de almacenar moitos datos en variables de estado, recoméndase usar almacenamento non persistente como **memory** ou **calldata** cando sexa posible, xa que son máis baratas. Ademais, para reducir o almacenamento, os desenvolvedores poden utilizar hashes en lugar de gardar grandes cantidades de datos sen procesar.
- Optimizar o uso de funcións externas e internas. As funcións externas son máis caras que as internas. Cando se desenvolven contratos intelixentes, é mellor usar funcións internas (**internal**) en lugar de externas (**external**) sempre que sexa posible, para minimizar os custos de gas asociados coas chamadas de función.
- Evitar a lóxica complexa dentro dos contratos. Manter a lóxica do contrato simple e evitar operacións complexas ou condicionais múltiples pode reducir o custo en gas. Para cálculos intensivos, é preferible realizalos fóra da blockchain e almacenar só os resultados esenciais ou mínimos necesarios no contrato intelixente.
- Eliminar código redundante e non utilizado. É importante revisar e refactorizar o código para eliminar calquera liña de código innecesaria ou

redundante. Cada liña de código contribúe ao tamaño do contrato e, por tanto, ao custo en gas do seu despregamento e execución.

- Aproveitar as bibliotecas e contratos ben optimizados. Utilizar bibliotecas e contratos estándar como os de OpenZeppelin ou ferramentas como Solidity Optimizer para xerar código eficiente pode axudar a evitar erros comúns e mellorar a eficiencia do contrato. Estas bibliotecas xa foron auditadas e optimizadas para reducir o uso de gas.

Aplicando estas prácticas, os desenvolvedores poden minimizar o custo de gas dos seus contratos intelixentes, mellorando a súa eficiencia e reducindo o impacto financeiro das operacións na blockchain.

Capítulo 5

Sistemas de arquivos descentralizados

Os sistemas de arquivos peer-to-peer (P2P) ou sistemas de arquivos descentralizados son sistemas de arquivos distribuídos que permiten aos usuarios compartir arquivos directamente entre si sen depender dun servidor centralizado (Androutsellis-Theotokis y Spinellis, 2004). Constrúense sobre unha rede P2P (ver apéndice E) onde cada usuario achega ao sistema unha parte do seu espazo de almacenamento e do ancho de banda da rede. Os arquivos almacénanse polo xeral en varios nodos da rede. Estes sistemas de almacenamento utilizan algoritmos criptográficos para protexer e cifrar os datos, algoritmos de dispersión para manter a súa integridade e dispoñibilidade, algoritmos de consenso para garantir a coherencia entre nodos e autenticación descentralizada para verificar ás persoas usuarias. Como consecuencia, dispoñemos dunha solución de almacenamento máis robusto e eficiente que os sistemas de arquivos tradicionais, que permite compartir e acceder a datos de maneira segura, rápida e fiable sen depender dun provedor centralizado.

Do mesmo xeito que as redes sobre as que se constrúen, os sistemas de arquivos P2P utilizan polo xeral unha táboa hash distribuída (*distributed hash table*, DHT) para indexar e localizar os arquivos na rede. Cando un usuario quere acceder a un arquivo, pode localizalo a través dalgún nodo que teña unha copia da DHT e descargalo directamente do usuario ou usuarios que o aloxan nese momento.

Os sistemas de arquivos P2P teñen varias vantaxes sobre os sistemas de arquivos cliente-servidor tradicionais, herdadas das redes P2P. Son máis resistentes aos fallos e poden seguir funcionando aínda que algúns nodos da rede se desconecten. Tamén proporcionan velocidades de descarga máis rápidas, xa que os arquivos poden descargarse de múltiples fontes simultaneamente. Ademais, os sistemas de arquivos P2P adoitan garantir mellor a privacidade e seguridade dos datos, xa que os arquivos non se almacenan nun

servidor centralizado que podería ser vulnerable a ataques. Algúns exemplos populares de sistemas de arquivos P2P son BitTorrent (Zhang et al., 2011), eDonkey (Heckmann et al., 2004) e Gnutella (Kan, 2002). Estes sistemas utilízanse habitualmente para compartir arquivos de gran tamaño, como películas, música e software, pero tamén poden empregarse para compartir outros tipos de información, por exemplo conxuntos de datos científicos (datasets) ou copias de seguridade.

Nos apartados seguintes analizaremos brevemente os fundamentos e principios clave destes sistemas de arquivos, así como a súa arquitectura e funcionamento. Centrarémonos no sistema de arquivos IPFS (Interplanetary File System), un sistema de almacenamento descentralizado que se converteu nunha referencia para proporcionar almacenamento persistente no ámbito das cadeas de bloques. Antes de nada, analizaremos brevemente as limitacións dos sistemas de arquivos tradicionais no contexto das cadeas de bloques.

5.1 Limitacións dos sistemas de arquivos tradicionais

Os sistemas de arquivos tradicionais presentan diferenzas clave cos sistemas de arquivos P2P en canto á súa arquitectura e funcionamento. Nos sistemas centralizados os datos almacénanse nun único lugar controlado por unha entidade provedora de servizos (p. ex., un conxunto de servidores ou un servizo de almacenamento na nube), o que implica dependencia desta entidade para a xestión e acceso aos devanditos datos. Estas solucións adoitan ser propietarias e vulnerables a ataques cibernéticos ou perda de información debido a erros humanos ou fallos no hardware. Conceptualmente, un sistema de almacenamento na nube tradicional non deixa de ser un sistema de arquivo centralizado, xa que a xestión da información almacenada depende exclusivamente do provedor de servizos na nube.

Os sistemas de arquivos tradicionais presentan varias limitacións en termos de seguridade, accesibilidade e puntos de fallo en comparación cos sistemas de arquivos descentralizados. Por exemplo, nos sistemas de arquivos centralizados, a xestión do acceso aos recursos realízase mediante permisos e grupos de usuarios. Este sistema de permisos por tipo de usuario é vulnerable a ataques de suplantación ou ataques tipo phishing xa que, se se produce un acceso non autorizado ás credenciais dun usuario, dispónse acceso a toda a información accesible para o devandito usuario, o cal permitiría a un atacante eliminar, modificar, cifrar ou almacenar nova información non autorizada. En cambio, os sistemas de arquivos descentralizados dispoñen de mecanismos

para garantir a inmutabilidade da información almacenada e utilizan técnicas criptográficas de forma nativa para protexer e cifrar os datos, o que reduce drasticamente o risco de roubo ou alteración de información nos casos de suplantación de identidade.

Ademais, os sistemas de arquivos tradicionais adoitan ser propietarios e centralizados, o que significa que unha única entidade controla todo o acceso aos datos almacenados, xeralmente a entidade que opera os servidores onde se almacenan os datos ou o servizo de almacenamento na nube. Isto pode limitar a capacidade das persoas usuarias para compartir ou colaborar libremente en proxectos, especialmente se se trata de organizacións globais con usuarios dispersos xeograficamente. Doutra banda, os sistemas de arquivos descentralizados, debido á arquitectura P2P subxacente, permiten aos usuarios compartir e acceder a datos evitando un control centralizado.

Finalmente, nos sistemas de arquivos tradicionais existen varios puntos de fallo críticos que poden chegar a comprometer a integridade da información de forma catastrófica, como a perda de datos por erros humanos, fallos no hardware ou erros no sistema operativo ou nas aplicacións de xestión da información. Ademais, estes sistemas adoitan ser propensos a ataques cibernéticos e roubo de información debido ás súas vulnerabilidades en seguridade. Os sistemas de arquivos descentralizados reducen drasticamente a probabilidade de perda de datos, xa que estes se distribúen entre os nodos da rede P2P.

5.2 Fundamentos dos sistemas de arquivos descentralizados

Un sistema de arquivos descentralizado baséase en tres principios fundamentais: o direccionamiento por contido, a descentralización e a tolerancia a fallos. Nun sistema de ficheiros tradicional podemos localizar un contido a partir dun identificador de ficheiro e do roteiro de acceso ao devandito ficheiro, mentres que nos sistemas descentralizados localizamos un contido a partir dunha clave única que se calcula a partir do propio contido, de maneira que dous contidos diferentes non poden compartir a mesma clave. Cando queremos obter un contido almacenado, no canto de indicar onde está localizado (roteiro de acceso e nome de ficheiro) indicamos directamente que contido queremos obter.

A descentralización implica que os datos se parten en fragmentos pequenos que se almacenan de maneira distribuída entre múltiples nodos da rede P2P, o que reduce a dependencia do provedor centralizado e aumenta a re-

sistencia a puntos de fallo. Esta distribución de datos entre nodos permite recuperar facilmente calquera fragmento perdido ou danado para reconstruír o arquivo orixinal. A tolerancia a fallos lógrase mediante algoritmos de dispersión e consenso que manteñen a coherencia e consistencia nunha rede descentralizada, evitando conflito entre diferentes versións do mesmo arquivo e preservando a integridade dos datos.

Estes sistemas adoitan utilizar ademais algoritmos de cifrado para protexer e garantir a seguridade dos datos almacenados, reducindo o risco de roubo ou alteración de información. Por outra banda, a autenticación dos usuarios realízase a través de protocolos descentralizados, por exemplo mediante criptografía de clave pública ou firmas dixitais, o que evita depender dun provedor centralizado para a verificación do usuario.

En definitiva, os sistemas de arquivos descentralizados, ao basearse nunha rede P2P onde os datos se almacenan e distribúen entre múltiples nodos, eliminan a dependencia do provedor centralizado e melloran a súa robustez fronte a puntos de fallo, á vez que se dá soporte á replicación automática e transparente para o usuario para garantir a dispoñibilidade do contido mesmo se algúns nodos fallan. A distribución de datos garante que os arquivos se dividan en fragmentos pequenos para o seu almacenamento entre nodos, permitindo recuperar facilmente calquera fragmento perdido ou danado para reconstruír o arquivo orixinal, e os algoritmos de dispersión e consenso manteñen a coherencia e consistencia nunha rede descentralizada, evitando conflitos entre diferentes versións do mesmo arquivo e preservando a integridade dos datos. Podemos dicir que un sistema de arquivos descentralizado é unha solución máis segura, eficiente e resistente a fallos que se basea en principios como a descentralización, distribución de datos e tolerancia a fallos.

5.3 Arquitectura e funcionamento

Unha arquitectura de sistema de arquivos descentralizado baséase nun conxunto de nodos conectados a unha rede P2P. Estes nodos divídense en varios tipos, en función no seu rol dentro da rede, aínda que un nodo concreto pode asumir varios roles:

- Os nodos de almacenamento proporcionan almacenamento físico para almacenar fragmentos de contidos ou proporcionar almacenamento persistente.

- Os nodos de procura manteñen copias das táboas DHT para identificar a localización dos contidos a partir dos seus identificadores.
- Os nodos de direccionamiento dan soporte ao enrutado das solicitudes de contido e a súa entrega desde os nodos de almacenamento aos usuarios.

Como indicamos antes, unha característica fundamental dos sistemas de almacenamento P2P é que a información almacenada identifícase polo seu contido e non pola súa localización, como ocorre nos sistemas de almacenamento tradicionais. O direccionamiento por contido é un proceso mediante o cal se localiza e accede a contidos específicos almacenados na rede e non a determinados arquivos, utilizando o protocolo P2P para atopar roteiros óptimos para transferir datos. As claves para identificar contidos concretos son en realidade claves de hash xeradas a partir do contido dun arquivo ou bloque de datos que queremos almacenar, que se utilizan para identificar e verificar a integridade dos datos almacenados na rede, evitando así posibles problemas como o roubo de datos ou alteración indebida do contido.

En caso de perda de datos, os algoritmos descentralizados identificarán automaticamente e recuperarán os contidos a partir das copias existentes na rede. A replicación e recuperación de datos baséanse na colaboración entre nodos da rede para garantir unha maior resistencia a fallos, seguridade e acceso rápido a contidos previamente almacenados. Por outra banda, os algoritmos descentralizados non permiten cambios no contido dun arquivo unha vez que foi almacenado, o que garante a inmutabilidade e integridade.

Un compoñente fundamental que asegura a integridade dos datos almacenados en sistemas descentralizados é a proba de custodia. Esta proba permite verificar que un provedor de almacenamento continúa mantendo e custodiando os datos na súa posesión sen necesidade de acceder a eles directamente. A proba de custodia (Proof of Custody ou PoC) é un mecanismo de seguridade utilizado en redes de blockchain e sistemas de almacenamento descentralizado para garantir que os nodos ou participantes que almacenan datos realmente os teñen e están a manter unha copia válida e accesible destes. Este concepto é chave para evitar fraudes ou manipulacións en sistemas onde os datos son distribuídos entre moitos nodos.

En termos simples, a proba de custodia require que os nodos demostren, de forma periódica e sen necesidade de revelar o contido dos datos, que aínda posúen e están a custodiar os datos que se lles asignaron. Este proceso lévese a cabo mediante técnicas criptográficas que permiten probar que o nodo ten

acceso e control sobre os datos que lle foron confiados sen ter que revelar o contido almacenado. Para elo, o sistema envía un desafío criptográfico a un nodo e o nodo debe responder cunha proba válida que confirme a posesión do dato, o que permite asegurar que o nodo non eliminou ou alterou os datos que está obrigado a custodiar, todo elo sen revelar o contido exacto do dato, senón só a proba criptográfica que confirma a súa posesión.

Por outra banda, a proba de replicación (Proof of Replication, PoRep) é un mecanismo criptográfico utilizado nestes sistemas para garantir que un provedor de almacenamento está a almacenar múltiples copias físicas independentes dos datos solicitados. A diferenza da proba de espazo-tempo (Proof of Space-Time), que se enfoca en demostrar que os datos foron almacenados durante un período de tempo específico, a proba de replicación asegura que o provedor non estea só a facer referencia aos mesmos datos duplicados, senón que efectivamente realizou unha réplica física independente dos mesmos. Este tipo de proba é fundamental para garantir que os provedores de almacenamento cumpren cos requisitos de redundancia, mantendo múltiples copias dos datos de maneira distribuída, o que mellora a durabilidade e resistencia dos arquivos almacenados na rede.

Na práctica, o provedor de almacenamento debe demostrar de forma continua e verificable que cada réplica é única e está realmente almacenada no seu hardware, evitando fraudes onde os provedores simplemente pretendan ter múltiples copias. Este proceso é necesario para que os sistemas de almacenamento descentralizado sexan confiables e seguros, dándolle aos usuarios unha garantía de que os seus datos estarán protexidos e sempre accesibles a través de múltiples réplicas distribuídas na rede.

5.4 InterPlanetary File System

O sistema de arquivos interplanetario (Interplanetary File System, IPFS) é un protocolo e unha rede deseñados para crear un método peer-to-peer para almacenar e compartir contido de calquera tipo nun sistema de arquivos descentralizado, sendo o seu obxectivo principal o de mellorar a eficiencia, resistencia a fallos e privacidade dos sistemas de arquivos tradicionais. Foi deseñado inicialmente por Juan Benet (Benet, 2014), e agora é un proxecto de código aberto desenvolvido con axuda da comunidade.

IPFS está deseñado para substituír á arquitectura tradicional cliente-servidor da World Wide Web, creando un método distribuído e descentralizado de com-

partir arquivos. IPFS é compatible con calquera tipo de arquivo e permite o acceso rápido a contidos previamente almacenados na rede, independentemente da súa localización xeográfica ou a capacidade do nodo local. Este sistema ofrece unha alternativa descentralizada e eficiente aos sistemas centralizados tradicionais como FTP, HTTP ou SMB.

En lugar de recuperar contidos dun único servidor, IPFS recupéraos dunha rede de pares que almacenan os mesmos contidos. Isto crea un sistema máis sólido e resistente á censura, xa que o contido non está ligado a un único punto de fallo. Para iso, IPFS utiliza un sistema de direccionamiento único, denominado almacenamento direccionable por contido ou *content-addressable storage* (CAS). En IPFS, cada contido recibe un hash único que se utiliza como dirección na rede. Cando un usuario solicita un contido, IPFS busca outros nodos da rede que teñan o mesmo contido e recupérao deles. Isto tamén significa que se un contido xa está na rede, non se duplicará, o que aforra espazo e reduce o uso de ancho de banda.

Cando se envía un arquivo a un nodo IPFS, o nodo almacena unha copia do arquivo no seu almacenamento local e crea unha clave de hash do contido, denominada CID (*content identifier*), que representa o contido do arquivo. Este hash pode utilizarse para recuperar o arquivo desde calquera outro nodo IPFS da rede. O fragmento de código seguinte ilustra o almacenamento dun ficheiro nun nodo IPFS.

```
1 // A función almacenaFicheiroIPFS toma dous argumentos:  ipfs.js
2 // a dirección do nodo IPFS (por exemplo, 'http://localhost:5001')
3 // e o nome do arquivo (por exemplo, 'exemplo.txt').
4 //
5 // A función crea unha instancia de cliente IPFS utilizando a
6 // dirección de nodo pasada como parámetro. Despois utiliza
7 // o método ipfs.add para agregar o arquivo a IPFS. O CID (hash
8 // identificativo) do arquivo almacenado móstrase na consola se
9 // a operación ten éxito.
10 //
11 // Un exemplo de chamada sería:
12 //
13 // almacenaFicheiroIPFS('http://localhost:5001', 'exemplo.txt');
14 //
15 // Utilizamos o módulo ipfs-http-client e o módulo fs para manexar
16 // ficheiros. Para instalar ipfs-http-client:
17 //
18 // npm install --save ipfs-http-client
19 //
20 const clienteIPFS = require('ipfs-http-client');
```



```
21 const fs = require('fs');
22
23 async function almacenaFicheiroIPFS(dirNodoIPFS, ficheiro) {
24   // crear unha instancia de cliente IPFS. O constructor recibe
25   // a URI do nodo utilizado.
26   const ipfs = clienteIPFS(dirNodoIPFS);
27
28   try {
29     // engade o arquivo a IPFS
30     const result = await ipfs.add({
31       path: ficheiro,
32       content: fs.createReadStream(ficheiro)
33     });
34
35     // saca por pantalla o CID do ficheiro engadido
36     console.log('Arquivo almacenado en IPFS con CID:\n'
37       `${result.cid.toString()}`);
38   } catch (error) {
39     console.error('Erro ao engadir arquivo a IPFS: ${error}');
40   }
41 }
```

O nodo IPFS tamén divide o arquivo en anacos máis pequenos, chamados bloques, e crea un grafo acíclico dirixido de Merkle (DAG, ver apéndice C) que representa a estrutura do arquivo. Cada bloque do DAG de Merkle identifícase co seu propio hash, e os hashes dos bloques utilízanse para crear o hash da raíz do grafo, que á súa vez é o identificador único ou CID do arquivo. A continuación, o nodo IPFS transmite o CID do arquivo a outros nodos, permitíndolles descubrir e recuperar o arquivo cando o soliciten. O arquivo almacénase de forma distribuída na rede IPFS, o que significa que poden almacenarse varias copias do arquivo en diferentes nodos, incrementando así a redundancia e a tolerancia a fallos. Cando un usuario solicita un arquivo pola súa clave hash (é dicir, polo seu CID), o cliente IPFS envía unha solicitude do arquivo á rede IPFS. A continuación, a rede IPFS utiliza o CID para localizar os nodos que teñen copias dos bloques nos que dividiu o arquivo para entregalos de volta ao cliente IPFS do usuario. Entón, o cliente volve a ensamblar o arquivo a partir dos bloques devoltos pola rede e preséntallo ao usuario. Os diferentes bloques nos que se divide un arquivo poden estar almacenados en nodos diferentes.

En resumo, ao enviar un arquivo a un nodo IPFS créase un identificador único de contido para o arquivo e almacénase de forma distribuída na rede IPFS, o que permite recuperalo facilmente desde calquera outro nodo da rede. De todos os xeitos, en IPFS tamén é posible ancorar contidos, é dicir, garantir que un contido específico, normalmente un arquivo ou un conxunto de arquivos, se

almacene de forma permanente ou semipermanente e está dispoñible nun nodo da rede IPFS concreto. Cando un contido está ancorado (*pinned*), significa que o nodo retén unha copia dese contido, impedindo que o recolector de lixo de IPFS o elimine. IPFS dispón dun mecanismo de recollida de lixo (*garbage collection*) para recuperar espazo de almacenamento eliminando o contido que non se utiliza ou ao que non se fai referencia. Ancorar contido asegura que non será enviado ao lixo, o que permite que sexa accesible de forma fiable e permanente. Pódense ancorar e desancorar contidos manualmente, segundo se desexe, e algúns nodos IPFS tamén admiten o ancorado automático ou dinámico en función de patróns ou políticas de uso. Como o ancorado ten custos de almacenamento e ancho de banda asociados, algúns servizos de ancorado poden cobrar unha taxa por prestar os devanditos servizos de almacenamento persistente.

IPFS é unha potente ferramenta para o desenvolvemento da Web3, xa que facilita a creación dunha web distribuída e descentralizada, máis resistente, rápida e segura que as arquitecturas cliente-servidor tradicionais. Ten moitos usos posibles, como crear un método máis eficaz e seguro de compartir arquivos de gran tamaño ou mesmo unha versión da internet resistente á censura.

5.5 IPFS e as cadeas de bloques: casos de uso

As cadeas de bloques están deseñadas principalmente para manexar transaccións rápidas e eficientes en lugar de almacenar grandes volumes de datos de maneira persistente, polo que, se se utilizan como sistemas de arquivos, a capacidade de procesamento e o rendemento diminuirán significativamente. En consecuencia, non se considera unha boa idea almacenar grandes cantidades de información persistente nunha cadea de bloques debido aos custos computacionais (ver apartado 4.3), a diminución da eficiencia e o problema de escalabilidade que iso conleva. No seu lugar, é preferible utilizar sistemas descentralizados como IPFS, que están deseñados especificamente para almacenar e xestionar arquivos de gran tamaño de maneira descentralizada, eficiente e escalable. Ademais, ao basearse no direccionamento por contido, é posible garantir a inmutabilidade dos datos se así se desexa.

Por iso, IPFS e blockchain adoitan utilizarse xuntos para crear aplicacións descentralizadas que aproveitan os puntos fortes de ambas as tecnoloxías. Pode utilizarse para almacenar grandes cantidades de datos, incluídos arquivos inmutables, e proporciona unha forma de direccionar estes datos mediante

un hash único que pode utilizarse para recuperar os datos desde calquera nodo da rede. O ancorado permite ademais dispoñer de contidos nun nodo IPFS concreto, mellorando a dispoñibilidade e o tempo de acceso. Blockchain, pola súa banda, permite crear un libro de contabilidade distribuído que pode utilizarse para almacenar e verificar datos de forma segura e inmutable. Combinando estas tecnoloxías, os desenvolvedores poden crear aplicacións descentralizadas capaces de almacenar e compartir datos de gran tamaño de forma segura e resistente á censura. Por exemplo, unha plataforma de redes sociais Web3 baseada nunha cadea de bloques podería utilizar IPFS para almacenar contidos xerados polos usuarios, como fotos e vídeos, á vez que utiliza a cadea de bloques para verificar a autenticidade dos contidos e proporcionar un método descentralizado de verificación da identidade dos usuarios.

Outro caso de uso é a creación dunha plataforma descentralizada de intercambio de arquivos de calquera tipo (imaxes, clips multimedia, aplicacións informáticas, historiais médicos, ...), na que IPFS pode utilizarse para almacenar os arquivos, mentres que a cadea de bloques se utiliza para xestionar as transaccións entre usuarios, garantindo a rastrexabilidade total das devanditas transaccións, e en xeral todas as propiedades asociadas á tecnoloxía de cadeas de bloques.

IPFS tamén se utiliza a miúdo xunto cos tokens non funxibles ou NFT que veremos no capítulo 7 como unha forma de almacenar o activo dixital ou os metadatos asociados ao NFT. Ao almacenar o activo ou os metadatos en IPFS, o NFT vólvese descentralizado, inmutable e máis seguro. Un modo habitual de utilizar IPFS con NFT consiste en almacenar os metadatos en IPFS e, a continuación, crear un NFT que faga referencia ao CID de IPFS. Deste xeito, calquera que posúa o NFT pode utilizar o hash para recuperar os metadatos subxacentes desde IPFS, e a partir dos metadatos o propio activo, mesmo se o creador orixinal do NFT xa non está dispoñible. O activo dixital tamén se pode almacenar en IPFS. Por exemplo, tal como indicamos no capítulo 7, un artista dixital pode crear unha obra de arte única e almacenala en IPFS, xunto con metadatos sobre a obra como o título, o nome do artista, a data de creación, e o propio CID da obra de arte. A continuación, podería crear un NFT que faga referencia ao CID dos metadatos, que á súa vez inclúe o CID da obra, o que permitiría a posibles compradores verificar a autenticidade e a propiedade da obra de arte.

Filecoin é unha rede de almacenamento descentralizado baseada no protocolo IPFS que ofrece servizos de almacenamento e acceso a datos na nube

sen depender dun provedor centralizado (Bauer, 2022b). Filecoin dispón da súa propia cadea de bloques para garantir a integridade, seguridade e transparencia do proceso de almacenamento e reembolso de recursos por parte dos nodos que participan na rede. En Filecoin, os usuarios poden almacenar arquivos no sistema utilizando pagando cunha criptomoeda propia (o FIL) para compensar aos provedores de almacenamento que ofrecen espazo nos seus dispositivos. Estes provedores encárganse de manter a dispoñibilidade e persistencia dos datos, garantindo o seu acceso rápido e seguro para todos os usuarios. A rede Filecoin é eficiente, resistente a fallos e ofrece unha alternativa descentralizada ao modelo centralizado de servizos de almacenamento na nube como Amazon S3 ou Google Cloud Storage. Grazas á súa base de código aberto e comunidade globalmente dispersa, Filecoin promove o acceso universal a datos e a creación de aplicacións descentralizadas para unha variedade de escenarios, desde contido multimedia ata rexistros de propiedade dixital.

Sia (Simplified Internet Access) é unha plataforma descentralizada de almacenamento na nube que utiliza blockchain e contratos intelixentes para crear un mercado para provedores e usuarios de almacenamento (Austria et al., 2021). Dun xeito semellante a Filecoin, os usuarios poden alugar espazo de almacenamento a hosts da rede utilizando a criptomoeda Siacoin (SC) como criptomoeda nativa.

5.6 Outros sistemas de arquivos descentralizados

Ademais de IPFS e dos sistemas de almacenamento descentralizado comentados no apartado anterior, existen varias iniciativas comerciais para explotar o concepto de almacenamento descentralizado, cada unha co seu propio modelo de negocio e enfoque de almacenamento distribuído e de recuperación de datos. Swarm, Storj e Madsafe estarían entre os sistemas de arquivos descentralizados comerciais máis destacados.

Swarm (Hartman, Murdock y Spalink, 1999) é unha plataforma de almacenamento e distribución de contidos descentralizada desenvolvida como parte do ecosistema Ethereum. Permite o aloxamento descentralizado de sitios web e o almacenamento distribuído de arquivos. Swarm está deseñado para integrarse con Ethereum, o que lle permite xestionar contidos dirixidos por aplicacións baseadas en Ethereum.

Storj (Figueiredo et al., 2021) é un sistema de almacenamento descentralizado baseado no protocolo IPFS que busca proporcionar unha solución

eficiente e segura para o almacenamento, acceso e compartición de datos. Utiliza un enfoque diferente a Filecoin, xa que non requiren de provedores de espazo de almacenamento dedicados; en cambio, calquera usuario cunha conexión a Internet pode converterse en provedor de almacenamento e recibir recompensas por iso. Os datos divídense en bloques ou *shards* que se distribúen entre nodos da rede, garantindo así unha maior resistencia a fallos e seguridade contra ataques cibernéticos ou perda de información. A comunicación entre os usuarios e provedores de almacenamento encríptase para protexer a privacidade e o contido do arquivo.

A rede Maidsafe (Paul, Hutchison e Irvine, 2014) pretende crear unha Internet descentralizada e autónoma combinando un sistema descentralizado de almacenamento de arquivos cunha rede de comunicación entre pares. Está deseñada para ser autogestionable, autorreparable e resistente á censura.

Estas solucións ofrecen unha alternativa descentralizada a servizos como Dropbox ou Google Drive, permitindo un acceso rápido e seguro a datos previamente almacenados na rede sen depender de provedores centralizados. Ambos dispoñen dunha comunidade e usuarios dedicada ao desenvolvemento de ferramentas e solucións descentralizadas para unha variedade de escenarios, desde aloxamento web ata intercambio de arquivos multimedia.

Capítulo 6

Oráculos

Na tecnoloxía blockchain, un oráculo é un servizo de terceiros que proporciona datos ou eventos externos a un contrato intelixente nunha blockchain. Os oráculos permiten aos contratos intelixentes interactuar co mundo exterior e executar accións baseadas en eventos do mundo real.

Os oráculos (ver figura 6.1) son contratos intelixentes que actúan como intermediarios entre outros contratos intelixentes e as fontes de datos externas, proporcionando ao contrato intelixente datos como prezos, informes meteorolóxicos ou resultados deportivos. Tamén poden proporcionar datos sobre eventos noutras blockchains ou datos que non están dispoñibles de forma nativa na blockchain, como datos financeiros fóra da cadea ou datos de sensores do Internet das cousas (IoT).

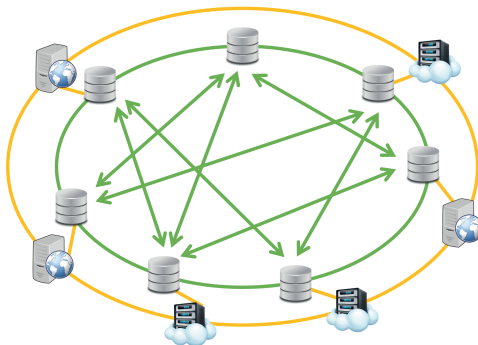


Figura 6.1. Exemplo de rede blockchain á que se conectan oráculos a algún dos seus nodos. Devanditos nodos proporcionan interfaces para o programador de aplicacións (API) para acceder á información. O acceso á información realízase mediante contratos intelixentes.

Os sistemas de oráculos poden clasificarse a grandes liñas de acordo con dous criterios en función da súa arquitectura e o seu propósito: oráculos basea-

dos en software ou en hardware, dependendo da orixe dos datos xestionados; e oráculos centralizados ou descentralizados, dependendo de se o oráculo monitoriza unha soa entidade ou se busca obter datos de múltiples fontes independentes. Cada tipo ten as súas propias características, casos de uso e diferenzas.

Os oráculos baseados en software son programas informáticos que proporcionan datos aos contratos intelixentes. Poden extraer información de fontes en liña, APIs, bases de datos, etc. Son flexibles e poden adaptarse facilmente a diferentes casos de uso. Con todo, o seu principal desafío é garantir a integridade e fiabilidade dos datos que proporcionan. Tipicamente utilízanse para realizar predicións de mercados financeiros, proporcionar información meteorolóxica, taxas de cambio, etc. Por exemplo, se un contrato intelixente necesita coñecer o prezo actual do Bitcoin para executar unha operación, pode facer uso dun oráculo baseado en software que extraia esa información de múltiples plataformas de criptomoedas para pasarlle ao contrato. Normalmente accédese a eles a través dunha interface de programador (API) desde calquera aplicación. Algúns exemplos serían CoinGecko API¹ (información sobre prezos, volume e capitalización de criptomoedas), Alpha Vantage API² (datos financeiros, incluíndo taxas de cambio, prezos de accións, e datos de indicadores técnicos) ou OpenWeatherMap API³ (datos meteorolóxicos e información do clima en tempo real).

Os oráculos baseados en hardware utilizan sensores físicos ou dispositivos de hardware para proporcionar datos do mundo real sobre variables como a temperatura, humidade, localización xeográfica, etc. Ofrecen unha fonte de datos máis directa e confiable que os oráculos software, pero poden ser máis custosos de implementar e manter en comparación cos oráculos baseados en software. Parte do ecosistema da internet das Cousas (IoT), son habituais na monitorización ambiental, control de cadeas de frío, ou en aplicacións de seguridade e vixilancia. Por exemplo, nun sistema de vixilancia ambiental para unha cadea de subministración de produtos perecedoiros, poderíase utilizar un sensor de temperatura conectado á blockchain a través dun oráculo baseado en hardware. O sensor podería xerar información en tempo real sobre a temperatura dun contedor de carga e enviar eses datos á blockchain, onde un contrato intelixente implementaría a lóxica correspondente baseada nestes

¹www.coingecko.com

²www.alphavantage.co

³openweathermap.org

datos, por exemplo para activar unha alerta se a temperatura supera certo limiar.

Os oráculos centralizados obteñen datos dunha única fonte central, que pode ser unha empresa, unha interface centralizada, ou calquera entidade que proporcione información. Son máis simples de implementar pero poden ser vulnerables a ataques e manipulacións, xa que dependen dunha soa entidade. Podémoslos atopar proporcionando prezos de activos, información financeira centralizada, eventos específicos dunha fonte fiable, etc.

Os oráculos descentralizados obteñen datos de múltiples fontes independentes, co que se reduce a dependencia dunha soa entidade e mellórase a resistencia á manipulación. Por tanto, ofrecen unha maior robustez e resistencia á censura que os centralizados, xa que non dependen dunha soa fonte. Son habituais na monitorización de datos de procesos electorais, monitorización de resultados deportivos en sistemas de apostas, ou na provisión de información descentralizada de mercados financeiros. Dentro dos oráculos descentralizados, teriamos os oráculos baseados en consenso, que obteñen datos de varias fontes e utilizan algoritmos de consenso para determinar o valor correcto e garantir a integridade dos datos en contornas onde a información pode cambiar de maneira rápida ou inesperada.

Un exemplo dun oráculo descentralizado é Chainlink (ver o apartado 6.2), unha plataforma que facilita a conexión entre contratos intelixentes e fontes de datos externas de maneira descentralizada (Bauer, 2022a). Chainlink utiliza unha rede de nodos oráculo para proporcionar información do mundo real a contratos intelixentes en diversas blockchains. Estes nodos comparten e validan a información utilizando contratos intelixentes na propia rede Chainlink, asegurando un consenso descentralizado. Cando un contrato intelixente necesita coñecer un dato dos monitorizados por Chainlink (p. ex., a taxa de cambio de determinada criptomoeda), envía unha solicitude á rede Chainlink á que responden varios nodos oráculo coa información solicitada. A resposta compárase e utilízase un algoritmo ou unha función de consenso (p. ex., o cálculo da mediana dos valores devoltos) para determinar o valor final que se proporcionará ao contrato intelixente.

Os oráculos son importantes nas aplicacións de blockchain que requiren datos externos para activar a execución de contratos intelixentes, como en moitas aplicacións DeFi (ver capítulo 8), os mercados de predición e os sistemas de xestión da cadea de subministración. Ao proporcionar unha vía para que os contratos intelixentes accedan a datos externos, os oráculos permiten o desenvolvemento de aplicacións blockchain máis complexas e sofisticadas.

6.1 Arquitecturas de oráculos

Os dous modelos principais de arquitecturas de oráculos son o modelo solicitude-resposta (modelo *pull*) e o modelo publicación-subscrición (modelo *push*). Cada un ten as súas propias características e adáptase a diferentes casos de uso no contexto das blockchains e os contratos intelixentes.

No modelo solicitude-resposta, tamén coñecido como *pull*, o contrato intelixente realiza activamente unha solicitude para obter datos do oráculo cando os necesita, é dicir, cando se activa unha condición específica no contrato. O oráculo responde á solicitude proporcionando os datos solicitados.

Con este modelo, o contrato intelixente ten un maior control sobre cando e que datos solicitar, polo que resulta máis eficiente xa que os recursos do oráculo utilízanse só cando se necesitan datos. Con todo, pode haber certo atraso entre a activación do contrato e a obtención de datos, xa que a solicitude realízase só cando se considera necesaria sen coñecer o estado da fonte de datos.

No modelo publicación-subscrición, tamén coñecido como *push*, o oráculo proporciona activamente datos ao contrato intelixente cando ocorren eventos definidos con anterioridade. O contrato intelixente subscríbese previamente a certos eventos ou tipos de datos proporcionados polo oráculo. A partir dese momento, o oráculo envía datos automaticamente ao contrato cando se producen os eventos de interese, sen que o contrato o solicite activamente, utilizando unha función de *callback* definida durante o proceso de subscrición.

Neste modelo, os datos poden chegar de inmediato ao contrato tan pronto como estean dispoñibles, á vez que se reduce a complexidade ao non ser necesario que o contrato intelixente inicie activamente peticións de datos. Con todo, os oráculos baseados neste modelo poden consumir máis recursos, xa que están continuamente enviando datos, mesmo se non son necesarios.

A elección entre os modelos *pull* e *push* dependerá do caso de uso específico e dos requisitos do contrato intelixente (p. ex., frecuencia de actualización requirida polo caso de uso, importancia da inmediatez dos datos ou requisitos de eficiencia no uso de recursos na rede blockchain). Algúns casos de uso poden beneficiarse da inmediatez do modelo baseado en subscrición, mentres que outros poden preferir o control e a eficiencia do modelo baseado en peticións e respostas. Por exemplo, o uso do modelo *pull* é habitual na obtención de taxas de cambio ou información do clima, mentres que o modelo *push* é

máis apropiado para a monitorización continua de sensores IoT ou cando se necesitan actualizacións en tempo real de eventos específicos.

6.2 Oráculos e finanzas descentralizadas

Vexamos como un oráculo pode proporcionar prezos de accións a un contrato intelixente nunha plataforma de finanzas descentralizadas (ver capítulo 8) para crear e negociar activos sintéticos que seguen o prezo das accións. En finanzas, un activo sintético é un instrumento financeiro que imita o valor doutro activo, as accións no noso caso, sen necesidade de posuír directamente o activo subxacente. Neste exemplo, un servizo de oráculo encargaríase de proporcionar ao contrato intelixente datos sobre o prezo das accións en tempo real procedentes dunha fonte de datos de confianza, como por exemplo unha bolsa de valores. O contrato intelixente utilizaría estes datos para calcular o prezo do activo sintético e executar operacións entre usuarios.

Por exemplo, supoñamos que desexamos crear un activo sintético que siga o prezo das accións de Tesla. Para iso, enviaríamos unha solicitude ao contrato intelixente, que á súa vez enviaría unha solicitude ao oráculo para coñecer o prezo actual das accións de Tesla. O oráculo proporcionaría ao contrato intelixente o prezo actual das accións, e o contrato intelixente utilizaría estes datos para calcular o valor do activo sintético. Se alguén desexara negociar este activo sintético, o contrato intelixente volvería solicitar o prezo actual das accións ao oráculo para garantir que a operación se executa ao prezo de mercado en cada momento. Unha vez executada a operación, o contrato intelixente actualizará a propiedade do activo sintético e transferirá os fondos involucrados na operación.

Neste exemplo, o oráculo actúa como unha fonte fiable de datos sobre o prezo das accións en tempo real que permite ao contrato intelixente fixar con precisión o prezo e executar operacións con activos sintéticos que seguen o prezo das accións. Isto permite negociar activos sintéticos que representan activos do mundo real, sen depender de intermediarios centralizados ou institucións financeiras tradicionais.

A continuación ofrecemos un exemplo sinxelo de contrato intelixente que contacta cun oráculo para obter a taxa de conversión entre Bitcoin (BTC) e o dólar estadounidense (USD). Para iso utilizamos a plataforma Chainlink, que entre outros servizos ofrece oráculos (data feeds) para aplicacións DeFi.

Para acceder a estas fontes de datos, utilizamos oráculos despregados por Chainlink en diversas redes.

```

1 // SPDX-License-Identifier: MIT  test-oracle.sol
2 // Contrato intelixente para probar un oráculo
3 // (data feed) de Chainlink.
4 // O oráculo proporciona a taxa de conversión BTC/USD.
5 //
6 // Versión de Solidity utilizada
7 pragma solidity ^0.8.20;
8
9 // Importa unha interface de Chainlink chamada AggregatorV3Interface.
10 // As interfaces definen funcións sen a súa implementación, o que
11 // deixa aos contratos que implementan a interface que definan
12 // a implementación eles mesmos.
13 // Neste caso, AggregatorV3Interface define, entre outras funcións,
14 // a sinatura da función latestRoundData que imos utilizar.
15 import "@chainlink/contracts/src/v0.8/" +
16     "interfaces/AggregatorV3Interface.sol";
17
18 // Usaremos a función toString do kit de ferramentas Hardhat para
19 // converter os datos devoltos polo oráculo (a través do seu
20 // contrato intelixente proxy) en cadeas de caracteres,
21 // para con iso construír a cadea JSON devolta como resultado
22 // polo noso contrato intelixente.
23 import "@openzeppelin/contracts/utils/Strings.sol";
24
25 // Definimos o contrato
26 contract BTCUSDRateJSON {
27
28     // Declaramos unha variable para almacenar a dirección do
29     // oráculo Chainlink
30     AggregatorV3Interface internal _priceFeed;
31
32     // O construtor do noso contrato simplemente invoca o
33     // construtor do oráculo. Este, á súa vez, só establece
34     // a dirección do oráculo Chainlink na rede Sepolia.
35     //
36     // Se inicializa un obxecto de interface chamado _priceFeed
37     // que utiliza AggregatorV3Interface para conectarse ao
38     // oráculo que ofrece a taxa de conversión BTC/USD,
39     // xa despregado na dirección
40     // 0x1b44F3514812d835EB1BDB0acB33d3fA3351Ee43
41     // da rede Sepolia.
42     //
43     // A interface permite ao noso contrato executar funcións
44     // nese contrato proxy xa despregado.
45     constructor() {

```

```

46 // Pódese substituír a dirección pola dirección de
47 // calquera outro proxy de calquera outro
48 // data feed de Chainlink.
49 // Esta é a dirección do contrato de oráculo na testnet
50 // Sepolia que proporciona taxas de conversión BTC / USD.
51 _priceFeed = AggregatorV3Interface(
52     0x1b44F3514812d835EB1BDB0acB33d3fA3351Ee43
53 );
54 }
55
56 // Función para devolver o tipo de cambio BTC/USD.
57 // Esta función utiliza o obxecto _priceFeed
58 // para executar a función latestRoundData() do oráculo.
59 // Cando o contrato se despreza, inicializa o obxecto
60 // _priceFeed para que apunte ao contrato intelixente en
61 // 0x1b44F3514812d835EB1BDB0acB33d3fA3351Ee43 a través
62 // do construtor (ver máis arriba).
63 //
64 // A función devolve unha cadea JSON con toda a información.
65 function getConversionRateBTCUSD() public view
66 returns (string memory) {
67     // Obtemos a última taxa de conversión a través do oráculo
68     // chamando á función latestRoundData de _priceFeed.
69     // Esta función devolve a información de conversión fixada
70     // na última rolda dispoñible.
71     (
72         uint80 roundID,
73         int256 price,
74         uint256 startedAt,
75         uint256 timeStamp,
76         uint80 answeredInRound
77     ) = _priceFeed.latestRoundData();
78
79     // A resposta do agregador inclúe varias variables:
80     // IDE da rolda onde se fixou o valor; taxa de conversión;
81     // marca temporal de cando comezou a rolda; marca de
82     // cando se actualizou a rolda; IDE da rolda na que se
83     // calculou a taxa.
84     // Usámolas para construír un rexistro JSON e devolveo.
85     string memory jsonString =
86         string.concat('{ "RoundID": "',
87             Strings.toString(roundID),
88             '", "Taxa": "',
89             Strings.toString(uint256(price)),
90             '", "Comezo": "',
91             Strings.toString(startedAt),
92             '", "TimeStamp": "',
93             Strings.toString(timeStamp),
94             '", "naRolda": "',

```

```

95         Strings.toString(answeredInRound),
96         '"}');
97
98     return jsonString;
99 }
100 }

```

Este contrato ofrece unha función `getConversionRateBTCUSD` que devolve unha cadea JSON con toda a información relevante sobre a taxa de conversión: o seu valor e cando e como se fixou dita taxa. A continuación ofrecemos un pequeno programa en JavaScript para interaccionar co devandito contrato e sacar por pantalla a información da taxa de conversión.

```

1                                      interact-json.js
2 // Interacción co contrato intelixente de conversión de tipos
3 // de cambio comentado anteriormente.
4 //
5 // Devolve o tipo de conversión BTC/USD,
6 // xunto con outra información.
7 //
8 // Para executar: npx hardhat run scripts/interacct.js
9
10 // Dirección do noso contrato intelixente, obtida tras o
11 // despregamento e almacenada no arquivo .env
12 const CONTRACT_ADDRESS = process.env.CONTRACT_ADDRESS_JSON;
13
14 // Toma a ABI (Application Binary Interface) do noso contrato
15 // para interactuar con el.
16 const contract =
17     require("test-oracle.sol/BTCUSDRateJSON.json");
18
19 // Asinante
20 // Unha carteira (wallet) que ten a capacidade de asinar
21 // transaccións.
22 // No noso caso, creada con MetaMask.
23 // Para identificala, necesitamos a clave pública da carteira
24 // (de .env).
25 // Para poder pagar con esta carteira, o arquivo
26 // hardhat.config.js ten información sobre a clave privada
27 // e a rede utilizada (Sepolia testnet no noso caso).
28 const ownerAddress = process.env.PUBLIC_KEY;
29 const signer = ethers.provider.getSigner(ownerAddress);
30
31 // Contrato
32 // Un obxecto da biblioteca ethers.js que representa
33 // un contrato específico despregado na cadea.

```

```
34 // No noso caso, o noso contrato BTCUSDRateJSON
35 // Para identificalo, necesitamos a dirección do contrato,
36 // a ABI e a carteira.
37 const financeContract =
38   new ethers.Contract(CONTRACT_ADDRESS, contract.abi, signer);
39
40 // Agora imos contactar co noso contrato intelixente para
41 // obter o tipo de conversión BTC/USD xunto coa información
42 // adicional proporcionada polo oráculo.
43 // Como xa vimos noutro exemplo, en JavaScript, usamos funcións
44 // asíncronas para interactuar con redes blockchain.
45 async function main() {
46
47   try {
48     // Obtén o último valor da taxa de conversión BTC/USD.
49     // O método devolve unha cadea JSON.
50     const presentValue =
51       await financeContract.getConversionRateBTCUSD();
52
53     // Imos parsear (converter) a cadea JSON a un obxecto.
54     const valObj = JSON.parse(presentValue);
55
56     // Mostramos a información por consola utilizando o obxecto
57     // que acabamos de crear. Acondicionamos os valores devoltos
58     // para facelos máis lexibles aos humanos: convertemos o
59     // prezo a dólares e os timestamps a datas e horas.
60     console.log("ID da rolda: " +
61       valObj.RoundID);
62     console.log("Taxa de conversión: " +
63       valObj.Taxa / 10**8 + " USD/BTC");
64     console.log("Comezo: " +
65       new Date(valObj.Comezo * 1000));
66     console.log("Finalización: " +
67       new Date(valObj.TimeStamp * 1000));
68     console.log("Rolda final: " +
69       valObj.naRolda);
70   } catch (error) {
71     // Non deberíamos chegar aquí.
72     console.log("Houbo un error...");
73     console.error(error);
74   }
75
76 }
77
78 // Imos!
79
80 main();
```

Se executamos este programa con:

```
npx hardhat run scripts/interact.js -network sepolia
```

obtemos por exemplo:

IDE da rolda:	18446744073709553674
Taxa de conversión:	28399.39341 USD/BTC
Comezo:	Sun Apr 02 2023 12:18:00 GMT+0200
Finalización:	Sun Apr 02 2023 12:18:00 GMT+0200
Rolda final:	18446744073709553674

6.3 Os oráculos e a xeración de números aleatorios

Xerar números verdadeiramente aleatorios nun contrato intelixente non é unha tarefa sinxela, xa que os contratos intelixentes en redes blockchain operan nunha contorna determinista, o que significa que dadas as mesmas entradas e condicións, produciranse sempre os mesmos resultados. Este determinismo é un aspecto fundamental da tecnoloxía blockchain, xa que por exemplo garante que todos os nodos da rede poidan chegar a un consenso sobre o estado da cadea de bloques. Con todo, a verdadeira aleatoriedade é, por definición, impredecible, o que entra en conflito coa natureza determinista das cadeas de bloques e por tanto dos contratos intelixentes. Para complicar aínda máis as cousas, os contratos intelixentes teñen un acceso limitado a información externa, e non poden interactuar directamente con fontes de datos fóra da cadea ou co mundo físico. De feito, esta característica é a verdadeira razón de ser dos oráculos.

En consecuencia, os xeradores de números pseudoaleatorios que utilizan a información dispoñible nun contrato intelixente como semente, poden ser predicibles se un atacante coñece o valor da semente ou o estado interno da cadea de bloques, e por tanto poderían ser vulnerables á manipulación. Ademais, a xeración de números pseudoaleatorios é comparativamente custosa computacionalmente. O custo de executar cálculos complexos nunha blockchain terá que ser cuberto polo creador ou os usuarios do contrato, polo que implementar unha xeración de números pseudoaleatorios complexa e custosa pode facer que os contratos intelixentes sexan prohibitivamente caros de usar.

Un método habitual para xerar números aleatorios nunha cadea de bloques consiste en xerar un hash dunha secuencia de bytes composta por certos valores extraídos da cadea de bloques, en principio difíciles de predicir, por

exemplo o número de bloque, a marca temporal do bloque, o valor de dificultade do minado do mecanismo de consenso de proba de traballo, o valor de randao (ver apéndice D) xerado polos validadores de bloques no mecanismo de proba de participación (Kalinin y Ryan, 2021), a cantidade de gas remanente no contrato, ou a dirección que invocou o contrato.

O código seguinte xera un número pseudoaleatorio entre 1 e 100 utilizando tres dos valores anteriores nunha cadea Ethereum:

```
1 | pragma solidity ^0.8.18;
2 |
3 | contract RandomNumberGenerator {
4 |     function generateRandomNumber() public view returns (uint) {
5 |         // Obtemos a marca de tempo deste bloque e prevrandao.
6 |         // prevrandao é o valor de Randao da última actualización
7 |         // de Randao, é dicir, a última ranura na que alguén
8 |         // actualizou realmente o Randao e produciu un novo bloque.
9 |         // Tamén usamos a dirección que invoca este contrato.
10 |        uint seed = uint(keccak256(abi.encodePacked(
11 |            block.timestamp,
12 |            block.prevrandao,
13 |            msg.sender)));
14 |
15 |        // Xeramos un número pseudoaleatorio entre 1 e 100.
16 |        uint randomNum = (seed % 100) + 1;
17 |
18 |        return randomNum;
19 |    }
20 | }
```

No exemplo anterior, o uso de `block.timestamp` pode ser controlado polos validadores ata certo punto, xa que aínda que a marca de tempo do bloque actual debe ser estritamente maior que a marca de tempo do último bloque, a única garantía que podemos ter é que o seu valor estará nalgún punto entre as marcas de tempo de dous bloques consecutivos na cadea canónica. Un validador malicioso podería facerlle tomar calquera valor sempre que se cumpra a condición anterior. Do mesmo xeito, `block.prevrandao` é manipulable en certa medida xa que un validador malicioso puiden omitir un bloque se o valor que resulta non lle é favorable, aínda que pode ser unha solución interesante en moitas aplicacións porque se xera mediante un algoritmo criptograficamente seguro e é único para cada bloque.

Debido a estes problemas, moitos desenvolvedores e proxectos de cadeas de bloques recorren a servizos de oráculo externos, como a Función Aleatoria

Verifiable (Verifiable Random Function, VRF) de Chainlink (Blaut, Ma y Wolter, 2023). VRF baséase nun proceso criptográfico para xerar un número do que se pode garantir o seu aleatoriedade. O proceso comeza cun valor semente, proporcionado polo usuario, o contrato intelixente, ou a aplicación que solicita o número aleatorio. Ao facer a solicitude, o nodo Chainlink xera un número aleatorio fóra da cadea de bloques e se *compromete* co devandito número realizando un cálculo coñecido como compromiso criptográfico. Este compromiso permanece oculto e ninguén coñece o valor real do número aleatorio nesta fase máis aló do nodo que o xerou.

Tras un período de espera indicado polo contrato que solicitou o número aleatorio (p. ex., tras completarse o minado dun determinado número de bloques ou tras que ocorra determinado evento), o nodo Chainlink revela publicamente o número aleatorio comprometido xunto cunha proba criptográfica que serve para garantir que o número tense xerado correctamente. O contrato intelixente ou a aplicación que solicitou o número aleatorio comproba a proba criptográfica para verificar que o número revelado xerouse de forma xusta e segura, e que o número é impredecible e non foi manipulado. As técnicas criptográficas utilizadas para o compromiso e a xeración de probas adoitan implicar algoritmos como a criptografía de curva elíptica (ECC) e o uso de probas de coñecemento cero. Estes métodos criptográficos están ben establecidos e son amplamente recoñecidos pola súa seguridade e fiabilidade.

Na práctica, para xerar un número aleatorio instanciamos un contrato envoltorio do xerador VRF. Por exemplo, para xerar números aleatorios sen necesidade de subscribirse a Chainlink pagándoos por adiantado, instanciaríamos o contrato **VRFV2WrapperConsumerBase**. Despois de financiar o noso contrato e inicializar algunhas variables (p. ex., cantos números desexamos xerar, o gas máximo que estamos dispostos a pagar ao VRF para que nos devolva os números, o número de confirmacións de bloque que VRF ten que esperar, ou certas direccións utilizadas por VRF), lanzariamos un procedemento asíncrono para completar a xeración:

1. Chamariamos á función **requestRandomness** do VRF para solicitar os números. Dita función devolve inmediatamente un identificador de solicitude.
2. Cando o xerador VRF termine a súa tarefa, invocará a función de callback **fulfillRandomWords** pasándolle como parámetros o identificador de solicitude da fase anterior e unha cadea en memoria cos números aleatorios xerados. Nós reescribiremos o código da devandita función no noso

contrato para utilizar os números calculados da maneira que queiramos. O máis habitual é gardar eses valores ou pasalos a outras funcións do noso contrato e/o emitir un evento indicando que os números están dispoñibles.

O fragmento de código seguinte ilustra este proceso:

```
1  function resetRandomSeed() external {
2      // Pide a VRFV2WrapperConsumerBase unha palabra aleatoria.
3      // Parámetros:
4      //   - callbackGasLimit: A cantidade máxima de gas a pagar
5      //   para completar a función callback de VRF. Se non é
6      //   suficiente, reverterase.
7      //   - requestConfirmations: 0 número de confirmacións
8      //   de bloque que o servizo VRF esperará para responder.
9      //   - numwords: 0 número de números aleatorios a solicitar
10     //   (1 no noso caso).
11     // A función devolve un requestId.
12     // Este contrato paga ao wrapper o prezo da solicitude.
13     uint256 requestId = requestRandomness(
14         callbackGasLimit,
15         requestConfirmations,
16         1
17     );
18
19     // Chamada á función do wrapper calculateRequestPrice
20     // para estimar o custo total da transacción
21     // e inicializar o IDE da petición.
22     // O resto dos campos RequestStatus se inicializaron
23     // ao crear o contrato.
24     _request.request = requestId;
25     _request.paid =
26         VRF_V2_WRAPPER.calculateRequestPrice(callbackGasLimit);
27 }
28
29 // Función de callback. O wrapper a chama cando terminou,
30 // pasando como parámetros o _requestId correspondente e a lista
31 // de palabras aleatorias xeradas (unha soa palabra no noso caso).
32 // Neste exemplo, recibe un número aleatorio e almacénalo en _nonce.
33 // En xeral, se sobreescribiría para implementar a lóxica para
34 // manexar os valores aleatorios despois de ser devoltos ao contrato.
35 function fulfillRandomWords(uint256 _requestId,
36     uint256[] memory _randomWords) internal override {
37     require(_request.paid > 0, "Solicitude VRF non atopada.");
38     _request.fulfilled = true;
39     _request.randomSeed = _randomWords[0];
40     _nonce = _randomWords[0];
```

```
41 |  
42 |         emit randomSeedAvailable(_requestId,  
43 |                                 _request.randomSeed, _request.paid);  
44 |     }
```

A xeración de números aleatorios non predicibles ou reproducibles é crucial para situacións nas que a imparcialidade e a imprevisibilidade son esenciais, como nos xogos, as apostas e outras aplicacións como a xestión de cadeas de subministracións, as aplicacións estatísticas ou certas aplicacións de finanzas descentralizadas.

6.4 Redes de oráculos e oráculos entre cadeas

Unha rede de oráculos é un sistema descentralizado que proporciona aos contratos intelixentes nunha blockchain acceso a fontes de datos e servizos externos. Os contratos intelixentes adoitan estar limitados aos datos e eventos que están dispoñibles dentro da propia rede blockchain. As redes de oráculos son redes descentralizadas, tipicamente P2P, que proporcionan aos contratos intelixentes acceso a fontes de datos externas a través de oráculos, como feeds de prezos, datos meteorolóxicos e outra información fóra da cadea, actuando como intermediarios entre a blockchain e o mundo exterior. As redes de oráculos utilizan unha arquitectura descentralizada que garante a seguridade e fiabilidade dos datos que proporcionan. Isto significa que os datos obtidos pola rede de oráculos son validados e verificados por múltiples nodos da propia rede, reducindo o risco de datos fraudulentos ou inexactos.

O uso de redes de oráculos permite aos contratos intelixentes interactuar co mundo real de forma segura e fiable, o que os fai máis potentes e versátiles. De feito, existe xa un número importante de redes de oráculos asociadas ás redes blockchain máis populares. Como vimos nos exemplos anteriores, Chainlink é unha rede de oráculos descentralizada que permite aos contratos intelixentes acceder a fontes de datos fóra da cadea, API e outros recursos. É o servizo de oráculos máis utilizado en Ethereum e soporta unha ampla gama de fontes de datos e protocolos.

Band Protocol⁴ é outra rede de oráculos descentralizada que permite aos contratos intelixentes acceder a datos fóra da cadea. Utiliza un mecanismo de consenso de proba delegada de participación (DPoS) e admite unha serie

⁴www.bandprotocol.com

de fontes de datos, como prezos de criptomoedas, datos meteorolóxicos e resultados deportivos.

Binance Smart Chain Oracle⁵ é unha rede de oráculos que proporciona acceso seguro e fiable a datos fóra da cadea para contratos intelixentes en Binance Smart Chain. Outras redes de oráculos populares son TRON Oracle na rede TRON⁶ ou MakerDAO e Uniswap⁷ nos mercados de criptomoedas.

Os oráculos entre cadeas, tamén coñecidos como oráculos intercadena ou *cross-chain oracles* en inglés, son sistemas deseñados para proporcionar información e facilitar a interoperabilidade entre diferentes cadeas de bloques. En esencia, nun contexto fragmentado con múltiples blockchains onde cada unha opera independentemente das demais, estes oráculos permiten que ditas blockchains se comuniquen entre si e compartan datos do mundo real ou información sobre eventos específicos, por exemplo información sobre activos, estados de contratos, eventos etc.

Os oráculos entre cadeas proporcionan unha capa de interoperabilidade entre blockchains, permitindo que unha blockchain obteña e utilice información doutra blockchain de maneira segura e confiable. Esta comunicación pode ser bidireccional, é dicir, que a información flúa en ambas as direccións entre as cadeas, o cal é esencial para aplicacións que requiren coordinación entre múltiples blockchains e onde a seguridade e a confianza son aspectos críticos. Os oráculos entre cadeas deben garantir que os datos transferidos entre blockchains sexan precisos e non manipulados.

Estes oráculos pódense utilizar, por exemplo, para a transferencia segura de activos dixitais entre diferentes blockchains ou para despregar contratos intelixentes interoperables que operan en diferentes blockchains. Para lograr unha interoperabilidade efectiva, a miúdo utilízanse estándares e protocolos específicos de ponte que permiten a interoperabilidade entre blockchains, como os desenvolvidos polas iniciativas Polkadot ou Cosmos (Schulte et al., 2019).

A implementación exitosa de oráculos entre cadeas é esencial para o desenvolvemento de ecosistemas blockchain máis amplos e conectados, permitindo que as aplicacións e activos dixitais móvanse sen problemas a través de diferentes blockchains de maneira segura e confiable.

⁵www.bnbchain.org

⁶trondao.org

⁷app.uniswap.org

6.5 Aspectos de seguridade

Aínda que os oráculos desempeñan un papel importante á hora de permitir que as aplicacións descentralizadas interactúen con datos do mundo real, tamén introducen varios retos de seguridade e potenciais vulnerabilidades. Os oráculos poden ser vulnerables á manipulación de datos, xa sexa a través dun nodo comprometido ou dun actor malicioso que proporcione datos falsos. Isto pode dar lugar a que se proporcionen datos inexactos a un contrato intelixente, dando lugar a unha execución incorrecta do contrato. Os oráculos centralizados poden converterse ademais nun único punto de fallo, o que significa que se o oráculo falla, toda a aplicación blockchain pode verse afectada. Isto pode dar lugar a importantes perdas financeiras ou outras consecuencias negativas.

Os oráculos poden ser vulnerables a accesos non autorizados, xa sexa a través dun nodo comprometido ou a través dunha conexión insegura. Isto pode dar lugar a que datos sensibles queden expostos ou sexan manipulados por partes non autorizadas. Por outra banda, os oráculos poden utilizarse como vector de ataque para explotar vulnerabilidades nos contratos intelixentes. Por exemplo, un actor malicioso podería utilizar un oráculo para proporcionar datos falsos que activen código vulnerable no contrato intelixente, o que lle permitiría roubar fondos ou executar accións non autorizadas. Os oráculos poden ser vulnerables tamén a ataques de rede como os ataques distribuídos de denegación de servizo (DDoS), que poden interromper a capacidade do oráculo para proporcionar datos precisos.

Para mitigar estes problemas de seguridade, é importante aplicar medidas de seguridade adecuadas cando se traballa con oráculos, como a implementación de múltiples oráculos para ter redundancia, o uso de oráculos descentralizados, o cifrado de datos e conexións, a realización de auditorías periódicas e o uso de mecanismos seguros de autenticación e autorización. Tamén é importante considerar coidadosamente os riscos e as posibles vulnerabilidades asociadas aos oráculos á hora de deseñar e implementar aplicacións descentralizadas.

Noutras palabras, traballar con oráculos na tecnoloxía blockchain require seguir certas boas prácticas para garantir a fiabilidade, seguridade e integridade dos datos que se proporcionan ao contrato intelixente. Á hora de seleccionar un provedor de oráculos, é importante elixir un cun historial probado de subministración de datos fiables e precisos. O provedor elixido debería contar tamén con mecanismos transparentes para verificar a autenticidade e exactitude

dos datos que proporciona. Para garantir a fiabilidade e seguridade dos datos proporcionados, recoméndase mesmo utilizar múltiples oráculos que proporcionen datos redundantes. Isto pode axudar a previr calquera punto único de fallo ou manipulación por parte dun único oráculo. É importante implementar mecanismos de xestión de erros e de retroceso (*rollback*) no contrato intelixente para manexar situacións nas que os datos proporcionados polo oráculo son inexactos ou non están dispoñibles.

Os oráculos descentralizados ou redes de oráculos que dependen dunha rede de nodos independentes proporcionan maior seguridade e resistencia á manipulación que os oráculos centralizados. Os oráculos descentralizados tamén proporcionan un maior nivel de transparencia e responsabilidade, xa que os datos ofrecidos son verificados por múltiples fontes independentes.

Para evitar manipulacións e garantir a autenticidade dos datos, recoméndase utilizar métodos criptográficos como firmas dixitais ou funcións hash para verificar os datos que proporciona o oráculo. Ademais, as auditorías periódicas do provedor do oráculo e do contrato intelixente poden axudar a garantir a fiabilidade e seguridade dos datos que se proporcionan. As auditorías tamén poden axudar a identificar calquera vulnerabilidade ou debilidade potencial no oráculo ou o contrato intelixente.

Capítulo 7

Tokens non funxibles

Un token non funxible (non-funxible token ou NFT) é un tipo de activo dixital que representa un artigo ou contido único, como unha obra de arte, un obxecto coleccionable ou un ben inmovible virtual no metaverso. A diferenza dos tokens funxibles, como as criptomoedas, que son intercambiabes entre si e sempre teñen o mesmo valor intrínseco (todos os bitcoins valen o mesmo, por exemplo), cada NFT é único e ten un valor distinto. Os NFT son un fenómeno característico da tecnoloxía blockchain que permite a creación e propiedade de activos dixitais únicos que anteriormente eran difíciles de verificar e monetizar.

No mundo da arte dixital, os NFT revolucionaron a forma en que os artistas poden monetizar e autenticar as súas obras. Permiten aos creadores vender pezas únicas directamente aos coleccionistas, garantindo a propiedade e a escaseza dixital, o que por outra banda aumenta o valor das obras. Isto abriu novas oportunidades para artistas emerxentes e tamén para artistas máis recoñecidos, permitíndolles alcanzar un público global sen necesidade de acudir aos intermediarios tradicionais como as galerías ou os tratantes de arte.

Na industria dos videoxogos, os NFT utilízanse para representar elementos únicos dentro dos xogos, como personaxes, armas ou terreos virtuais. Isto permite aos xogadores posuír verdadeiramente os seus activos de xogo, podendo comercializalos ou transferilos entre diferentes plataformas. Algúns xogos baseados en blockchain construíron economías enteiras ao redor destes activos dixitais únicos. Por exemplo, Axie Infinity (Delic y Delfabbro, 2024) é un xogo de estratexia e batalla inspirado en xogos como Pokémon. Os xogadores crían, adestran e loitan con criaturas dixitais chamadas Axies. Cada Axie é un NFT único coas súas propias características xenéticas e habilidades. Os aspectos craves do xogo que involucran NFT recóllense no cadro 7.1. O modelo de Axie Infinity foi particularmente popular en países en desenvolvemento, onde algúns

xogadores puideron xerar ingresos significativos a través do xogo. Con todo, é importante notar que, como calquera economía baseada en criptomoedas, está suxeita a volatilidade e riscos.

Cadro 7.1. Aspectos crave de Axie Infinity que involucran aos NFT.

Aspecto	Relación cos NFT
Criaturas	Cada Axie é un NFT único que pode comprarse, venderse ou intercambiarse no mercado do xogo.
Terreos virtuais	O xogo inclúe parcelas de terra virtual que tamén son NFT. Os xogadores poden comprar, vender e desenvolver estas terras.
Elementos do xogo	Outros elementos como partes de corpo de Axie, pociones e elementos decorativos tamén son NFT.
Economía do xogo	Os xogadores poden gañar tokens SLP (Smooth Love Potion) xogando, que se utilizan para criar novos Axies. Estes tokens teñen valor no mundo real e poden intercambiarse.

O sector da moda e dos artigos de luxo adoptou os NFT como unha forma de autenticar produtos e combater a falsificación. As marcas poden emitir NFT xunto cos seus produtos físicos, proporcionando un certificado dixital de autenticidade que é inmutable e facilmente verificable.

No ámbito da música, os NFT ofrecen aos artistas novas formas de conectar cos seus fans e monetizar o seu traballo. Poden vender edicións limitadas de álbums, experiencias exclusivas ou mesmo dereitos de regalías sobre cancións específicas, todo apoiado pola tecnoloxía blockchain.

Os NFT baseados na tecnoloxía blockchain facilitan unha propiedade segura e transparente do activo dixital. Cada NFT almacénase nunha rede blockchain, que proporciona un rexistro público da propiedade e as transaccións, o que permite verificar a autenticidade e a propiedade do NFT. Os NFT gañaron popularidade no mundo da arte e a cultura popular como unha oportunidade para monetizar obras dixitais como pezas únicas, de maneira que os coleccionistas poidan posuír e intercambien estes activos dixitais. Tamén se utilizaron en xogos, no intercambio de estampillas, ou no mercado de bens inmoables

virtuais, facilitando posuír e intercambiar obxectos e propiedades únicos no metaverso.

7.1 Creación dun NFT

Para crear un NFT, o primeiro paso sería seleccionar unha plataforma block-chain que admita a creación e propiedade de activos dixitais. As plataformas máis populares para crear NFT son nestes momentos Ethereum, Binance Smart Chain e Flow. A continuación temos que identificar ou crear o activo dixital que queremos converter nun NFT. Pode ser unha obra de arte, un clip de música ou vídeo, un estampilla en formato dixital ou calquera outro contido ou activo. Unha vez que seleccionamos o noso activo dixital, temos que acuñar o NFT para o mesmo. Isto implica crear un token único que represente o activo dixital na rede blockchain elixida. Ademais da identificación única do propio activo, o NFT terá información sobre o mesmo, como o seu nome, descrición, creador, localización, etc. Esta información configuraría os metadatos do NFT. Tamén teremos que fixar o prezo e as condicións para a súa transferibilidade. O prezo pode ser unha cantidade determinada ou fixarse a través dunha poxa. Por outra banda, se poden establecer royalties ou regalías para o creador do novo NFT, que supoñerán unha porcentaxe das futuras vendas do NFT.

Unha vez creado e acuñado o NFT, poñeríámolo á venda nun mercado que admita NFT. Algúns mercados populares para NFT son OpenSea¹, Nifty Gateway² e SuperRare³. Unha vez que un comprador adquire o noso NFT, teremos que transferirlle a propiedade do mesmo. Isto faise a través dunha transacción que actualiza os rexistros de propiedade na blockchain onde está depositado o NFT. Cando un comprador adquire un NFT, o contrato intelixente pode programarse para transferir automaticamente a propiedade ao comprador e actualizar os rexistros de propiedade na blockchain.

Os contratos intelixentes son o instrumento xeralmente utilizado para automatizar a compra, a venda e a transferencia de propiedade dos NFT e en xeral dos activos dixitais. Poden utilizarse para automatizar as vendas, xa que un contrato intelixente pode programarse para vender automaticamente un NFT ao mellor ofertante nunha poxa, ou para vender un NFT a un prezo fixo. Os contratos intelixentes poden utilizarse tamén para distribuír automaticamente royalties aos creadores dun NFT. O contrato pode programarse para transferir

¹opensea.io

²www.niftygateway.com

³superrare.com

automaticamente unha porcentaxe do prezo de venda ao creador cando o NFT se revenda nun mercado secundario. Finalmente, os contratos intelixentes poden utilizarse para definir condicións e normas para os NFT. Por exemplo, un contrato intelixente pode programarse para restrinxir a revenda dun NFT durante un determinado período de tempo, ou para esixir que se cumpran certas condicións antes de poder transferir un NFT.

A continuación presentamos un exemplo sinxelo de acuñado dun NFT e a súa carga na rede de proba Sepolia. O noso activo dixital é o representado na figura 7.1. Os metadatos asociados ao mesmo serían os recollidos no rexistro JSON seguinte:

```

1  {                                                                    nft-metadata.json
2      "attributes": [
3          {
4              "trait_type": "Location",
5              "value": "New York"
6          },
7          {
8              "trait_type": "Style",
9              "value": "Andy Warhol"
10         }
11     ],
12     "description": "My first NFT: A really coooooool hydrant.",
13     "image": "ipfs://QmfBhAcyKQSZJep6qYYcuDQP8hrFL08yJYQN39UPuDVraQ",
14     "name": "CoolHydrant"
15 }
```

Podemos ver que a referencia ao activo nos metadatos é un identificador de contido (CID) en IPFS, almacenado no campo **image** (ver capítulo 5). Á súa vez, será o CID do rexistro JSON cos metadatos o que almacenaremos na cadea de bloques ao acuñar o NFT.

O código do contrato intelixente para acuñar os nosos NFT, escrito en Solidity, é o seguinte:

```

1  // SPDX-License-Identifier: MIT                                     MyNFT.sol
2  // Creación dun NFT na rede de proba Sepolia, segundo o tutorial
3  // Ethereum titorial "Como escribir e despregar un NFT"
4  //
5  // Baseado en [https://docs.openzeppelin.com/contracts/3.x/erc721]
6  // (https://docs.openzeppelin.com/contracts/3.x/erc721)
7  //
8  //
```



Figura 7.1. Activo dixital para crear un NFT.

```

9 // Versión de Solidity, utilizando o versionado semántico.
10 pragma solidity ^0.8.20;
11
12 // ERC721.sol contén a implementación da estándar ERC-721, que
13 // o noso contrato NFT herdará. Para ser un NFT válido, o contrato
14 // intelixente debe implementar todos os métodos do estándar
15 // ERC-721.
16 import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
17
18 // Ownable.sol establece o control de acceso ao noso contrato
19 // intelixente, de modo que só o propietario do contrato
20 // intelixente poida acuñar NFTs. O control acceso é opcional. Se
21 // calquera poidese acuñar un NFT usando determinado contrato
22 // intelixente, omitiríamos a palabra Ownable
23 // na liña 41 e 61, e onlyOwner na liña 86.)
24 import "@openzeppelin/contracts/access/Ownable.sol";
25
26 // O contrato ERC721URIStorage é unha implementación de ERC721 que
27 // inclúe os metadatos estándar (IERC721Metadata), así como un
28 // mecanismo para definir metadatos para cada NFT. De aí vén o
29 // método _setTokenURI: utilizámolo para almacenar un metadato NFT
30 // fóra da cadea, máis concretamente en IPFS.
31 import "@openzeppelin/contracts/token/ERC721/extensions/" +
32     "ERC721URIStorage.sol";
33
34 // Así, o contrato utilizará almacenamento externo para os
35 // metadatos do NFT e para o propio NFT. Como vimos,
36 // só pode ser invocado polo seu propietario. Utilizamos
37 // un contador para levar a conta de NFT xerados e garantir

```

```

38 // que os NFT xerados teñen todos eles IDs únicos.
39 // Omite a referencia á interface Ownable se queres que
40 // calquera poida acuñar NFT con este contrato.
41 contract MyNFT is ERC721URIStorage, Ownable {
42     using Counters for Counters.Counter;
43
44     // O noso contrato intelixente utiliza un contador
45     // para levar a conta do número total de NFT acuñados e
46     // establecer o IDE único do noso novo NFT.
47     // A cada NFT acuñado cun contrato intelixente débesele
48     // asignar un IDE único. Neste caso, o IDE único vén
49     // determinado polo número total de NFT creados co
50     // contrato. Por exemplo, o primeiro NFT acuñado co
51     // noso contrato intelixente ten un IDE de "1", o segundo
52     // "2" etc.)/ etc.)
53
54     uint256 private _tokenIds; // 0 contador.
55
56     // O primeiro argumento do construtor de ERC721 é o nome do
57     // contrato intelixente, e o segundo é o seu símbolo. O
58     // construtor do noso contrato non solicita argumentos.
59     // Tamén se inicializa o contrato Ownable asignando como
60     // primeiro dono o creador do contrato.
61     constructor() ERC721("MyNFT", "NFT") Ownable(msg.sender) {
62         _tokenIds = 0;
63     }
64
65     // Esta é a verdadeira (e única) función ofrecida polo noso
66     // contrato intelixente: a capacidade de acuñar NFTs. Para o
67     // resto de métodos definidos no estándar ERC721 utilizamos
68     // as implementacións por defecto que proporciona ERC721.sol.
69     // Só pode ser invocada polo propietario do contrato
70     // intelixente.
71     //
72     // Omite onlyOwner se desexas que calquera poida utilizala.
73     //
74     // A función recibe como parámetros:
75     // (1) A carteira do receptor do NFT (o creador do contrato
76     //     no noso caso).
77     // (2) O URI dos metadatos do token. Esta cadea debe
78     //     apuntar a un documento JSON que describe os metadatos
79     //     do NFT.
80     // Os metadatos dun NFT son realmente o que lle dá vida,
81     // permitíndolle ter propiedades configurables, como
82     // un nome, descrición, imaxe e outros atributos.
83     // Devolve un número que representa o IDE do NFT
84     // recentemente acuñado.
85     function mintNFT(address recipient, string memory tokenURI)
86         public onlyOwner

```

```

87     returns (uint256)
88     {
89         // Incrementa o IDE do NFT para garantir IDs únicos
90         // e obtén o IDE deste NFT.
91         _tokenIds++;
92         uint256 newItemId = _tokenIds;
93
94         // Imos acuñar o NFT. _mint() chama a certos métodos
95         // da biblioteca ERC-721 herdada, e finalmente
96         // devolve un número que representa o IDE do NFT
97         // recién acuñado.
98         _mint(recipient, newItemId);
99
100        // Almacenamos de forma persistente o tokenURI que fai
101        // referencia aos metadatos deste NFT
102        _setTokenURI(newItemId, tokenURI);
103
104        return newItemId;
105    }
106 }
```

Como imos utilizar a testnet Sepolia, que é unha testnet de Ethereum, baseamos os nosos NFT no estándar ERC721. Trátase dunha interface estándar para NFT na blockchain de Ethereum definida en 2018 para normalizar a forma en que se crean, transfírense e xestionan os NFT en devandita blockchain. Os tokens ERC721 son únicos e cada token ten o seu propio identificador e metadatos que proporcionan información adicional sobre o activo que representa. O estándar ERC721 foi amplamente adoptado pola comunidade Ethereum e moitos mercados populares de NFT, como os citados antes, admiten tokens ERC721.

Como todas as transaccións, a transacción para forxar un NFT leva un custo asociado en gas. Ao solicitar a transacción, debemos indicar o límite de gas, é dicir, o prezo máximo que estamos dispostos a pagar expresado en weis (1 ETH = 10^{18} weis), no noso caso 500.000 weis ou 0,0000005 ETH.

O fragmento de código seguinte en Node.js utilizaríase para facer efectivo o acuñado e envío á rede de probas Sepolia do noso NFT.

```

1 // Script para acuñar un NFT utilizando o contrato  mint-nft.js
2 // intelixente previamente despregado coa función mintNFT.
3 //
4 // Uso: node scripts/mint-nft.js
5 //
6 // As variables de contorna obtéñense do arquivo .env
```

```

7
8 require("dotenv").config()
9 // URL da API da aplicación Alchemy
10 // creada para despregar o noso contrato. Alchemy
11 // proporciónanos un acceso á testnet Sepolia, onde
12 // despregamos o noso contrato.
13 const API_URL = process.env.API_URL;
14
15 // Dirección da carteira autorizada a acuñar NFTs
16 // co contrato (deseñámolo para que só a carteira
17 // autorizada poida crealos)
18 const PUBLIC_KEY = process.env.PUBLIC_KEY;
19
20 // Clave privada da carteira. Utilízase para cargar
21 // os gastos de gas incorridos no acuñado do NFT.
22 const PRIVATE_KEY = process.env.PRIVATE_KEY;
23
24 // Alchemy Web3 é unha envoltura ao redor de Web3.js,
25 // a librería máis popular, case podemos dicir que
26 // o estándar, para acceder á Web3, que proporciona
27 // métodos API mellorados e outros elementos interesantes.
28 // Está deseñada para requirir unha configuración mínima.
29 const { createAlchemyWeb3 } = require("@alch/alchemy-web3");
30
31 // Creamos un obxecto Web3 (web3) para acceder a Sepolia
32 // a través de Alchemy.
33 const web3 = createAlchemyWeb3(API_URL);
34
35 // Tomamos a ABI (Application Binary Interface) do
36 // noso contrato para interactuar con el.
37 const contract =
38     require("../artifacts/contracts/MyNFT.sol/MyNFT.json")
39
40 // Dirección do contrato NFT creado
41 const contractAddress = process.env.CONTRACT_ADDRESS;
42
43 // Creamos un obxecto Contract (nftContract) usando a ABI e a
44 // dirección.
45 const nftContract =
46     new web3.eth.Contract(contract.abi, contractAddress);
47
48 // Definimos unha función para chamar á función de acuñado
49 // do noso contrato.
50 async function mintNFT(tokenURI) {
51     // Obtemos un nonce.
52     const nonce =
53         await web3.eth.getTransactionCount(PUBLIC_KEY, 'latest');
54
55     // Creamos a transacción de acuñado chamando ao mintNFT do noso

```

```

56 // contrato)
57 const tx = {
58   'from': PUBLIC_KEY,      // Creador: nós (a nosa carteira)
59   'to': contractAddress,   // 0 contrato de acuñado de NFTs
60   'nonce': nonce,          // o nonce para a transacción
61   'gas': 500000,           // 500.000 wei (0,0000005 ETH)
62   'data': nftContract.methods.mintNFT(PUBLIC_KEY,
63     tokenURI).encodeABI()
64 }
65
66 // Agora que creamos a nosa transacción, necesitamos
67 // asinala para poder enviala á web3. Aquí é onde usaremos
68 // a nosa clave privada.
69 // web3.eth.sendSignedTransaction daranos o hash da
70 // transacción, que podemos usar para asegurarnos
71 // que a nosa transacción foi forxada e non foi
72 // descartada pola rede Sepolia.
73 // Como se pode ver no código que segue, na sección da
74 // transacción engadimos unha comprobación de erros para
75 // saber se a nosa transacción se forxou correctamente.
76 const signPromise = web3.eth.accounts.signTransaction(tx,
77   PRIVATE_KEY)
78 signPromise
79   .then((signedTx) => {
80     web3.eth.sendSignedTransaction(
81       signedTx.rawTransaction,
82       function (err, hash) {
83         if (!err) {
84           console.log(
85             "O hash da transacción é: ",
86             hash,
87             "\nEstado da transacción no mempool \
88               de Alchemy."
89           )
90         } else {
91           console.log(
92             "Algo saiu mal ao forxar a transacción:",
93             err
94           )
95         }
96       }
97     )
98   })
99   .catch((err) => {
100     console.log(" signPromise fallóu:", err)
101   })
102 }
103
104 // Lanzamos o acuñado utilizando o CID en IPFS do JSON cos

```



```

105 // metadatos do NFT.
106 mintNFT("ipfs://QmfJQ4xLorVNXLQ3S8cSGKtpBVYs44WD63C4Bi7yoWktR2")

```

O código anterior utiliza a biblioteca Alchemy Web3, unha envoltura ao redor de Web3.js, unha biblioteca moi popular para acceder á Web3, que proporciona métodos API mellorados e outros elementos interesantes. Outra opción común é utilizar a biblioteca ethers.js. O fragmento seguinte utiliza dita biblioteca a través do Hardhat Toolbox.

```

1 // Script para crear un NFT utilizando o contrato  mint-nft2.js
2 // intelixente previamente despregado coa función mintNFT.
3 //
4 // Uso: node scripts/mint-nft.js
5 //
6 // As variables de contorna obtéñense do arquivo .env
7
8 require('dotenv').config()
9
10 // Dirección da carteira autorizada a crear NFTs
11 // co contrato (deseñámolo para que só a carteira
12 // autorizada poida crealos)
13 const PUBLIC_KEY = process.env.PUBLIC_KEY;
14
15 // Clave privada da carteira. Utilízase para cargar
16 // os gastos de gas incorridos na creación do NFT.
17 const PRIVATE_KEY = process.env.PRIVATE_KEY;
18
19 // Clave da API da aplicación Alchemy
20 // creada para despregar o noso contrato.
21 // Alchemy proporciónanos un acceso á testnet Sepolia,
22 // onde despregamos o noso contrato.
23 const API_KEY = process.env.API_KEY;
24
25 // Creamos un obxecto Hardhat Runtime Environment
26 // (hre) para acceder a Sepolia a través de Alchemy.
27 const hre = require("hardhat");
28 const provider =
29     new hre.ethers.AlchemyProvider("sepolia", API_KEY);
30
31 // Tomamos a ABI (Application Binary Interface) do
32 // noso contrato para interactuar con el.
33 const abiJSON =
34     require("../artifacts/contracts/MyNFT.sol/MyNFT.json");
35
36 // Dirección do contrato NFT creado
37 const contractAddress = process.env.CONTRACT_ADDRESS;

```

```

38
39 // Creamos un obxecto Contract (nftContract) usando a ABI, a
40 // dirección e o provider (Alchemy)
41 const nftContract = new hre.ethers.Contract(contractAddress,
42     abiJSON.abi, provider);
43
44 // Creamos un obxecto Wallet (wallet) para cargar os gastos do
45 // minado do NFT.
46 const wallet = new hre.ethers.Wallet(PRIVATE_KEY, provider);
47
48 // Cargamos o URI do NFT (CID en IPFS do rexistro JSON cos
49 // metadatos)
50 const tokenURI = process.env.NFT_URI;
51
52 // Función para a forxa do noso NFT.
53 async function main() {
54
55     // Trátase dunha transacción que fai cambios no
56     // contrato intelixente, polo que ten que ir asinada.
57     // Utilizamos para iso a carteira, que ten información
58     // sobre a clave privada (para asinar)
59     const contractSigned = nftContract.connect(wallet);
60
61     // Chamamos á función de forxa do contrato intelixente.
62     // Pasámoslle como parámetros a dirección do creador do
63     // contrato (único que pode crear NFTs) e o URI do NFT.
64     const tx = await contractSigned.mintNFT(PUBLIC_KEY, tokenURI);
65
66     // Sacamos por consola o hash da transacción
67     console.log(tx.hash);
68
69     await tx.wait();
70     console.log ("Creación do NFT completada con éxito");
71 }
72
73 main()
74     .then(result => {
75         console.log('OK: Forxa completada.');

```

Tras executar este programa (ben coa biblioteca web3 ou a biblioteca ethers), o noso NFT estará cargado na rede Sepolia. Se consultamos a nosa carteira, apareceranos no apartado de NFT dos que somos propietarios, como podemos ver na figura 7.2.



Figura 7.2. NFT na nosa carteira, no apartado onde aparecen os NFT dos que somos propietarios na rede Sepolia.

7.2 Combinando tokens funxibles e non funxibles

O estándar ERC1155 de Ethereum define a posibilidade de manexar tanto tokens funxibles como non funxibles nun só contrato intelixente, de maneira que nun só contrato se poidan definir unha ampla variedade de activos dixitais, como por exemplo definir para un xogo en liña tanto as moedas que utilizarás para adquirir elementos de xogo como os propios elementos de xogo únicos. Ademais de simplificar a xestión de activos dixitais, esta flexibilidade tamén reduce os custos de transacción e mellora a eficiencia xeral da cadea de bloques, xa que se poden realizar múltiples operacións nunha soa transacción.

O proceso de creación dun token, sexa este funxible ou non funxible, é bastante similar ao da creación destes tokens cos estándares que vimos no capítulo 4 ou neste propio capítulo:

1. Programación do contrato intelixente. Créase un contrato intelixente que implementa o estándar ERC-1155. Agora, este contrato actúa como unha *fábrica* que pode producir e xestionar múltiples tokens de ambos os tipos.

2. Asignación dun identificador único. Cada tipo de token, sexa fungible ou non fungible, recibe un identificador único dentro do contrato. Este identificador é como un *número de modelo* para ese tipo de token.
3. Establecemento da cantidade inicial. Defínese a cantidade inicial de tokens que se crearán. Para tokens fungibles, esta cantidade adoita ser moi grande, dependendo do uso que vaiamos facer del (por exemplo, se o imos a utilizar como unha nova criptomoneda).
4. Creación de metadatos. Defínense os metadatos do token, que poden incluír un nome, símbolo, e posiblemente unha descrición ou imaxe asociada no caso dos NFT.
5. Programación da función de acuñado. Programamos unha ou varias funcións para acuñar ou crear novos tokens de cada tipo. Estas funcións controlan cantos tokens se crean e quen pode crealos.
6. Programación de funcións de transferencia. Escrebemos o código das funcións que permiten transferir estes tokens entre diferentes direccións (é de dicir, carteiras) na blockchain. Tamén podemos utilizar as funcións por defecto definidas nos contratos normalizados que desenvolven o estándar ERC1155, como fixemos no caso dos NFT tradicionais.
7. Despregamento do contrato. Unha vez que o contrato está listo, desprégase na rede Ethereum do mesmo xeito que despregamos calquera contrato intelixente, o que o fai activo e operativo.

Dentro do contrato intelixente, cada tipo de token trátase de maneira semellante. As diferenzas establécense ao definir os distintos parámetros que caracterizan o token:

- Cada NFT é único, ou dito doutro xeito, un token NFT é diferente de calquera outro token, mentres que os tokens fungibles son intercambiables e son todos funcionalmente iguais. Cada NFT ten o seu propio identificador único, mentres que todos os tokens fungibles do mesmo tipo comparten o mesmo identificador.
- Para os tokens NFT adóitase definir unha cantidade máxima de 1 por cada identificador, e en todo caso unhas poucas unidades nalgúns situacións especiais (por exemplo, 3 copias de determinado amuleto dixital que hai que acadar para avanzar nun xogo). Doutra banda, os

tokens funxibles adoitan existir en grandes cantidades (por exemplo, tokens que se utilizan como moeda de cambio nun xogo). En definitiva, os NFT adoitan acuñarse individualmente ou en series limitadas, mentres que os tokens funxibles poden acuñarse en grandes cantidades.

- Os NFT adoitan ter metadatos máis extensos e específicos.

A continuación presentamos o código dun contrato intelixente para acuñar dous tipos de tokens:

- Un token funxible denominado CARTINHO.
- Un NFT chamado MyNewNFT.

```

1 // SPDX-License-Identifier: MIT TokenBlend.sol
2 pragma solidity ^0.8.0;
3
4 // Importamos os contratos base que imos utilizar:
5 // - ERC1155.sol: prové a funcionalidade básica do estándar
6 //   ERC1155, mesmo para as funcións que non imos modificar
7 //   na nosa implementación.
8 // - Ownable.sol: soporta a definición de funcionalidade
9 //   exclusiva para o dono do contrato (inicialmente o seu
10 //   creador).
11 import "@openzeppelin/contracts/token/ERC1155/ERC1155.sol";
12 import "@openzeppelin/contracts/access/Ownable.sol";
13
14 contract TokenBlend is ERC1155, Ownable {
15     // Definimos os ID dos nosos tokens CARTINHO será
16     // un token funxible e ESTAMPILLA un token non
17     // funxible. Estas constantes permítenos referirnos
18     // facilmente a estes tokens específicos
19     // nas funcións do contrato.
20     uint256 public constant CARTINHO = 0;
21     uint256 public constant ESTAMPILLA = 1;
22
23     // Contador para NFTs únicos. O contrato utiliza
24     // _nextTokenId para xerar IDs a partir de 2, asegurando
25     // que non haxa conflito con CARTINHO (0) e ESTAMPILLA (1)
26     uint256 private _nextTokenId = 2;
27
28     // Variables de estado para guardar nome
29     // e símbolo.
30     string public name;
31     string public symbol;

```

```
32
33 // Mapping para almacenar os URI dos metadatos
34 // dos NFTs (p. ex., en IPFS)
35 mapping(uint256 => string) private _tokenURIs;
36
37 // Mapping para asociar IDs de tokens a nomes
38 mapping(uint256 => string) private _tokenNames;
39
40 // O construtor acuña os tokens, asigna
41 // nomes aos novos tokens. Recibe como
42 // parámetros:
43 // - O nome e o símbolo xeral do
44 //   contrato intelixente, que serán inmutables.
45 // - O enderezo base dos metadatos dos diferentes
46 //   tokens que imos crear. Pasamos este parámetro
47 //   ao construtor do contrato ERC1155.
48 constructor(string memory _name, string memory _symbol,
49             string _base_url)
50     ERC1155(_base_url) {
51
52     // Asignamos nome e símbolo ao contrato.
53     name = _name;
54     symbol = _symbol;
55
56     // Asignamos nome aos nosos tokens.
57     _tokenNames[CARTINHO] = "CARTINHO";
58     _tokenNames[ESTAMPILLA] = "MyNewNFT";
59
60     // Acuñamos 1.000.000 de unidades do token
61     // fungible (CARTINHO)
62     _mint(msg.sender, CARTINHO, 1000000, "");
63
64     // Acuñamos 1 unidade do token non fungible
65     // (ESTAMPILLA)
66     _mint(msg.sender, ESTAMPILLA, 1, "");
67 }
68
69 // Función para acuñar máis tokens fungibles.
70 // Pasamos como parámetro a dirección á que transferiremos
71 // os novos tokens fungibles, e a cantidade de tokens.
72 // Os novos tokens manteñen o ID orixinal (ID = 0)
73 function mintFungible(address account, uint256 amount)
74     public onlyOwner {
75
76     // _mint: función definida en ERC1155.sol. Creamos os tokens
77     // (amount) e llelos asignamos á dirección indicada (account)
78     _mint(account, CARTINHO, amount, "");
79 }
80
```

```

81 // Función para acuñar un novo NFT. No canto de pasarlle
82 // como parámetro a cantidade de tokens, pasámoslle o URI
83 // dos metadatos do novo NFT. Asignámoslle un novo ID, que
84 // se devolve á dirección que invocou esta función
85 // (é dicir, o dono do contrato).
86 //
87 // Alternativamente, podemos deixar os metadatos nun ficheiro
88 // chamado <newItemId>.json no mesmo servidor que especificamos
89 // ao crear o contrato intelixente, onde <newItemId> é o ID
90 // do novo NFT que estamos a crear.
91 function mintNFT(address account, string memory tokenURI)
92     public onlyOwner returns (uint256) {
93
94     // Obtemos o ID do novo NFT e incrementamos
95     // o contador.
96     uint256 newItemId = _nextTokenId++;
97
98     _mint(account, newItemId, 1, "");
99
100     // Asignamos os metadatos ao novo NFT.
101     _setTokenURI(newItemId, tokenURI);
102
103     return newItemId;
104 }
105
106 // Función para establecer o URI dos metadatos dun NFT
107 // (p. ex., o seu CID en IPFS).
108 function _setTokenURI(uint256 tokenId, string memory tokenURI)
109     internal virtual {
110     require(bytes(tokenURI).length > 0,
111         "Falta proporcionar un URI");
112
113     // Metemos o URI no mapping.
114     _tokenURIs[tokenId] = tokenURI;
115 }
116
117 // Función para obter o URI dos metadatos dun token.
118 function uri(uint256 tokenId) public view virtual
119     override returns (string memory) {
120     string memory tokenURI = _tokenURIs[tokenId];
121
122     // Comprobamos se temos o URI no mapping.
123     if (bytes(tokenURI).length > 0) {
124         // Está no mapping. O devolvemos.
125         return tokenURI;
126     }
127
128     // Se non temos o URI no mapping, obtémolo
129     // do ficheiro <tokenId>.json almacenado no

```

```

130 // servidor correspondiente, tal como especificamos
131 // ao crear o contrato.
132 return super.uri(tokenId);
133 }
134
135 // Función para obter o nome asignado a un token
136 // acuñado con este contrato.
137 function tokenName(uint256 tokenId) public
138     view returns (string memory) {
139     return _tokenNames[tokenId];
140 }
141 }

```

O contrato importa e herda de dos contratos `ERC1155.sol` e `Ownable.sol` de OpenZeppelin, que proporcionan a implementación base do estándar ERC-1155 e a funcionalidade de propiedade, respectivamente, do mesmo xeito que fixemos co contrato `MyNFT`. Con este contrato acuñamos un millón de `CARTINHOS` e un único NFT que chamaremos `MyNewNFT` (ver figura 7.3). Os metadatos do noso novo NFT serían por exemplo:

[illegible]

O contrato define tamén dúas funcións de acuñado, **MintFungible** para acuñar máis CARTINHOS e **MintNFT** para crear un novo NFT, que terá un novo ID único e un novo URI asociado aos seus metadatos.

O construtor do contrato asigna un nome e un símbolo globais ao contrato e pasa ao construtor de ERC1155 o esquema de URI para obter os metadatos dos diferentes tokens que imos crear, de xeito que ao invocar a función



Figura 7.3. Activo dixital para crear un novo NFT.

`uri` predefinida cun ID de token, obtéñense os metadatos de arquivos JSON chamados `ID.json` do servidor correspondente. Por exemplo, o URI

```
https://tokenstore.myserver.gal/{id}.json
```

reemplazarase polo URI:

```
https://tokenstore.myserver.gal/00...004.json
```

se tentamos acceder ao token cuxo ID é 4. En xeral, converteremos o ID do token correspondente nun número hexadecimal de 64 díxitos (os `..` do exemplo anterior representan 59 zeros consecutivos).

Neste contrato, reescribimos a función orixinal para devolver a URI almacenada no propio contrato no mapping `_tokenURIs`, salvo que esa URI non fora inicializada. Nese caso, recuperaremos o comportamento orixinal da función `uri`. Tamén engadimos unha función interna `_setTokenURI` para asignar a

URI dos metadatos do novo NFT, e asignámola cando o acuñamos. En calquera caso, os metadatos dos dous tokens creados orixinalmente (CARTINHO e MyNewNFT) serán accesibles tal como indica a estándar ERC1155, e dicir, en ficheiros chamados respectivamente `00...01.json` e `00...02.json` almacenados no servidor indicado no esquema de URI que pasamos ao constructor do contrato (o nome de cada ficheiro terá 63 ceros seguidos dos díxitos 1 e 2 respectivamente, e a extensión `.json`).

Capítulo 8

Finanzas descentralizadas

As finanzas descentralizadas (DeFi) refírense ao ecosistema financeiro que opera nunha rede descentralizada utilizando a tecnoloxía de cadeas de bloques. Permite aos usuarios acceder a servizos financeiros, como préstamos, mercados financeiros e investimentos, sen necesidade de intermediarios como a banca e outras institucións financeiras tradicionais.

As Finanzas Descentralizadas (DeFi) baséanse en tres principios fundamentais que as diferencian dos sistemas financeiros tradicionais: a descentralización, a transparencia e o acceso público. DeFi opera nunha rede blockchain, polo que non hai ningunha autoridade central ou intermediario que controle ou supervise o sistema. No seu lugar, os usuarios interactúan directamente entre si a través de contratos intelixentes (ver o capítulo 4) para automatizar as transaccións financeiras. Ao materializarse os contratos e transaccións directamente a través de lóxica programada, non son necesarios axentes que materialicen devanditos contratos e transaccións. Ademais, ao almacenarse ditas transaccións de maneira persistente e inmutable na blockchain, tampouco se necesitan axentes que almacenen e manteñan dita información. Isto elimina a necesidade de bancos ou outras institucións financeiras tradicionais e promove un sistema financeiro máis democrático e inclusivo.

As DeFi considéranse un ecosistema transparente porque á todas as transaccións rexístranse no libro maior público da blockchain (ver o capítulo 1), o que facilita o seguimento e a verificación da súa autenticidade. Os usuarios poden ver exactamente como se utilizan os seus fondos e asegurarse de que o sistema funciona de forma xusta e imparcial. Este nivel de transparencia fomenta a responsabilidade e xera confianza entre os usuarios.

As DeFi garanten ademais o acceso público porque o ecosistema está aberto a calquera persoa con conexión a Internet, independentemente da súa

localización ou situación financeira. Ao eliminarse os intermediarios, elimínanse os custos asociados á devandita intermediación, custos que poden resultar elevados ata o punto de constituír unha barreira de entrada e excluír a determinados grupos sociais do acceso aos servizos financeiros. Ao ofrecer unha plataforma máis inclusiva e accesible, as DeFi promoven unha maior equidade e igualdade de oportunidades no sector financeiro.

De feito, o impacto das DeFi nos sistemas financeiros tradicionais é moi significativo porque desafía o statu quo ofrecendo unha alternativa máis transparente, segura e accesible. Ao eliminar os intermediarios, as DeFi permiten reducir os custos operativos á vez que melloran a eficiencia das transaccións financeiras. Ademais, as plataformas DeFi ofrecen maior flexibilidade e liquidez, permitindo aos usuarios acceder aos servizos financeiros en calquera momento e desde calquera lugar.

Con todo, as DeFi tamén prantexan retos diferentes aos dos sistemas financeiros tradicionais, xa que non se adaptan de maneira sinxela aos mesmos marcos reguladores e mecanismos de supervisión que rexen as institucións financeiras convencionais, o que pode crear riscos para os investidores e suscitar preocupación pola seguridade e a estabilidade do sector financeiro. En xeral, aínda que as DeFi ofrecen interesantes oportunidades de innovación e pódense considerar unha opción disruptiva para a operativa financeira, o seu impacto nos sistemas financeiros tradicionais dependerá de como evolucionen e se adapten aos marcos normativos e ás demandas do mercado.

Nos apartados seguintes imos presentar de maneira sucinta os aspectos máis relevantes do ecosistema DeFi, comezando polos mercados financeiros descentralizados. Máis adiante presentaremos as plataformas de préstamo descentralizadas e o concepto de minería de liquidez como un instrumento para proporcionar liquidez a ambos os tipos de plataformas. A continuación, faremos un breve repaso doutros produtos derivados e estratexias de xestión de activos na contorna DeFi. Completaremos esta presentación identificando algúns retos e consideracións normativas relevantes.

8.1 Mercados financeiros descentralizados

As bolsas ou mercados financeiros descentralizados (*decentralized exchanges* ou DEX) son mercados dixitais que permiten aos usuarios negociar criptomoedas ou outros activos dixitais de igual a igual, sen necesidade dos intermediarios ou brokers que operan nas bolsas e mercados centralizados.

Ao funcionar sobre unha blockchain, herdán as características de seguridade, transparencia e autonomía para os usuarios desta tecnoloxía.

Nun DEX, as ordes emparéllanse directamente entre compradores e vendedores a través de contratos intelixentes, que automatizan o proceso de negociación e garanten que todas as transaccións se executen de forma xusta e imparcial sen intermediarios ou brokers. Por tanto, redúcese de maneira drástica o custo das transaccións e elimínanse moitos dos requisitos e limitacións existentes nos mercados tradicionais. O DEX mantén un libro de ordes descentralizado accesible publicamente na blockchain onde se rexistran todas estas intencións de compra e venda de activos. Os usuarios crean pares de intercambio (*trading pairs*) mediante a vinculación de dous activos diferentes (p. ex., un par de intercambio podería consistir nun criptoactivo popular, como Ethereum ou Bitcoin, e outro criptoactivo diferente). Cando un usuario coloca unha orde de compra ou venda, un contrato intelixente no DEX tenta encaixar automaticamente esa orde con outras ordes rexistradas no libro de ordes. Se hai unha coincidencia, execútase o intercambio de maneira automática. Os usuarios poden interactuar cun DEX a través das súas carteiras de criptomonedas (*wallets*), e os DEX a miúdo proporcionan interfaces de usuario amigables e integracións con diversas carteiras para mellorar a experiencia do usuario.

Os DEX ofrecen varias vantaxes sobre os mercados centralizados, como comisións máis baixas, maior liquidez, máis control dos usuarios sobre os seus activos e maior privacidade ao non ser necesario o rexistro dos usuarios ante ningunha entidade. Ademais, ao non depender de puntos únicos de fallo, redúcese o risco de hackeos ou fraudes.

Con todo, tamén presentan algúns inconvenientes, como a limitación dos activos que se poden negociar aos pares de negociación establecidos, a lentitude das transaccións e as posibles vulnerabilidades no código dos contratos intelixentes que implementan a operativa. Por outra banda, algúns DEX poden ter unha curva de aprendizaxe máis pronunciada para os usuarios non técnicos en comparación coas plataformas centralizadas, que adoitan ser máis simples conceptualmente. Ademais, o feito de depender dunha entidade que as opera garante uns mínimos servizos de apoio e consultoría.

Un dos DEX máis populares e amplamente utilizados é Uniswap¹. Uniswap opera na rede Ethereum e foi pioneiro no concepto de xeración automática de mercado (*Automated Market Maker*, AMM), que permite aos usuarios intercambiar criptomonedas de maneira descentralizada sen a necesidade dun libro

¹uniswap.org

de ordes tradicional. No seu lugar, utilízanse contratos intelixentes e a liquidez proporcionada polos propios usuarios para realizar os intercambios a través dos *pools*. Un pool en Uniswap refírese a unha carteira que contén diferentes pares de negociación de tokens creados polos usuarios, que deciden depositar os seus tokens no pool e compartilos con outros usuarios do sistema. Os pares de tokens poden conter calquera parella de criptoactivos ERC20 da rede Ethereum, xa sexa un token establecido (como o propio Ethereum) ou un novo na escena das criptomoedas, e créanse automaticamente cando os usuarios depositan os seus tokens no pool e execútanse segundo as regras do sistema Uniswap. Isto significa que os prezos dos pares de tokens determínanse de acordo coas regras de oferta e demanda, sen intermediarios.

Os usuarios poden retirar os seus tokens dun pool en calquera momento e recibir as cantidades correspondentes do outro token do par. Isto permite aos usuarios negociar con flexibilidade e control sobre os seus activos dixitais. Ademais, os pares de tokens vólvense máis líquidos cando hai unha maior oferta e demanda no sistema.

Uniswap ten un modelo de gobernanza descentralizada baseada nun token propio chamado UNI, que permite aos seus titulares participar en decisións importantes sobre o protocolo. Este token serve para múltiples propósitos dentro do ecosistema, ademais da gobernanza, como o pago de comisións e a incentivación á participación dos usuarios. Como token de gobernanza, os titulares de UNI poden participar nos procesos de toma de decisións relacionados co desenvolvemento e a evolución da plataforma Uniswap, votando propostas e actualizando parámetros, por exemplo os importes das comisións ou a creación de pools de liquidez. Ademais, os usuarios que proporcionan liquidez aos pools de Uniswap son recompensados con tokens UNI, o que crea un incentivo para a participación e axuda a aumentar a liquidez xeral da plataforma.

Outras plataformas DEX relevantes serían SushiSwap², un DEX construído sobre Uniswap con características engadidas, como recompensas de provisión de liquidez e gobernanza baseada en NFT, ou Curve Finance³, un DEX centrado no comercio de stablecoin con pools de liquidez altamente eficientes e oráculos de prezos integrados para manter a estabilidade.

²www.sushi.com

³curve.fi

8.2 Préstamos descentralizados

No ecosistema DeFi, os préstamos descentralizados son unha forma de prestar e recibir diñeiro sen a necesidade de intermediarios financeiros como a banca ou as entidades de crédito. Do mesmo xeito que no caso dos DEX, estes préstamos utilizan a tecnoloxía blockchain para garantir a seguridade, transparencia e autonomía de prestamistas e prestatarios.

Os préstamos descentralizados poden tomar as mesmas formas que os préstamos tradicionais, incluíndo préstamos con intereses variables ou fixos, préstamos en criptomoedas ou en moeda convencional, préstamos instantáneos, préstamos avalados ou sen aval e préstamos para diferentes propósitos (investimento, consumo, hipotecarios, etc.). Os prestamistas poden obter rendibilidade ao prestar o seu diñeiro a outros usuarios, mentres que os prestatarios poden acceder a financiamento máis barato e con menos trámites e requisitos que a través de fontes tradicionais.

Os protocolos de préstamo e endebedamento tamén se implementan a través de contratos intelixentes que executan automaticamente as transaccións e garanten a seguridade do proceso. Estes protocolos poden tomar diversas formas, dependendo do tipo de préstamo, existindo contratos intelixentes específicos para cada modalidade de préstamo. Estes contratos reflicten as condicións específicas do préstamo, como o tempo de vencemento, o interese para pagar, o tipo e contía do aval requirido e as penalizacións polo atraso no pago.

Os usuarios poden acceder aos servizos de préstamo e endebedamento mediante plataformas específicas. Unha plataforma popular neste ámbito é Compound⁴. Trátase dunha plataforma DeFi que permite aos usuarios prestar e pedir prestado criptomoedas sen necesidade de intermediarios, baseada nun algoritmo de cálculo de tipos de interese dinámicos de acordo coas leis da oferta e a demanda, onde devanditos tipos axústanse automaticamente en función da oferta e a demanda de cada activo. Compound opera na cadea de bloques de Ethereum, onde os préstamos se perfeccionan mediante contratos intelixentes que garanten a transparencia das operacións. Os usuarios convértese en prestamistas depositando criptomoeda en Compound (Ethereum, BAT, DAI, ESDC, etc.), pola que reciben cantidades dun token específico a cambio, chamado COMP. Estes tokens representan a súa participación e xeran intereses en tempo real. Os prestatarios toman prestado a cambio dun interese, tamén calculado en tempo real.

⁴compound.finance

Compound é unha plataforma DeFi de préstamos anónimos, é dicir, sen requisitos de tipo *know your customer* (KYC), o que significa que os usuarios poden participar sen ter que pasar por un proceso de verificación de identidade. A gobernanza en Compound lévase a cabo a través dun sistema de votación descentralizado no que os titulares de tokens COMP poden propoñer cambios nos parámetros do protocolo, como os tipos de interese, os límites de préstamo ou os tipos de garantía. Estas propostas sométense á votación da comunidade e, se reciben suficiente apoio doutros titulares de COMP, impleméntanse na plataforma.

Outras plataformas relevantes no mundo dos préstamos DeFi son a xa comentada Uniswap, Aave, MakerDAO ou Yearn Finance, cada unha coas súas características específicas. Todas estas plataformas ofrecen unha experiencia de uso amigable e permiten aos usuarios prestar ou recibir diñeiro sen a necesidade de coñecementos técnicos avanzados sobre blockchain ou criptomoedas.

8.3 Minería de liquidez

A minería de liquidez ou *yield farming* refírese á práctica de proporcionar liquidez a un DEX ou plataforma de préstamo como as comentadas previamente, a cambio de obter recompensas en forma de tokens, por exemplo comisións por cada transacción, pagos de intereses ou novos tokens que se crean a través do proceso de minería. Consiste en achegar activos dixitais (p. ex., criptomoedas) a un fondo común, que logo se utilizan para facilitar as operacións entre compradores e vendedores na plataforma. Como vimos nos apartados anteriores, os mineiros de liquidez tamén poden recibir como recompensa tokens de gobernanza da plataforma correspondente, que representan unha participación no seu goberno. A minería de liquidez incentiva aos usuarios para achegar os seus activos á plataforma, o que axuda a aumentar a súa liquidez e o seu valor global.

Trátase dun compoñente importante dos ecosistemas DeFi porque fomenta a participación e o compromiso dos usuarios. Ao ofrecer recompensas por achegar liquidez, as plataformas poden atraer a máis usuarios e crear unha comunidade máis ampla ao redor das súas ofertas. Ademais, a minería de liquidez axuda a garantir que sempre haxa suficientes activos dispoñibles na plataforma, o que é esencial para manter a súa funcionalidade e estabilidade. De todos os xeitos, as taxas de rendemento na minería de liquidez poden ser altamente variables e dependen de factores como a demanda de liquidez no

protocolo e as recompensas ofrecidas. Os mineiros buscan maximizar os seus rendementos elixindo os pools que ofrecen as mellores taxas.

Ademais das recompensas e a participación na gobernanza, algunhas plataformas DeFi ofrecen a opción de depositar os tokens recibidos como recompensa nun contrato intelixente, para que se poidan usar para financiar o funcionamento da propia blockchain, nun proceso denominado (*staking*). Ao bloquear estes tokens en contratos intelixentes de staking, os usuarios poden seguir obtendo rendementos adicionais, e mesmo nalgúns casos permítense estratexias máis elaboradas, como a creación de estratexias de minería de liquidez composta, onde os rendementos obtidos reinvéstense automaticamente para maximizar os ingresos.

Aínda que a minería de liquidez pode chegar a converterse nunha actividade lucrativa, tamén conleva riscos. Os protocolos DeFi poden ser propensos a bugs ou ataques, e as taxas de rendemento poden cambiar rapidamente, volatilidade que afecta directamente os rendementos esperados.

8.4 Derivados DeFi e xestión de activos

No contexto de DeFi, créanse a miúdo activos sintéticos e produtos financeiros complexos aproveitando a flexibilidade que proporciona a lóxica programada nos contratos intelixentes. Estes instrumentos financeiros derivados caracterízanse porque o seu valor está asociado ao dun activo ou activos subxacentes, en moitos casos dun modo moi elaborado. Entre os activos sintéticos máis representativos están as opcións, os futuros, os índices sintéticos e os tokens apalancados e inversos.

Os activos sintéticos son representacións ou xemelgos dixitais de activos do mundo real. De todos os xeitos, estes activos non están directamente apoiados polo activo subxacente, senón que se crean sinteticamente a través de contratos intelixentes. Un exemplo de activo sintético podería ser un token indexado ao prezo do ouro ou dunha materia prima. A través de contratos intelixentes, os usuarios poden posuír un token que reflecta o valor do ouro sen posuír fisicamente nada do devandito metal precioso.

As opcións son contratos derivados que outorgan ao titular o dereito (pero non a obrigação) de comprar ou vender un activo subxacente a un prezo específico nunha data determinada. As opcións non son exclusivas do mundo DeFi, pero neste contexto os contratos de opcións, en realidade programas software máis que contratos en si, poden utilizarse para crear estratexias

complexas, como estrangulamientos ou spreads de estratexia, que se aplican automaticamente e que permiten aos usuarios xestionar o risco e buscar rendementos en diferentes escenarios de mercado. Os contratos de futuros, que tampouco son específicos do mundo DeFi, son acordos para comprar ou vender un activo subxacente nunha data futura a un prezo acordado. Os futuros permiten aos participantes apostar sobre o prezo futuro dun activo, e en DeFi utilízanse moitas veces para crear produtos que permitan aos usuarios apostar sobre o prezo futuro de criptomoedas, índices ou outros activos.

Os índices sintéticos créanse a miúdo mediante a combinación de varios activos subxacentes ou a exposición a diversos activos. Estes índices non necesariamente replican índices tradicionais, senón que se constrúen sintéticamente mediante a lóxica dun contrato intelixente. Por exemplo, un índice sintético podería seguir o rendemento combinado de varias criptomoedas. Os participantes poden obter exposición diversificada a través dun token que represente este índice sintético.

Os tokens apalancados e inversos proporcionan exposición apalancada ou inversa a un activo subxacente. Poden utilizarse para estratexias de apancamento ou cobertura. Unha estratexia de apancamento implica o uso de capital prestado para realizar investimentos, mentres que unha estratexia de cobertura implica protexer unha posición existente ou anticiparse a perdas potenciais mediante a adopción dunha posición compensatoria noutro activo ou instrumento financeiro. Por exemplo, un token apalancado 3x pode conseguir o triplo do rendemento diario dun activo subxacente. Doutra banda, un token inverso podería buscar obter o inverso do rendemento diario para compensar posibles perdas. As permutas de taxas de interese son contratos derivados que permiten o intercambio de fluxos de efectivo asociados a taxas de interese. En DeFi, pódense implementar permutas de taxas de interese para xestionar riscos asociados con taxas de interese variables en protocolos de préstamos e liquidez.

En calquera caso, hai que ter en conta que os derivados e produtos financeiros complexos presentan riscos significativos, tanto no mundo físico como no mundo DeFi, polo que o seu uso require un coñecemento profundo dos mesmos e en xeral da operativa financeira de produtos complexos. A natureza descentralizada de DeFi non elimina en absoluto os riscos asociados con estes instrumentos financeiros. Os usuarios deben comprender completamente os produtos nos que participan e sopesar coidadosamente os riscos e beneficios antes de involucrarse en estratexias DeFi complexas.

Un exemplo relevante na xestión de activos DeFi é Yearn Finance⁵. Trátase dun protocolo e plataforma DeFi que opera na cadea de bloques de Ethereum e que se converteu nun dos proxectos DeFi máis coñecidos. A misión principal de Yearn Finance é optimizar e automatizar os rendementos dos investimentos no espazo DeFi, utilizando estratexias avanzadas para maximizar os rendementos dos seus usuarios mediante a optimización da colocación de fondos en diferentes protocolos DeFi. Yearn Finance busca proporcionar unha experiencia de investimento sinxela e eficiente para os usuarios, permitíndolles depositar fondos no protocolo e deixar que os algoritmos xestionen automaticamente os investimentos oportunos para obter os mellores rendementos posibles.

Yearn Finance opera bóvedas (*vaults*), que son pools de liquidez xestionados de maneira proactiva. Os usuarios poden depositar fondos nestes vaults, e o protocolo utiliza estratexias automatizadas para maximizar os rendementos. Para iso, proporciona dúas funcións crave: *earn* (gañar) e *zap*. A función *earn* permite aos usuarios depositar stablecoins e outros activos en Yearn Finance para gañar intereses, mentres que *zap* permite aos usuarios realizar transaccións máis eficientes e optimizadas ao combinar múltiples operacións nunha soa transacción, buscando as mellores oportunidades de rendemento en cada caso. Ademais, Yearn Finance permite a participación de estratexias externas que poden propoñer e xestionar estratexias de investimento no protocolo.

O token nativo de Yearn Finance é YFI, que inicialmente se distribuíu de maneira equitativa á comunidade e permite aos posuidores participar na gobernanza do protocolo, propoñer e votar sobre cambios.

8.5 Retos e consideracións normativas

DeFi xurdiu como unha das aplicacións máis prometedoras da tecnoloxía blockchain, xa que as plataformas DeFi permiten aos usuarios realizar actividades financeiras de calquera tipo sen intermediarios. Con todo, estas plataformas tamén presentan desafíos únicos no que respecta a a escalabilidade, as vulnerabilidades de seguridade e os riscos asociados relacionados cos exploits de contratos intelixentes.

Un dos principais problemas das plataformas DeFi é a súa incapacidade para xestionar grandes volumes de transaccións simultaneamente, o que pode chegar a conxestionar a rede e retardar os tempos de procesamento e resposta

⁵yearn.fi

para os usuarios, algo que pode ser crítico no mundo das finanzas. Ademais, a medida que se desenvolven contratos intelixentes máis complexos, aumenta o risco de que se introduzan vulnerabilidades que poidan ser explotadas por axentes malintencionados. Hai que ter en conta que o código informático dos contratos intelixentes é público e pode ser analizado por calquera.

Outra preocupación relacionada coas plataformas DeFi é a súa susceptibilidade ás brechas de seguridade debido á súa dependencia de blockchains públicas e código fonte aberto. De feito, os piratas informáticos teñen atacado con éxito estes sistemas no pasado, provocando perdas significativas para os usuarios. Por último, os exploits de contratos intelixentes supoñen un risco importante para as plataformas DeFi, xa que poden provocar resultados inesperados ou mesmo a perda total dos fondos dos usuarios se non se deseñan e implementan correctamente. A pesar destes retos, DeFi segue crescendo rapidamente co lanzamento continuo de novos proxectos. A medida que o sector madura, espérase que os desenvolvedores sigan abordando estes problemas mediante mellores prácticas de codificación, mellores medidas de seguridade e protocolos de proba máis sólidos.

Con todo, esta rápida expansión tamén suscitou inquietude entre os reguladores, que debaten sobre como supervisar estas plataformas e protexer aos consumidores de posibles riscos. Un dos principais retos aos que se enfronta a regulación das DeFi é a natureza descentralizada destas plataformas blockchain, que dificulta a aplicación dos marcos reguladores tradicionais, xa que non existe unha autoridade central que controle a rede ou aos seus participantes. Ademais, moitos proxectos DeFi sitúanse formalmente en xurisdicións con normas laxas ou pouco claras ao redor das criptomoedas e os activos dixitais, o que dificulta a aplicación das normas de cumprimento.

Outro problema relacionado coa regulación das DeFi é a falta de transparencia que rodea a estas plataformas. Debido á súa estrutura descentralizada, pode ser difícil para os reguladores obter información sobre as actividades dos usuarios ou identificar posibles riscos, o que levou a algúns países a considerar a aplicación de requisitos máis estritos de coñecemento do cliente (KYC) e de loita contra o branqueo de diñeiro co fin de aumentar a supervisión destas plataformas.

Por último, exponse cuestións xurisdicionais en relación coa regulación do propio financiamento dos proxectos, xa que moitos deles operan en varios países e serían competencia de diversos organismos reguladores, en función de onde se atopen os usuarios ou os provedores de servizos, o que á súa vez crea confusión e posibles conflitos entre os distintos marcos xurídicos.

A pesar destes retos, producíronse algúns avances no desenvolvemento dun enfoque máis coherente da regulación das DeFi, con iniciativas como a do Grupo de Acción Financeira Internacional (*Financial Action Task Force*, FATF), que publicou orientacións sobre os activos dixitais (FATF, 2023), e reguladores como a Comisión do Mercado de Valores de Estados Unidos (SEC), que emprendeu accións contra determinados proxectos por supostas infraccións da lexislación sobre valores. A medida que o sector siga evolucionando, é probable que xurda máis claridade sobre como deben regularse as plataformas DeFi para equilibrar a innovación coa protección do investidor.

Capítulo 9

Blockchain como servizo na nube

Blockchain como servizo (Blockchain as a Service, BaaS) é un servizo baseado na nube que proporciona aos clientes unha plataforma para desenvolver, despregar e xestionar as súas aplicacións blockchain, sen necesidade de configurar e manter a súa propia infraestrutura de cadea de bloques (Khanna et al., 2022). Os provedores de BaaS ofrecen unha contorna de blockchain preconstruído que os clientes poden utilizar para crear, probar e despregar as súas propias aplicacións descentralizadas, a miúdo a través dunha interface web fácil de usar.

Os provedores de BaaS adoitan ofrecer unha gama de protocolos de cadea de bloques entre os que elixir, incluídas cadeas de bloques públicas como Ethereum ou Bitcoin, así como cadeas de bloques privadas como Hyperledger Fabric ou R3 Corda. Tamén proporcionan funcións como a xestión de nodos, algoritmos de consenso, xestión de identidades e almacenamento de datos, o que facilita que os clientes se centren en desenvolver os seus propias dApps lugar de dedicar recursos e tempo á xestión da infraestrutura.

BaaS é cada vez máis popular, xa que ofrece varios beneficios ás empresas e organizacións que buscan adoptar a tecnoloxía blockchain e en xeral entrar na Web 3.0. Estas vantaxes inclúen a redución de custos, unha posta en produción máis rápida, unha maior escalabilidade e unha seguridade mellorada. BaaS tamén permite ás empresas experimentar coa tecnoloxía blockchain antes de realizar investimentos significativos na súa propia infraestrutura blockchain.

En definitiva, BaaS é unha opción atractiva para as empresas que queren aproveitar as vantaxes da tecnoloxía blockchain sen ter que realizar grandes investimentos no desenvolvemento e a xestión dunha infraestrutura blockchain propia.

Cadro 9.1. Algúns provedores de blockchain como servizo na nube.

BaaS	Site
AWS Managed Blockchain	aws.amazon.com/managed-blockchain
Google Blockchain Node	cloud.google.com/blockchain-node-engine
IBM Blockchain Platform	ibm.com/blockchain
Microsoft Web3 Solution	azure.microsoft.com/solutions/web3
Oracle Blockchain	oracle.com/blockchain

Entre os actores máis relevantes do mercado de BaaS están os operadores de servizos na nube máis importantes (ver cadro 9.1). Amazon ofrece sobre a súa plataforma AWS unha variedade de servizos de blockchain, incluídas plataformas BaaS para Ethereum e Hyperledger Fabric. Google opera Blockchain Node Engine na Google Cloud, baseada na cadea de bloques Ethereum. Pola súa banda, IBM ofrece a súa propia gama de solucións blockchain, incluídas as plataformas BaaS para Hyperledger Fabric e Corda, mentres que Oracle ofrece unha plataforma BaaS chamada Oracle Blockchain para Hyperledger Fabric. Microsoft ofrece en Azure a solución Web3 para crear, iniciar e escalar solucións Web3 extremo a extremo. Outros actores relevantes no mercado de BaaS son SAP, Huawei, Baidu, Alibaba Cloud e Tencent Cloud.

9.1 Como funciona Blockchain como Servizo

Como acabamos de indicar, BaaS é unha solución que permite ás empresas utilizar a tecnoloxía de cadea de bloques sen necesidade de xestionar directamente a complexa infraestrutura que esta conleva. Do mesmo xeito que os servizos na nube permiten ás empresas almacenar datos ou executar aplicacións sen ter que manter servidores físicos, BaaS ofrece unha contorna xestionada onde as empresas poden desenvolver, implementar e operar aplicacións baseadas en blockchain. Isto significa que, en lugar de investir tempo e recursos en configurar e manter unha rede de blockchain, as empresas poden centrarse en aproveitar as vantaxes que esta tecnoloxía ofrece, como a seguridade, a transparencia e a inmutabilidade dos datos.

Na práctica, cando unha empresa utiliza BaaS, está a alugar o uso dunha infraestrutura blockchain operada e mantida por un provedor externo, como Microsoft, IBM ou Amazon. Estes provedores encárganse de todo o relacionado co funcionamento da blockchain, desde a xestión dos nodos (os servidores

que forman a rede) ata a implementación de medidas de seguridade robustas. Ademais, tamén ofrecen ferramentas fáciles de usar e paneis de control en liña para que as empresas poidan crear e personalizar as súas propias aplicacións de blockchain segundo as súas necesidades, sen requirir coñecementos técnicos avanzados.

Ao eliminar a necesidade de xestionar directamente a infraestrutura, BaaS reduce significativamente as barreiras de entrada para as empresas que desexan explorar e adoptar a tecnoloxía blockchain. Poden empezar con proxectos piloto para ir probando, e ir engadindo funcionalidades ou elementos a medida que se vaia gañando confianza coa tecnoloxía, progresando á velocidade máis adecuada segundo as necesidades e os recursos da empresa, todo iso sen preocuparse pola complexidade técnica subxacente. Isto fai que a tecnoloxía blockchain sexa accesible para un maior número de empresas, permitíndolles innovar e mellorar os seus procesos sen os desafíos e custos asociados coa xestión directa dunha rede de blockchain.

Usar BaaS ofrece moitas vantaxes ás empresas, especialmente a aquelas que desexan incorporar a tecnoloxía blockchain pero que non dispoñen dos coñecementos ou os recursos para abordar a súa complexidade técnica. Unha das principais vantaxes é a accesibilidade. BaaS democratiza o uso da tecnoloxía blockchain, permitindo que calquera empresa, independentemente do seu tamaño ou experiencia técnica, poida aproveitar os seus beneficios eliminando a necesidade de xestionar e manter unha infraestrutura blockchain propia. Con BaaS, calquera empresa pode centrarse en desenvolver aplicacións innovadoras sen preocuparse polos aspectos técnicos.

Outra vantaxe significativa é a redución de custos e de recursos necesarios para despregar aplicacións descentralizadas. Diseñar e despregar unha solución baseada en blockchain desde cero é certamente custoso e vai requirir dun tempo de desenvolvemento relativamente longo, xa que se necesita hardware especializado, persoal técnico capacitado e mantemento continuo. Con BaaS, as empresas poden evitar estes gastos iniciais e recorrentes, pagando só polo que se utiliza, de maneira similar a outros servizos dispoñibles na nube. Isto fai que a tecnoloxía blockchain sexa unha opción máis viable economicamente para proxectos de calquera tamaño.

Ademais, BaaS ofrece escalabilidade, algo que aínda hoxe é un reto importante en ámbito das solucións blockchain. As empresas poden comezar cun pequeno proxecto piloto e logo expandir o seu uso de blockchain segundo sexa necesario, sen ter que reconfigurar ou ampliar a infraestrutura. Os provedores de BaaS xestionan a capacidade da rede, asegurando que as aplicacións

poidan crecer sen grandes contratempos ou limitacións técnicas, o que é especialmente valioso para empresas en rápido crecemento.

Por último, a seguridade e confiabilidade son elementos moi importantes nos servizos BaaS. Os provedores de BaaS adoitan ter medidas de seguridade avanzadas e persoal especializado para garantir que as redes de blockchain sexan seguras e estean protexidas contra ameazas. Ademais, estas plataformas son en xeral moi fiables, ofrecendo unha alta dispoñibilidade e todo o soporte técnico necesario, o que permite ás empresas operar unha solución blockchain con garantías en canto á integridade dos seus datos e as operacións realizadas. En resumo, BaaS non só simplifica a adopción de blockchain, senón que tamén ofrece unha solución segura, escalable e rendible para as empresas.

9.2 Exemplos de plataformas BaaS

A continuación, exploraremos a modo de exemplo dúas das plataformas BaaS máis populares, que dalgún modo están revolucionado a maneira en que as empresas adoptan e utilizan a tecnoloxía blockchain. Analizaremos IBM Blockchain Platform e Amazon Managed Blockchain, coas súas características, vantaxes e casos de uso. Estas solucións non só simplifican o despregamento de blockchain, senón que tamén ofrecen ferramentas robustas e escalables que permiten ás empresas innovar e optimizar as súas operacións nunha ampla gama de sectores. A continuación, detallaremos como cada unha destas plataformas facilita a adopción de blockchain, integrándose con outros servizos e proporcionando unha contorna segura e eficiente para o desenvolvemento de aplicacións de blockchain.

9.2.1 IBM Blockchain Platform

IBM Blockchain Platform é unha solución BaaS baseada en Hyperledger Fabric, unha das plataformas de blockchain máis avanzadas e utilizadas no ámbito empresarial. Ofrece unha contorna completamente xestionada que facilita a adopción de blockchain sen necesidade de investir nunha infraestrutura complexa. Esta plataforma está deseñada para soportar aplicacións empresariais de misión crítica, garantindo altos niveis de seguridade, privacidade e rendementos.

Entre as funcionalidades principais de IBM Blockchain Platform atópase a súa capacidade para crear redes de blockchain permissionadas (ver capítulo 2),

onde só os participantes autorizados poden unirse e realizar transaccións, o cal é especialmente importante en contornas empresariais onde se require confidencialidade na xestión dos datos. A plataforma tamén ofrece ferramentas robustas para a xestión de identidades e acceso, asegurando que só os usuarios cos permisos adecuados poidan interactuar coa rede.

Do mesmo xeito que o servizo BaaS de Microsoft, esta solución tamén ofrece a xestión avanzada de contratos intelixentes, que permite ás empresas programar e automatizar transaccións e acordos dentro da rede blockchain. IBM Blockchain Platform inclúe unha contorna de desenvolvemento fácil de usar que facilita a creación, proba e implementación de contratos intelixentes, mesmo para aqueles que non teñen moita experiencia técnica. Ademais, a plataforma ofrece funcionalidades para definir modelos de gobernanza, facilitando a definición de regras sobre como se manexan as transaccións e como se toman as decisións dentro da rede.

Unha das vantaxes destacadas de IBM Blockchain Platform é o seu enfoque cara á interoperabilidade, xa que a plataforma está deseñada para integrarse facilmente con outros sistemas empresariais, permitindo que as solucións de blockchain funcionen en conxunto con outras tecnoloxías existentes na empresa. Ademais, a plataforma ofrece soporte e asistencia técnica ao longo de todo o proceso de despregamento, con respaldo experto en cada etapa do deseño e desenvolvemento das solucións de cadea de bloques. Este soporte inclúe desde a configuración inicial ata a resolución de problemas e a optimización continua da rede.

En canto a casos de uso, IBM Blockchain Platform foi adoptada nunha ampla gama de sectores. Por exemplo, no ámbito da saúde, foi utilizada para crear redes seguras para o intercambio de datos de pacientes entre hospitais e provedores de servizos, garantindo a privacidade e a integridade da información médica (Panwar et al., 2022). Na industria alimentaria, desenvóléronse solucións baseadas nesta plataforma para rastrexar a orixe e o percorrido dos produtos alimenticios, mellorando a rastrexabilidade e reducindo os riscos de contaminación (Patelli y Mandrioli, 2020). Outro exemplo significativo é o uso de IBM Blockchain Platform no sector dos servizos financeiros, onde se utilizou para desenvolver sistemas de compensación e liquidación máis rápidos e seguros, eliminando a necesidade de intermediarios e reducindo os custos operativos (Elghaish, Abrishami y Hosseini, 2020).

9.2.2 Amazon Managed Blockchain

Amazon Managed Blockchain é a plataforma BaaS ofrecida por Amazon Web Services (AWS), que admite os dous protocolos de blockchain máis populares: Hyperledger Fabric e Ethereum. Isto permite ás organizacións elixir a tecnoloxía que mellor se adapte ás súas necesidades específicas, xa sexa para redes privadas e permissionadas con Hyperledger Fabric, ou redes públicas e descentralizadas baseadas en Ethereum.

Entre as características principais de Amazon Managed Blockchain atópase a súa capacidade para automatizar a creación e xestión de redes de blockchain. Con só uns clics, é posible establecer unha rede de blockchain que inclúe a configuración de nodos, a provisión de hardware e software, e a xestión de certificados de seguridade. Esta automatización reduce significativamente a complexidade técnica e acelera o tempo de implementación, permitindo que as empresas comencen a utilizar blockchain máis rapidamente.

Outra funcionalidade clave é que está deseñada para escalar automaticamente segundo as necesidades da rede. Isto significa que a medida que a rede crece ou se incrementa a carga de transaccións, a plataforma pode axustarse sen problemas para manexar maiores volumes de traballo. Ademais, ofrece unha integración profunda con outros servizos de AWS. Por exemplo, a plataforma intégrase facilmente con Amazon CloudWatch para a monitorización e análise do rendemento da rede, o que permite ter unha visión completa de como se comporta a rede blockchain e axustar a configuración segundo sexa necesario. Tamén se integra con AWS Key Management Service (KMS), proporcionando unha xestión segura das claves criptográficas necesarias para protexer a información na rede blockchain. Estas integracións permiten ás empresas aproveitar o ecosistema completo de AWS, optimizando o seu uso de blockchain xunto con outras solucións na nube.

En canto á súa utilización práctica, Amazon Managed Blockchain tamén foi adoptado en varios sectores. No caso das cadeas de subministración, utilízase para crear redes que permiten rastrexar a orixe e o percorrido dos produtos, mellorando a transparencia e a eficiencia lóxística. Por exemplo, implementouse para rastrexar produtos agrícolas desde os provedores ata os consumidores finais, garantindo a autenticidade e reducindo os riscos de falsificación (Pérez et al., 2022). No ámbito dos servizos financeiros, a plataforma utilízase para desenvolver solucións que facilitan as transaccións entre múltiples partes de maneira segura e eficiente, con sistemas de liquidación e compensación que eliminan intermediarios, reducen custos e melloran a

velocidade das transaccións (Mitrevska et al., 2022). Outro exemplo é o seu uso en redes de intercambio de datos entre diferentes entidades, como na atención sanitaria, onde permite compartir información médica de maneira segura entre hospitais, clínicas e provedores de servizos, garantindo a privacidade e a integridade dos datos (Long, Choi y Coffman, 2022).

9.3 Como comezar con BaaS

O primeiro paso para comezar con Blockchain como Servizo consistiría en seleccionar a plataforma máis adecuada segundo as necesidades específicas de determinado proxecto ou empresa. Ao avaliar as opcións, é importante considerar varios factores. Por exemplo, é necesario identificar o protocolo de blockchain que mellor se adapte aos obxectivos do proxecto. Hyperledger Fabric é ideal para redes privadas ou permissionadas, mentres que Ethereum é máis adecuado para redes públicas descentralizadas. Ademais, hai que ter presente a escalabilidade do proxecto, é dicir, se hai que dar soporte a unha expansión rápida ou esperamos unha evolución máis pausada. A integración con outros servizos na nube que xa se estean utilizando, como bases de datos ou ferramentas de análise, tamén é relevante. Por exemplo, se xa se utilizan servizos de AWS, Amazon Managed Blockchain probablemente ofrezca unha mellor integración. O mesmo ocorrería con IBM Cloud e IBM Blockchain Platform. Finalmente, sería necesario estimar os custos de soporte técnico que ofrece cada plataforma para asegurarnos de que se axusta ao orzamento dispoñibles e aos requisitos do proxecto.

Unha vez seleccionada a plataforma de BaaS que mellor se adapta ao proxecto, o proceso adoita ser similar en calquera das plataformas BaaS de referencia:

1. Rexistro na plataforma, creando unha conta na plataforma de BaaS elixida. Isto implica rexistrarse na plataforma de nube correspondente.
2. Configuración inicial. Despois do rexistro, accederíase ao servizo de blockchain da plataforma. Aquí, deberase seleccionar o protocolo de blockchain que se desexa usar e configurar os detalles iniciais da rede, como o nome da rede, o número de nodos, e as políticas de gobernanza.
3. Creación da rede blockchain. Polo xeral seguindo paso a paso as instrucións proporcionadas pola plataforma, créase a rede blockchain. Isto

xeralmente implica seleccionar o tipo de rede (privada ou pública), configurar os nodos servidores que xestionan a rede e definir os permisos e roles dos participantes.

4. Despregamento e probas. Unha vez configurada a rede, é momento de despregala e comezar a desenvolver ou integrar aplicacións. As plataformas adoitan ofrecer ferramentas para probar a rede antes de poñela en produción, o que permite asegurarse de que todo funciona correctamente.

Comezar con BaaS pode parecer complexo, pero coa plataforma adecuada e os recursos de aprendizaxe apropiados, é posible aproveitar ao máximo esta tecnoloxía. Para sacar o máximo proveito de BaaS, as plataformas BaaS ofrecen documentación detallada nos seus sitios web oficiais, incluíndo guías paso a paso, referencias da interface do programador de aplicacións (API), e exemplos prácticos para familiarizarse coas funcións e características da plataforma. As empresas que proporcionan servizos BaaS a miúdo ofrecen tutoriais en vídeo e webinarios en vivo, onde expertos explican como usar a plataforma, comparten mellores prácticas e responden preguntas en tempo real. Ademais, as plataformas de aprendizaxe como Coursera, Udemy ou edX ofrecen cursos específicos sobre blockchain e a súa implementación a través de BaaS. Estes cursos cobren desde os conceptos básicos ata detalles avanzados sobre desenvolvemento e seguridade en blockchain.

Participar en comunidades en liña como Stack Overflow, GitHub, ou os foros oficiais das plataformas BaaS pode ser moi útil porque é posible atopar solucións a problemas comúns, obter consellos doutros usuarios, e compartir experiencias.

Capítulo 10

Recapitulación e tendencias futuras

A tecnoloxía blockchain, os contratos intelixentes e as aplicacións descentralizadas están a transformar rapidamente a forma en que interactuamos entre nós, traballamos, facemos negocios e mesmo nos divertimos. A tecnoloxía blockchain ofrece unha forma segura e transparente de almacenar e compartir datos sen necesidade de intermediarios, mentres que os contratos intelixentes permiten automatizar procesos empresariais complexos. As dApps tamén están a cambiar a forma en que utilizamos a tecnoloxía, permitíndonos interactuar entre nós dunha maneira máis descentralizada e democrática.

Para moitos, a xeneralización do uso das aplicacións descentralizadas e a tecnoloxía de cadeas de bloques supón o nacemento da terceira xeración de de a World Wide Web, chamada Web3 ou Web 3.0. Esta nova web caracterízase por basearse nun conxunto de tecnoloxías e protocolos que permiten a creación de aplicacións descentralizadas que poden executarse sobre redes blockchain. Estas aplicacións constrúense normalmente utilizando contratos intelixentes, que son programas informáticos autoexecutables que poden automatizar o intercambio de valor e información entre diferentes partes.

Ademais dos contratos intelixentes, Web3 tamén inclúe outras tecnoloxías como o almacenamento descentralizado, a xestión de identidades e os protocolos de comunicación descentralizados como os protocolos P2P. Estas tecnoloxías permiten crear aplicacións descentralizadas máis seguras, transparentes e fiables que as aplicacións centralizadas tradicionais.

A Web3 é considerada por moitos como a próxima evolución da internet, xa que ofrece unha nova forma de crear aplicacións e interactuar con elas, máis democrática, descentralizada e aberta. O obxectivo de Web3 é crear un

mundo dixital máis transparente e equitativo, no que os usuarios teñan máis control sobre os seus datos e interaccións en liña.

De todos os xeitos, as tecnoloxías nas que se basea a Web3 son aínda relativamente novas e, como vimos, existen varios riscos asociados ao seu uso. As aplicacións Web3 constrúense sobre redes blockchain, que xeralmente se consideran seguras. Con todo, os contratos intelixentes que alimentan estas aplicacións aínda poden ser vulnerables á piratería informática ou outras ameazas de seguridade, o que podería resultar na perda de fondos ou outros activos. Na mesma liña, as aplicacións descentralizadas da Web3 poden ser complexas e difíciles de construír e manter, o que pode aumentar o risco de erros ou fallos no código. As aplicacións Web3 atópanse aínda nas primeiras fases de desenvolvemento e adopción, e existe o risco de que non consigan unha adopción ou uso xeneralizados.

Por outra banda, o panorama regulatorio para as aplicacións Web3 aínda está a evolucionar, e existe o risco de que vaian aparecendo normas ou leis que poidan restrinxir o uso ou desenvolvemento destas aplicacións. Tamén existen dúbidas sobre a interoperabilidade das distintas propostas. Existen moitas redes e protocolos de blockchain diferentes, o que pode dificultar que as aplicacións Web3 poidan interaccionar entre si.

Aínda que quedan moitos retos por superar, como a escalabilidade, a interoperabilidade e a incerteza normativa, os beneficios potenciais destas tecnoloxías son evidentes. Desde a redución da fraude e o aumento da transparencia ata a habilitación de novos modelos de negocio e o empoderamento das persoas, blockchain, os contratos intelixentes e as aplicacións descentralizadas teñen o potencial de revolucionar a forma en que vivimos e traballamos.

A medida que estas tecnoloxías continúan evolucionando e madurando, é importante que os desenvolvedores, as empresas e os gobernos se manteñan informados e comprometidos cos últimos desenvolvementos. Traballando xuntos, podemos liberar todo o potencial da cadea de bloques, os contratos intelixentes e as aplicacións descentralizadas e crear un futuro máis equitativo, seguro e descentralizado.

10.1 Para saber máis

Dada a popularidade que está a alcanzar a tecnoloxía blockchain e os rápidos avances no seu desenvolvemento, podemos atopar un número importante de publicacións sobre o tema. A continuación enumeramos algunhas delas

que consideramos interesantes. Este material ofrece unha boa introdución á tecnoloxía blockchain e as súas posibles aplicacións en diversos sectores, así como unha análise crítica das súas vantaxes e limitacións.

- *Mastering Blockchain: Unlocking the Power of Cryptocurrencies, Smart Contracts, and Decentralized Applications*, de Lorne Lantz e Daniel Cawrey (2020), ofrece unha introdución completa á tecnoloxía blockchain e as súas aplicacións, cubrindo temas como Bitcoin, Ethereum, contratos intelixentes e aplicacións descentralizadas.
- *The Internet of Money*, de Andreas M. Antonopoulos (2016), é unha recompilación de charlas impartidas polo autor sobre Bitcoin e a tecnoloxía blockchain. Ofrece unha introdución accesible ao tema e explora as posibles implicacións desta tecnoloxía nos nosos sistemas financeiros.
- *Blockchain Basics: A Non-Technical Introduction in 25 Steps*, de Daniel Drescher (2017), ofrece unha introdución non técnica á tecnoloxía blockchain, explorando os fundamentos do seu funcionamento e as súas aplicacións potenciais en diversos sectores.
- *Blockchain Revolution: How the Technology Behind Bitcoin is Changing Money, Business, and the World*, de Don Tapscott e Alex Tapscott (2018), explora o impacto potencial da tecnoloxía blockchain en diversos sectores, desde as finanzas ata a sanidade e a administración pública. Tiene una versión en castellano (Tapscott y Tapscott, 2017).
- *The Truth Machine: The Blockchain and the Future of Everything*, de Paul Vigna e Michael J. Casey (2019), explora as posibles implicacións da tecnoloxía blockchain en diversos aspectos da sociedade, como as finanzas, a identidade e a gobernanza, e ofrece unha análise crítica dos seus posibles beneficios e limitacións.

Obviamente, tamén temos á nosa disposición innumerables tutoriais en liña sobre o tema con recursos para que calquera poida adquirir un coñecemento máis profundo da tecnoloxía blockchain e de como desenvolver aplicacións utilizando plataformas de cadeas de bloques. Entre os máis relevantes e asentados están:

- Os tutoriais de Solidity no sitio web de Ethereum¹. Ethereum é unha das plataformas blockchain máis populares, e Solidity é a linguaxe de progra-

¹ethereum.org/en/developers/tutorials

mación utilizado para escribir contratos intelixentes na rede Ethereum. Os tutoriais de Solidity no sitio web de Ethereum proporcionan unha guía paso a paso para aprender Solidity e desenvolver contratos intelixentes na rede Ethereum.

- IBM Blockchain 101². Este tutorial de IBM ofrece unha introdución á tecnoloxía blockchain, cubrindo temas como as redes descentralizadas, os contratos intelixentes e os mecanismos de consenso. O tutorial tamén inclúe un exercicio práctico que permite aos alumnos crear unha rede blockchain sinxela utilizando Hyperledger Fabric.
- Fundamentos de Blockchain en Coursera³. Este curso, ofrecido pola Universidade de Buffalo, proporciona unha introdución á tecnoloxía blockchain, cubrindo temas como a historia de blockchain, Bitcoin, Ethereum e contratos intelixentes. O curso inclúe videoconferencias e exercicios prácticos de programación.
- Documentación sobre Hyperledger Fabric⁴. Hyperledger Fabric é unha plataforma blockchain que se utiliza amplamente en aplicacións empresariais. A documentación de Hyperledger Fabric proporciona unha guía completa para que os desenvolvedores aprendan a desenvolver e implementar aplicacións blockchain utilizando esta plataforma.
- ConsenSys Academy⁵ ofrece unha serie de cursos en liña sobre tecnoloxía blockchain, que abarcan temas como Ethereum, Solidity e o desenvolvemento de aplicacións descentralizadas. Os cursos están deseñados tanto para principiantes como para desenvolvedores experimentados, e inclúen tanto clases en vídeo como exercicios prácticos.

²developer.ibm.com/technologies/blockchain/tutorials

³coursera.org/learn/blockchain-basics

⁴hyperledger-fabric.readthedocs.io

⁵consensys.io/academy

Apéndice A

Criptografía de clave pública

A criptografía de clave pública, tamén coñecida como criptografía asimétrica, é unha técnica criptográfica que utiliza un par de claves, unha pública e outra privada, para cifrar e descifrar datos. A clave pública utilízase para cifrar os datos, mentres que a privada utilízase para descifralos. Na criptografía de clave pública, a clave pública pode distribuírse libremente, mentres que o propietario mantén en segredo a clave privada. Cando un usuario quere enviar datos cifrados a outra persoa, utiliza a clave pública do destinatario para cifrar os datos. O destinatario utiliza entón a súa clave privada para descifrar os datos.

A criptografía de clave pública considérase máis segura que a criptografía simétrica, na que se utiliza a mesma clave para cifrar e descifrar, porque aínda que se intercepte a clave pública, os datos non poden descifrarse sen a clave privada.

A criptografía de clave pública utilízase nunha ampla gama de aplicacións, como a comunicación segura, as firmas dixitais e o acceso seguro a sitios web e servizos en liña. É un compoñente fundamental de moitas tecnoloxías modernas, como SSL/TLS, PGP e SSH. Todas as contornas de programación habituais, incluídas as utilizadas para desenvolver a tecnoloxía blockchain, dispoñen de bibliotecas criptográficas que entre outras moitas funcións proporcionan elementos para asinar, cifrar e descifrar mensaxes utilizando criptografía de clave pública. Por exemplo, o código JavaScript seguinte ilustra o uso da devandita biblioteca.

```
1 // Exemplo de cifrado e descifrado mediante
2 // criptografía de clave pública (PKI).
3 // Utilizamos o módulo crypto de Node.js.
4
```

 pki.js

```

5  const crypto = require('crypto');
6
7  // Función para cifrar unha mensaxe utilizando
8  // unha clave pública
9  function cifra(message, publicKey) {
10
11     // Creamos un búfer a partir da mensaxe para pasarllo
12     // á función de cifrado da librería crypto.
13     const buffer = Buffer.from(message, 'utf8');
14
15     // Ciframos a mensaxe. O método publicEncrypt cifra
16     // un buffer de bytes coa clave pública pasada como
17     // parámetro. Devolve un buffer co contido orixinal
18     // cifrado.
19     const encrypted = crypto.publicEncrypt(publicKey, buffer);
20
21     // Convertemos o buffer devolto pola función nunha
22     // cadea de caracteres e devolvémolo.
23     return encrypted.toString('base64');
24 }
25
26 // Función para descifrar unha mensaxe utilizando
27 // a clave privada
28 function descifra(encryptedMessage, privateKey) {
29
30     // Creamos un búfer a partir da mensaxe para pasarllo
31     // á función de descifrado da librería crypto.
32     const buffer = Buffer.from(encryptedMessage, 'base64');
33
34     // Desciframos a mensaxe
35     const decrypted = crypto.privateDecrypt(privateKey, buffer);
36
37     // Convertemos o buffer devolto pola función nunha
38     // cadea de caracteres e devolvémolo.
39     return decrypted.toString('utf8');
40 }

```

O segmento de código seguinte ilustra o uso das funcións `cifra` e `descifra` anteriores.

```

1  //
2  // Exemplo de uso das funcións cifra e descifra definidas
3  // anteriormente
4  //
5  // Xeramos un novo par de claves pública/privada coa
6  // librería crypto. Utilizamos o algoritmo RSA, con claves
7  // de 2048 bits

```

 pkiuse.js

```

8  const { publicKey, privateKey }
9    = crypto.generateKeyPairSync('rsa', {
10      modulusLength: 2048,
11    });
12
13  // Cifra unha mensaxe utilizando a clave pública xerada
14  const message = 'Este é a miña mensaxe secreta';
15  const encryptedMessage = cifra(message, publicKey);
16  console.log('Mensaxe cifrada: ${encryptedMessage}');
17
18  // Descifra a mensaxe cifrada utilizando a clave privada
19  // xerada. A mensaxe final será o mesmo que o do principio.
20  const decryptedMessage = descifra(encryptedMessage, privateKey);
21  console.log('Mensaxe en claro: ${decryptedMessage}');

```

A criptografía de clave pública tamén se pode utilizar para asinar contidos dixitais e con iso garantir a autoría e a non repudiación dun contido. Basicamente, para asinar un contido dixital, calcúlase unha clave hash do mesmo (ver apéndice B máis adiante) e se cifra o hash ca clave privada. O resultado pódese utilizar como sinatura, é dicir, como garantía de que o contido orixinal foi asinado polo propietario da clave privada. Para comprobalo, un receptor do documento orixinal pode calcular a clave hash do mesmo e descifrar a firma coa clave pública do asinante. Se ambos os valores coinciden, podemos asegurar que o documento orixinal foi asinado polo propietario da clave privada, xa que é a única persoa que a coñece.

O exemplo seguinte presenta unha función en JavaScript para asinar un ficheiro.

```

1  // Exemplo de firma mediante criptografía de clave pública (PKI). Definición dunha función de firma.
2  //
3  //
4  // Utilizamos o módulo crypto e o módulo fs de Node.js.
5  //
6  //
7  const fs = require('fs');
8  const crypto = require('crypto');
9
10 // A función toma dous parámetros: a referencia a un ficheiro que almacena a clave privada
11 // e a referencia ao ficheiro que queremos asinar.
12 // Como resultado, devolve a firma.
13 function signFileWithPKI(privateKeyPath, dataPath) {
14   // Lemos a clave privada do ficheiro.
15   const privateKey = fs.readFileSync(privateKeyPath, 'utf8');

```

 pki-sign.js

```

17 // Lemos o contido a asinar,
18 const data = fs.readFileSync(dataPath, 'utf8');
19
20 // Creamos un obxecto de firma utilizando un hash SHA-256.
21 const sign = crypto.createSign('SHA256');
22 // Engadimos os datos para asinar ao obxecto.
23 sign.write(data);
24 // Pechamos o obxecto.
25 sign.end();
26
27 // Asinamos o obxecto coa clave privada
28 // e devolvemos unha cadea codificada en base64
29 const signature = sign.sign(privateKey, 'base64');
30 return signature;
31 }

```

E o extracto de código seguinte ilustra a firma dun ficheiro e a verificación da devandita firma.

```

1 //  pki-sign-ex.js
2 // Exemplo de asinado dun ficheiro e de comprobación
3 // da firma usando PKI.
4 // Utilizamos o módulo crypto e o módulo fs de Node.js.
5 //
6 const fs = require('fs');
7 const crypto = require('crypto');
8
9 // Xeramos un par de claves pública-privada aptas
10 // para asinar.
11 const { privateKey, publicKey } =
12   crypto.generateKeyPairSync('rsa', {
13     modulusLength: 2048,
14     publicKeyEncoding: { type: 'spki', format: 'pem' },
15     privateKeyEncoding: { type: 'pkcs8', format: 'pem' },
16   });
17
18 // Gardamos as claves pública e privada
19 // en senllos ficheiros
20 fs.writeFileSync('private_key.pem', privateKey);
21 fs.writeFileSync('public_key.pem', publicKey);
22
23 // Chamamos á función anterior para asinar un ficheiro
24 // chamado 'data.txt'
25 const signature =
26   signFileWithPKI('private_key.pem', 'data.txt');
27
28 // A continuación imos verificar a firma.

```

```
29 // Creamos un obxecto de verificación SHA-256 (o mesmo
30 // protocolo que utilizamos na firma)
31 const verify = crypto.createVerify('SHA256');
32 // almacenamos o ficheiro no obxecto
33 verify.write(fs.readFileSync('data.txt', 'utf8'));
34 // e o pechamos.
35 verify.end();
36
37 // Utilizamos a clave pública para verificar a firma.
38 // A codificación utilizada é a mesma que utilizamos
39 // para codificar a firma.
40 const isVerified =
41     verify.verify(publicKey, signature, 'base64');
42
43 // Imprimimos o resultado na consola (true ou false)
44 console.log('Firma verificada:', isVerified);
```


Apéndice B

Funciones de hash

Unha función hash é unha función matemática que toma datos de entrada de calquera tamaño e produce unha saída de tamaño fixo chamada hash. A función hash xera unha pegada dixital única dos datos de entrada, que adoita ser unha cadea de caracteres ou bytes. As funcións hash utilízanse habitualmente en criptografía para garantir a integridade e seguridade dos datos. Comparando o hash dos datos orixinais co hash dos datos recibidos, pódese determinar se os datos foron alterados ou manipulados en tránsito.

As funcións hash están deseñadas para ser funcións unidireccionais, o que significa que é inviable desde o punto de vista computacional determinar os datos de entrada orixinais a partir da saída hash. Esta propiedade das funcións hash faias útiles en moitas aplicacións, como as firmas dixitais, o almacenamento de contrasinais e a validación de datos. As funcións hash tamén se utilizan na tecnoloxía blockchain para crear un libro de transaccións seguro e a proba de manipulacións. Cada bloque da cadea de bloques contén un hash do bloque anterior, creando unha cadea de bloques que non pode alterarse sen invalidar toda a cadea.

Algunhas funcións hash comúns son SHA-256 e MD5, aínda que a elección da función hash depende da aplicación específica e do nivel de seguridade requirido. O segmento de código JavaScript seguinte ilustra o cálculo dunha clave de hash SHA-256 utilizando o módulo Crypto de Node.js.

```
1 //  
2 // Exemplo de xeración dun hash SHA-256 dunha  
3 // cadea de caracteres  
4 //  
5 // Utilizamos a librería crypto de Node.js  
6 //  
7 const crypto = require('crypto');
```

 sha256.js

```
8 //
9 // A función sha256 toma unha cadea de texto como
10 // argumento e devolve o valor SHA-256 desa cadea como
11 // un valor hexadecimal.
12 //
13 // A función utiliza o módulo crypto incorporado
14 // en Node.js, que proporciona unha variedade de
15 // funcións criptográficas, incluíndo a función de hash SHA-256.
16 function sha256(text) {
17
18     // Creamos una instancia dun obxecto hash SHA-256
19     const hash = crypto.createHash('sha256');
20
21     // Actualizamos o hash almacenado no obxecto hash
22     // coa cadea de texto pasada como parámetro
23     hash.update(text);
24
25     // Devolvemos o valor da clave de hash (digest)
26     // almacenada no obxecto como resultado.
27     return hash.digest('hex');
28 }
```

Para calcular unha clave SHA-256 dunha imaxe almacenada nun ficheiro utilizaríamos o fragmento de código seguinte:

```
1 const fs = require('fs');
2
3 const nomFichero = 'Imagen.png';
4
5 // Lemos o ficheiro
6 const fileContents = fs.readFileSync(nomFichero);
7
8 // Invocamos a nosa función sha256
9 const sha256Value = sha256(fileContents);
10
11 console.log('A clave SHA-256 do ficheiro ${nomFichero} é:
12     ${sha256Value}');
```

 sha256inv.js

Apéndice C

Árbores de Merkle

Unha árbore de Merkle, tamén coñecida como árbore hash, é unha estrutura de datos utilizada en informática e criptografía para verificar eficazmente a integridade de grandes conxuntos de datos. Debe o seu nome ao seu inventor, Ralph Merkle (Merkle, 1990).

Unha árbore de Merkle é unha árbore binaria na que cada nodo que non é folla está etiquetado co hash das etiquetas dos seus nodos fillos. Cada folla da árbore representa un único dato, e o valor hash de cada folla calcúlase utilizando unha función hash criptográfica, como por exemplo SHA-256. A árbore constrúese mediante o hash recursivo de pares de nodos fillos, ata que se produce un único hash raíz.

O hash raíz dunha árbore de Merkle tamén se coñece como *raíz de Merkle* e serve como resumen de todo o conxunto de datos. Ao comparar a raíz Merkle dun conxunto de datos recibidos cunha raíz Merkle calculada previamente, un receptor pode verificar que o conxunto de datos non foi manipulado nin modificado. Para iso, solicítanse ao remitente os hashes dun subconxunto de follas e os seus correspondentes nodos pai, e utilízanse devanditos hashes para verificar a autenticidade da raíz de Merkle.

As árbores de Merkle utilízanse habitualmente na tecnoloxía blockchain para verificar eficazmente a integridade das transaccións dun bloque. Utilizando unha árbore de Merkle, os nodos da rede blockchain poden verificar rapidamente que un bloque de transaccións non foi manipulado nin modificado. Ademais, as árbores de Merkle axudan a reducir a cantidade de datos que deben transmitirse e almacenarse na rede blockchain, facéndoa máis eficiente e escalable.

O fragmento de código seguinte ilustra o cálculo dunha árbore de Merkle a partir dunha lista de transaccións.

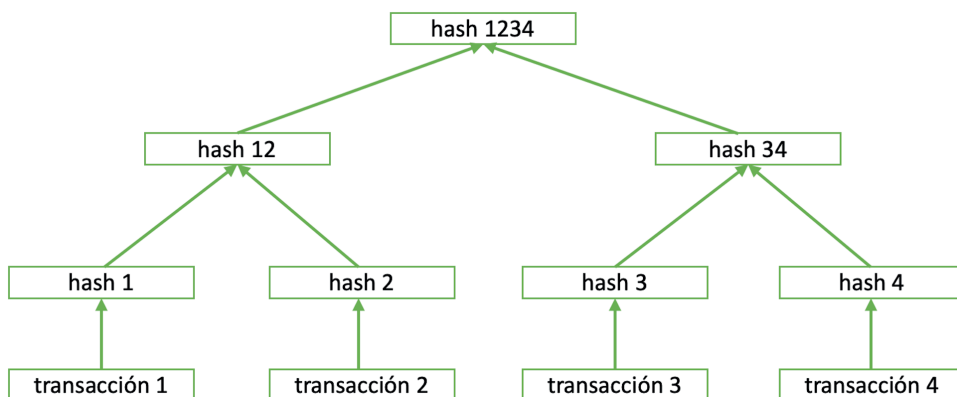


Figura C.1. Exemplo de árbore de Merkle de catro transaccións. A raíz da árbore é o que se almacenaría nun bloque para validar as transaccións.

```

1 | 📄 merkle.js
2 | // Este script utiliza o módulo crypto integrado en Node.js para
3 | // calcular hashes SHA-256 de cada transacción. Despois constrúe a
4 | // árbore Merkle de abaixo cara arriba combinando hashes adxacentes e
5 | // combinándoos ata que se obtén un único hash raíz. A función
6 | // calculaRaizMerkle() toma unha matriz de transaccións,
7 | // representadas mediante cadeas de caracteres, e devolve o hash
8 | // do nodo raíz da árbore de Merkle.
9 | //
10 | // Para probar o script, definimos un array de transaccións de
11 | // exemplo e chamamos á función calculaRaizMerkle() con elas. 0
12 | // hash raíz móstrase na consola.
13 | //
14 | const crypto = require('crypto');
15 |
16 | function calculaRaizMerkle(transacciones) {
17 |   // Converte cada cadea de transacción nun hash
18 |   const transactionHashes = transacciones.map((transaccion) => {
19 |     return crypto.createHash('sha256').
20 |       update(transaccion).digest('hex');
21 |   });
22 |
23 |   // Imos construír a árbore de Merkle de abaixo cara arriba
24 |   let nivel = transactionHashes;
25 |   while (nivel.length > 1) {
26 |     const siguienteNivel = [];
27 |     for (let i = 0; i < nivel.length; i += 2) {
28 |       const izquierda = nivel[i];
29 |       const derecha = i + 1 < nivel.length ? nivel[i + 1] : '';
30 |       const combinado =

```

```

31         crypto.createHash('sha256').
32             update(izquierda + derecha).digest('hex');
33         siguienteNivel.push(combinado);
34     }
35     nivel = siguienteNivel;
36 }
37
38 // Devolve a raíz da árbore de Merkle
39 return nivel[0];
40 }
41
42 // Exemplo de uso
43 const transaccions =
44     ['transacción 1',
45      'transacción 2',
46      'transacción 3',
47      'transacción 4'];
48
49 const raizMerkle = calculaRaizMerkle(transaccions);
50 console.log('Raíz Merkle:', raizMerkle);

```

Un grafo dirixido acíclico de Merkle ou DAG (Directed Acyclic Graph) de Merkle, é unha estrutura de datos máis xeral na que cada nodo pode ter múltiples nodos fillos, e cada nodo representa o hash dos seus nodos fillos. A estrutura é acíclica dirixida, o que significa que non hai ciclos no grafo. Os DAG de Merkle utilízanse no sistema de arquivos descentralizado IPFS para representar a estrutura de datos de direccións de contido para almacenar e recuperar datos.

Unha diferenza clave entre ambas as estruturas é que as árbores de Merkle adoitan ser árbores binarias, nos que cada nodo ten dous nodos fillos, mentres que os DAG de Merkle poden ter calquera número de nodos fillos. Ademais, os DAG de Merkle son máis flexibles e poden utilizarse para representar estruturas de datos máis complexas que as árbores de Merkle. Outra diferenza é que, nunha árbore de Merkle, todos os nodos folla están no mesmo nivel e adoitan estar dispostos nunha árbore binaria equilibrada. Nun DAG de Merkle, os nodos folla poden estar en calquera nivel e a estrutura pode ser máis complexa e variada.

En xeral, tanto as árbores de Merkle como os DAG de Merkle son estruturas de datos importantes que se utilizan para verificar a integridade de grandes conxuntos de datos mediante a creación dun resumo baseado en hash dos datos. A elección entre ambas estruturas depende do caso de uso específico e da complexidade dos conxuntos de datos que se quere representar.

Apéndice D

Prueba de participación en Ethereum

Os validadores implementan a proba de participación (proof-of-stake, PoS) en Ethereum de acordo con os pasos seguintes:

1. Achega de Ether. Os validadores deben depositar unha fianza de alomenos 32 ETH para participar no proceso de consenso. Este Ether bloquéase nun contrato de depósito, e utilízase para incentivar aos validadores a actuar con honestidade.
2. Proposta de bloques. Os validadores selecciónanse aleatoriamente para propoñer novos bloques á rede. Un bloque proposto contén unha lista de transaccións que o validador verificou.
3. Validación de bloques. Outros validadores verifican o bloque proposto e votan se o aceptan. Se máis do 50 % dos validadores votan a favor, o bloque engádese á cadea de bloques.
4. Pago de recompensas. Os validadores son recompensados con Ether por propoñer e validar bloques. Tamén se lles penaliza por propoñer ou validar bloques que finalmente se consideran non válidos.

Este proceso repítese continuamente, engadíndose novos bloques á cadea de bloques cada 12 segundos (en octube de 2023).

Os validadores selecciónanse para propoñer bloques mediante un proceso chamado *randao* (*Random Number Authority Oracle*, oráculo recoñecido para a xeración de números aleatorios), consistente na xeración de números aleatorios criptograficamente seguros que garanten que todos os validadores teñan as mesmas posibilidades de ser seleccionados. Randao funciona mediante a recompilación de contribucións de múltiples participantes que bloquean unha

cantidade de Ether nun contrato intelixente. Despois, estes participantes envían valores hash das súas contribucións e, finalmente, revelan a información orixinal. O valor hash agregado de todas as contribucións utilízase para xerar un número aleatorio.

Cada validador xera un número aleatorio utilizando a súa clave secreta e publica un hash dese número. A continuación, todos os hash combínanse para crear un único valor. En consecuencia, o valor de randao así calculado é característico e único do bloque validado.

Randao está deseñado para ser resistente á manipulación por parte de validadores individuais. Isto débese a que mesmo se un pequeno número de validadores tenta manipular o valor de randao, os seus esforzos veranse contrarrestados polos números aleatorios xerados polos demais validadores.

Un dos usos clave do número randao na proba de participación de Ethereum é seleccionar aleatoriamente aos validadores que son elixibles para crear novos bloques ou para propoñer bloques. O randao utilízase para asignar aos validadores responsabilidades específicas, como propoñer bloques ou dar fe da validez dos bloques. A aleatoriedade que proporciona randao axuda a garantir que a selección de validadores sexa impredecible e reduce o risco de colusión ou centralización, así como a equidade e a seguridade na rede.

Ademais de para seleccionar validadores que propoñan e validen bloques, tamén se utiliza para xerar recompensas por bloque. Canto máis Ether deposite un validador como fianza, máis probabilidades terá de ser seleccionado para propoñer e validar bloques, e maior será a súa parte da recompensa do bloque.

Cando un validador propón un bloque, tamén inclúe unha firma para demostrar que o validador depositou a cantidade necesaria de Ether e que verificou o bloque. Outros validadores verifican o bloque proposto comprobando a firma e asegurándose de que todas as transaccións do bloque son válidas. Unha vez que un bloque foi verificado e aceptado pola maioría dos validadores, engádese á cadea de bloques.

Recompénsase aos validadores con Ether por propoñer e validar bloques cunha cantidade de Ether proporcional á participación do validador, e do mesmo xeito penalízase aos validadores se propoñen ou validan bloques inválidos cunha penalización proporcional á gravidade da infracción. O randao tamén pode utilizarse para determinar as penalizacións, contribuíndo a que as condicións de penalización non sexan predicibles de antemán.

Este proceso desenvólvese nunha cadea paralela á cadea principal chamada cadea sentinela ou *beacon chain*. Esta cadea é responsable de implementar o consenso de proba de participación, coordinar aos validadores e mellorar a escalabilidade e a eficiencia na cadea de bloques de Ethereum, xa que a Beacon Chain é a base das cadeas de fragmentos, que dividirán a rede Ethereum en bloques máis pequenos e manexables. A Beacon Chain lanzouse en decembro de 2020, e fusionouse coa cadea de proba de traballo orixinal de Ethereum en setembro de 2022.

Randao en Ethereum é un intento de garantir a aleatoriedade nunha contorna onde as transaccións e a execución de contratos intelixentes son deterministas e transparentes. Con todo, é importante lembrar que, nalgúns casos, a aleatoriedade de randao pode verse comprometida se a maioría dos participantes actúan de maneira maliciosa. Por tanto, sempre é esencial considerar a seguridade e as implicacións ao utilizar sistemas de xeración de números aleatorios baseados en randao en aplicacións descentralizadas.

Apéndice E

Redes peer-to-peer

Unha rede entre iguais ou rede peer-to-peer (P2P) é un tipo de rede descentralizada na que todos os participantes teñen os mesmos privilexios e responsabilidades. Nunha rede P2P, cada nodo (ou participante) da rede pode actuar como cliente e como servidor, e pode comunicarse e intercambiar datos directamente con outros nodos sen necesidade de intermediarios. Nunha rede P2P non hai unha autoridade central ou un servidor que controle a rede, senón que cada nodo contribúe á funcionalidade da rede compartindo os seus recursos, como a súa potencia de cálculo, capacidade de almacenamento ou ancho de banda. Isto permite que as redes P2P sexan máis tolerantes a fallos, xa que non dependen dun único punto de fallo e poden seguir funcionando aínda que algúns nodos da rede teñan erros ou se desconecten.

As redes peer-to-peer xeralmente implementan algunha forma de rede virtual superposta sobre a topoloxía da internet, onde os nodos da superposición correspóndense cun subconjunto dos nodos da rede física. Os datos séguense intercambiando directamente a través da rede internet subxacente, pero na capa de aplicación os pares poden comunicarse entre si directamente, a través das ligazóns lóxicas de superposición, cada un dos cales corresponde a un roteiro a través da rede física subxacente. As ligazóns superpostas utilízanse para a indexación e o descubrimento de pares, e fan que o sistema P2P sexa independente da topoloxía da rede física. En función de como se vinculan os nodos entre si dentro da rede superposta, e de como se indexan e localizan os recursos, podemos clasificar as redes como non estruturadas ou estruturadas, ou como un híbrido entre ambas.

As redes peer-to-peer non estruturadas non impoñen unha estrutura particular na rede superposta, senón que están formadas por nodos que forman conexións ao azar entre si. Dado que non hai unha estrutura imposta globalmente, as redes non estruturadas son fáciles de construír e permiten optimizar

os recursos de maneira local en diferentes rexións da superposición. Ademais, dado que o papel de todos os pares na rede é o mesmo, as redes non estruturadas son moi robustas cando un gran número de pares únense e abandonan a rede con frecuencia.

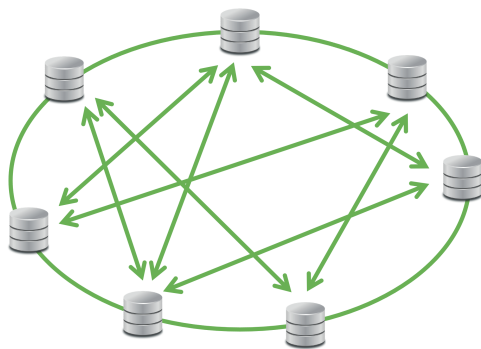


Figura E.1. Rede entre pares ou rede peer-to-peer (P2P). Cando un nodo únase á rede, establece conexións con algúns dos outros nodos. O protocolo da rede garante que todos os nodos participantes poden compartir información entre eles.

As principais limitacións das redes non estruturadas xorden precisamente desta falta de estrutura. Cando un usuario quere atopar unha información concreta na rede de pares, a consulta de procura débese diseminar a través de toda a rede para atopar tantos pares como sexa posible que compartan esa información. Esta diseminación por toda a rede de pares provoca unha cantidade moi elevada de tráfico de sinalización na rede, utiliza máis recursos de cálculo ao requirir que cada par procese todas as consultas de procura, e non garante que os datos se acaben localizando sempre. Ademais, como non hai correlación entre un par e o contido que ese par xestiona, non hai garantía de que a diseminación da procura atope un usuario que teña os datos desexados. É probable que os contidos populares estean dispoñibles en varios pares e que calquera par que os busque acabe atopándoos, pero se un usuario busca datos pouco frecuentes, compartidos só por uns poucos usuarios, é moi pouco probable que a procura teña éxito.

Nas redes estruturadas peer-to-peer, a superposición de nodos organízase nunha topoloxía específica, e o protocolo garante que calquera nodo poida buscar eficientemente determinado arquivo ou recurso compartido na rede, mesmo se o recurso é extremadamente pouco frecuente. O tipo máis común de redes P2P estruturadas implementa unha táboa de hash distribuída na que se asigna cada arquivo ou recurso a un peer concreto. Isto permite aos usuarios buscar recursos na rede utilizando dita táboa. Noutras palabras, as

relacións entre usuarios e recursos asignados almacénanse nunha táboa compartida, e calquera usuario da rede P2P pode recuperar eficientemente o recurso asociado a un usuario determinado.

De todos os xeitos, para enrutar o tráfico de forma eficiente a través da rede, os nodos dunha superposición estruturada deben manter listas de veciños que satisfagan criterios específicos. Isto faios menos robustos en redes cun alto índice de rotación, é dicir, cun gran número de usuarios que se unen e abandonan a rede con frecuencia.

As redes P2P poden utilizarse para diversas aplicacións, como o intercambio de arquivos, a mensaxería ou as chamadas de voz e vídeo. Un exemplo popular de rede P2P é BitTorrent, unha rede de intercambio de arquivos na que os usuarios comparten arquivos directamente entre si, en lugar de descargalos dun servidor central. As redes P2P tamén resoven moitos problemas aos que se enfrontan diversos sectores como as institucións públicas, as entidades financeiras, os mercados de capitais, as compañías de seguros, as cadeas de subministración, a mensaxería instantánea, a comunicación de voz, a transmisión de medios, a colaboración, a ciencia e a investigación, e a computación de alto rendemento, por nomear só algúns.

Ultimamente, as redes peer-to-peer experimentaron un rexurdir máis aló do intercambio de recursos multimedia: as criptomoedas baseadas en cadeas de bloques. Por exemplo, a criptomoeda Bitcoin xestiónase perante unha rede peer-to-peer construída sobre a propia internet. Está xestionada e controlada por todos os pares que aceptan participar. Todos os nodos están conectados nunha topoloxía plana, o que significa que non hai xerarquía, servizos centralizados, ou calquera cousa que implique unha autoridade central grazas ao modelo descentralizado das redes peer-to-peer.

Unha rede descentralizada presenta aspectos valiosos que resultaron fundamentais para a consolidación destas criptomoedas. Para empezar, en xeral son tecnoloxías robustas, gratuítas e de acceso aberto, de maneira que calquera poida unirse a elas e desenvolver novos contidos e aplicacións. Ademais son seguras e inmutables, e son abertas e transparentes á vez que proporcionan anonimato.

Bibliografía

- Abdallah, Salam y Nishara Nizamuddin (2023). «Blockchain-based solution for Pharma Supply Chain Industry». En: *Computers & Industrial Engineering* 177, página 108997. ISSN: 0360-8352. DOI: **10.1016/j.cie.2023.108997**.
- Abdelhady, Mohamed Ahmed (2019). *Hands-On Smart Contract Development with Hyperledger Fabric*. Packt Publishing.
- Adaś, Patryk (2021). *Mastering Web3.js*. Packt Publishing.
- Aggarwal, Shubhani y Neeraj Kumar (2021). «Chapter Twenty - Attacks on blockchain». En: *The Blockchain Technology for Secure and Smart Applications across Industry Verticals*. Editado por Shubhani Aggarwal, Neeraj Kumar y Pethuru Raj. Volumen 121. Advances in Computers. Elsevier, páginas 399-410. DOI: **<https://doi.org/10.1016/bs.adcom.2020.08.020>**.
- Ahn, Joongho, Eojin Yi y Moonsoo Kim (2024). «Ethereum 2.0 Hard Fork: Consensus Change and Market Efficiency». En: *The Journal of The British Blockchain Association*.
- Androutsellis-Theotokis, Stephanos y Diomidis Spinellis (dic. de 2004). «A survey of peer-to-peer content distribution technologies». En: *ACM Comput. Surv.* 36.4, páginas 335-371. ISSN: 0360-0300. DOI: **10.1145/1041680.1041681**.
- Antonopoulos, Andreas M. (2016). *The Internet of Money*. CreateSpace Independent Publishing Platform.
- Austria, Phillipe et al. (2021). «Performance and Cost Analysis of Sia, a Blockchain-Based Storage Platform». En: *2021 IEEE/ACIS 6th International Conference on Big Data, Cloud Computing, and Data Science (BCD)*, páginas 98-103. DOI: **10.1109/BCD51206.2021.9581866**.
- Bada, Abigael Okikijesu et al. (2021). «Towards a Green Blockchain: A Review of Consensus Mechanisms and their Energy Consumption». En: *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, páginas 503-511. DOI: **10.1109/DCOSS52077.2021.00083**.
- Baliga, Arati et al. (2018). «Performance Evaluation of the Quorum Blockchain Platform». En: *Arxiv*. DOI: **10.48550/arXiv.1809.03421**.

- Barger, Artem et al. (2021). «A Byzantine Fault-Tolerant Consensus Library for Hyperledger Fabric». En: *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, páginas 1-9. DOI: **10.1109/ICBC51069.2021.9461099**.
- Bauer, Davi Pedro (2022a). «Chainlink». En: *Getting Started with Ethereum: A Step-by-Step Guide to Becoming a Blockchain Developer*. Berkeley, CA: Apress, páginas 107-118. DOI: **10.1007/978-1-4842-8045-4_10**.
- (2022b). «Filecoin». En: *Getting Started with Ethereum: A Step-by-Step Guide to Becoming a Blockchain Developer*. Berkeley, CA: Apress, páginas 97-101. DOI: **10.1007/978-1-4842-8045-4_8**.
- Benet, Juan (2014). «IPFS - Content Addressed, Versioned, P2P File System». En: *arXiv preprint arXiv:1407.3561*.
- Bianchi, Daniele, Luca Rossini y Matteo Iacopini (2020). «Stablecoins and cryptocurrency returns: What is the role of Tether?». En: SSRN. DOI: **10.2139/ssrn.360545**.
- Blaut, Gabriel, Xuyang Ma y Katinka Wolter (2023). «Exploring Randomness in Blockchains». En: *2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, páginas 1-5. DOI: **10.1109/ICBC56567.2023.10174962**.
- Bouraga, Sarah (2022). «To Fork or Not To Fork? Bitcoin Forks' Success Analysis Using Twitter Data: Preliminary Results». En: *2022 4th Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS)*, páginas 13-16. DOI: **10.1109/BRAINS55737.2022.9908864**.
- Buterin, Vitalik (2014). *Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform*. ethereum.org.
- (2016). «Ethereum: platform review». En: *Opportunities and challenges for private and consortium blockchains* 45, páginas 1-45.
- Buterin, Vitalik y Nathan Schneider (2022). *Proof of Stake: The Making of Ethereum and the Philosophy of Blockchains*. Seven Stories Press.
- Carl, G. et al. (2006). «Denial-of-service attack-detection techniques». En: *IEEE Internet Computing* 10.1, páginas 82-89. DOI: **10.1109/MIC.2006.5**.
- Castro, Miguel y Barbara Liskov (nov. de 2002). «Practical Byzantine Fault Tolerance and Proactive Recovery». En: *ACM Trans. Comput. Syst.* 20.4, páginas 398-461. ISSN: 0734-2071. DOI: **10.1145/571637.571640**.
- Chain, Binance (2020). *Binance Smart Chain: A Parallel Binance Chain to Enable Smart Contracts*. Binance Chain.
- Chinnathambi, Kirupa (2018). *Learning React: A Hands-On Guide to Building Web Applications Using React and Redux*. Addison-Wesley Professional.

- Chittoda, J. (2019a). *Mastering Blockchain Programming with Solidity: Write production-ready smart contracts for Ethereum blockchain with Solidity*. Packt Publishing. ISBN: 9781839218637.
- Chittoda, Jitendra (2019b). *Mastering Blockchain Development with Solidity and Truffle*. Packt Publishing.
- Chohan, Usman W. (2021). «A History of Dogecoin». En: *Discussion Series: Notes on the 21st Century*. doi: **10.2139/ssrn.3091219**.
- Deirmentzoglou, Evangelos, Georgios Papakyriakopoulos y Constantinos Patsakis (2019). «A Survey on Long-Range Attacks for Proof of Stake Protocols». En: *IEEE Access* 7, páginas 28712-28725. doi: **10.1109/ACCESS.2019.2901858**.
- Delgado-von-Eitzen, Christian, Luis Anido-Rifón y Manuel J. Fernández-Iglesias (2021a). «Application of Blockchain in Education: GDPR-Compliant and Scalable Certification and Verification of Academic Information». En: *Applied Sciences* 11.10. ISSN: 2076-3417. doi: **10.3390/app11104537**.
- (2021b). «Blockchain Applications in Education: A Systematic Literature Review». En: *Applied Sciences* 11.24. ISSN: 2076-3417. doi: **10.3390/app112411811**.
- Delic, Amelia J. y Paul H. Delfabbro (2024). «Profiling the Potential Risks and Benefits of Emerging “Play to Earn” Games: a Qualitative Analysis of Players’ Experiences with Axie Infinity». En: *International Journal of Mental Health and Addiction* 22.1, páginas 634-647. doi: **10.1007/s11469-022-00894-y**.
- Divakaruni, Anantha y Peter Zimmerman (2023). «The Lightning Network: Turning Bitcoin into money». En: *Finance Research Letters* 52, página 103480. ISSN: 1544-6123. doi: **<https://doi.org/10.1016/j.frl.2022.103480>**.
- Dos Santos, Saulo et al. (2022). «A New Era of Blockchain-Powered Decentralized Finance (DeFi) - A Review». En: *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*, páginas 1286-1292. doi: **10.1109/COMPSAC54236.2022.00203**.
- Drescher, Daniel (2017). *Blockchain Basics: A Non-Technical Introduction in 25 Steps*. Apress.
- Elghaish, Faris, Sepehr Abrishami y M. Reza Hosseini (2020). «Integrated project delivery with blockchain: An automated financial system». En: *Automation in Construction* 114, página 103182. ISSN: 0926-5805. doi: **10.1016/j.autcon.2020.103182**.
- FATF (2023). *International Standards on Combating Money Laundering and the Financing of Terrorism & Proliferation*. Paris, France: FATF.

- Figueiredo, Sammy de et al. (2021). «Exploring the storj network: a security analysis». En: *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. SAC '21. Virtual Event, Republic of Korea: Association for Computing Machinery, páginas 257-264. ISBN: 9781450381048. doi: **10.1145/3412841.3441908**.
- Hartman, J.H., I. Murdock y T. Spalink (1999). «The Swarm scalable storage system». En: *Proceedings. 19th IEEE International Conference on Distributed Computing Systems (Cat. No.99CB37003)*, páginas 74-81. doi: **10.1109/ICDCS.1999.776508**.
- Heckmann, Oliver et al. (2004). *The eDonkey file-sharing network*. Gesellschaft für Informatik eV.
- Hoskinson, Charles (2017). *Why we are Building Cardano*. INPUT | OUTPUT.
- Infante, Roberto (2021). *Building Ethereum DApps: Decentralized Applications on the Ethereum Blockchain with Solidity and Hardhat*. Apress.
- Islam, Nazrul et al. (2023). «Is BlockChain Mining Profitable in the Long Run?». En: *IEEE Transactions on Engineering Management* 70.2, páginas 386-399. doi: **10.1109/TEM.2020.3045774**.
- Jafar, Uzma, Mohd Juzaidin Ab Aziz y Zarina Shukur (2021). «Blockchain for Electronic Voting System—Review and Open Research Challenges». En: *Sensors* 21.17. ISSN: 1424-8220. doi: **10.3390/s21175874**.
- Kalinin, Mikhail y Danny Ryan (2021). *Supplant DIFFICULTY opcode with PREVRANDAO Expose beacon chain randomness in the EVM by supplanting DIFFICULTY opcode semantics*. Ethereum Improvement Proposals EIP-4399. Ethereum.
- Kan, Gene (2002). «Gnutella». En: *Peer-to-Peer: Ökonomische, technologische und juristische Perspektiven*. Editado por Detlef Schoder, Kai Fischbach y René Teichmann. Berlin, Heidelberg: Springer Berlin Heidelberg, páginas 189-199. doi: **10.1007/978-3-642-56176-4_11**.
- Karantias, Kostis, Aggelos Kiayias y Dionysis Zindros (2020). «Proof-of-Burn». En: *Financial Cryptography and Data Security*. Editado por Joseph Bonneau y Nadia Heninger. Cham: Springer International Publishing, páginas 523-540. ISBN: 978-3-030-51280-4.
- Khanna, Abhirup et al. (2022). «Blockchain–Cloud Integration: A Survey». En: *Sensors* 22.14. ISSN: 1424-8220. doi: **10.3390/s22145238**.
- Kiffer, Lucianna, Dave Levin y Alan Mislove (2017). «Stick a fork in it: Analyzing the Ethereum network partition». En: *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*. HotNets '17. Palo Alto, CA, USA: Association for Computing Machinery, páginas 94-100. ISBN: 9781450355698. doi: **10.1145/3152434.3152449**.

- Klabnik, Steve y Carol Nichols (2019). *The Rust Programming Language*. No Starch Press.
- Kwon, Yujin et al. (2019). «Bitcoin vs. Bitcoin Cash: Coexistence or Downfall of Bitcoin Cash?». En: *2019 IEEE Symposium on Security and Privacy (SP)*, páginas 935-951. doi: **10.1109/SP.2019.00075**.
- Lamport, Leslie, Robert Shostak y Marshall Pease (jul. de 1982). «The Byzantine Generals Problem». En: *ACM Trans. Program. Lang. Syst.* 4.3, páginas 382-401. ISSN: 0164-0925. doi: **10.1145/357172.357176**.
- (2019). «The Byzantine generals problem». En: *Concurrency: The Works of Leslie Lamport*. New York, NY, USA: Association for Computing Machinery, páginas 203-226. ISBN: 9781450372701.
- Lantz, Lorne y Daniel Cawrey (2020). *Mastering Blockchain: Unlocking the Power of Cryptocurrencies, Smart Contracts, and Decentralized Applications*. O'Reilly Media.
- Lin, Shijie (2023). «Proof of work vs. Proof of stake in cryptocurrency». En: *Highlights in Science, Engineering and Technology* 39, páginas 953-961. doi: **10.54097/hset.v39i.6683**.
- Long, Audrey, Daniel Choi y Joel Coffman (2022). «Using Amazon Managed Blockchain for ePHI An Analysis of Hyperledger Fabric and Ethereum». En: *2022 IEEE World AI IoT Congress (AllIoT)*, páginas 276-282. doi: **10.1109/AIIoT54504.2022.9817198**.
- Ma, Guang kai, Chunpeng Ge y Lu Zhou (2020). «Achieving reliable timestamp in the bitcoin platform». En: *Peer-to-Peer Networking and Applications* 13.6, páginas 2251-2259. doi: **10.1007/s12083-020-00905-6**.
- Mahmoodi, Sina (2020). *Vyper Smart Contract Programming*. Packt Publishing.
- Manolache, Manuel Adelin, Sergiu Manolache y Nicolae Tapus (2022). «Decision Making using the Blockchain Proof of Authority Consensus». En: *Procedia Computer Science* 199. The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 & 2021): Developing Global Digital Economy after COVID-19, páginas 580-588. ISSN: 1877-0509. doi: **<https://doi.org/10.1016/j.procs.2022.01.071>**.
- Merkle, Ralph C. (1990). «A Certified Digital Signature». En: *Advances in Cryptology – CRYPTO' 89 Proceedings*. Editado por Gilles Brassard. New York, NY: Springer New York, páginas 218-238. ISBN: 978-0-387-34805-6.
- Miller, Andrew (2019). «Permissioned and permissionless blockchains». En: *Blockchain for distributed systems security*. Editado por Sachin Shetty, Charles A. Kamhoua y Laurent L. Njilla. IEEE Press, páginas 193-204.

- Mitrevska, Svetlana et al. (2022). «Blockchain as a Service, an overview on AWS and its BaaS». En: *2022 30th Telecommunications Forum (TELFOR)*, páginas 1-4. doi: **10.1109/TELFOR56187.2022.9983746**.
- Nakamoto, Satoshi (2008a). *Bitcoin: A Peer-to-Peer Electronic Cash System*. www.bitcoin.org.
- (2008b). *Bitcoin. A peer-to-peer electronic cash system*. Informe técnico. www.bitcoin.org.
- Nick, Jonas, Andrew Poelstra y Gregory Sanders (2020). «Liquid: A bitcoin sidechain». En: *Liquid white paper*.
- Panwar, Arvind et al. (2022). «A Blockchain Framework to Secure Personal Health Record (PHR) in IBM Cloud-Based Data Lake». En: *Computational Intelligence and Neuroscience 2022.1*, página 3045107. doi: **10.1155/2022/3045107**.
- Passaglia, Andrea (2017). *Vue.js 2 Cookbook*. Packt Publishing.
- Patelli, Niccolò y Mauro Mandrioli (2020). «Blockchain technology and traceability in the agrifood industry». En: *Journal of Food Science* 85.11, páginas 3670-3678. doi: **10.1111/1750-3841.15477**.
- Paul, Greig, Fraser Hutchison y James Irvine (mayo de 2014). «Security of the MaidSafe Vault Network». En: *Procs. of the Wireless World Research Forum Meeting 32 (WWRF32)*.
- Pérez, Daniel et al. (2022). «Traceability System for an Agricultural Supply Network based on Blockchain». En: *2022 IEEE International Conference on Automation/XXV Congress of the Chilean Association of Automatic Control (ICA-ACCA)*, páginas 1-6. doi: **10.1109/ICA-ACCA56767.2022.10006240**.
- Pierro, Giuseppe Antonio y Roberto Tonelli (2022). «Can Solana be the Solution to the Blockchain Scalability Problem?» En: *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, páginas 1219-1226. doi: **10.1109/SANER53432.2022.00144**.
- Reed, Jeff (2017). *Litecoin: An Introduction to Litecoin Cryptocurrency and Litecoin Mining*. North Charleston, SC, USA: CreateSpace Independent Publishing Platform. ISBN: 1974692949.
- Salah, Khaled et al. (2019). «Blockchain-Based Soybean Traceability in Agricultural Supply Chain». En: *IEEE Access* 7, páginas 73295-73305. doi: **10.1109/ACCESS.2019.2918000**.
- Sanka, Abdurrahshid Ibrahim y Ray C.C. Cheung (2021). «A systematic review of blockchain scalability: Issues, solutions, analysis and future research». En: *Journal of Network and Computer Applications* 195, página 103232. ISSN: 1084-8045. doi: <https://doi.org/10.1016/j.jnca.2021.103232>.

- Schreiber, Flora Rheta (1973). *Sybil: The Classic True Story of a Woman Possessed by Sixteen Separate Personalities*. Grand Central Publishing.
- Schulte, Stefan et al. (2019). «Towards Blockchain Interoperability». En: *Business Process Management: Blockchain and Central and Eastern Europe Forum*. Editado por Claudio Di Ciccio et al. Cham: Springer International Publishing, páginas 3-10. doi: **10.1007/978-3-030-30429-4_1**.
- Schwarz Müller, Maximilian (2018). *Material-UI: React components for faster and easier web development*. Packt Publishing Ltd.
- (2021). *Angular 12: The Complete Guide*. Packt Publishing.
- Shanaev, Savva et al. (2019). «Cryptocurrency Value and 51 % Attacks: Evidence from Event Studies». En: *The Journal of Alternative Investments* 22.3, páginas 65-77. ISSN: 1520-3255. doi: **10.3905/jai.2019.1.081**. eprint: <https://jai.pm-research.com/content/22/3/65.full.pdf>.
- Sidorenko, E. L. (2020). «Stablecoin as a New Financial Instrument». En: *Digital Age: Chances, Challenges and Future*. Editado por Svetlana Igorevna Ashmarina, Marek Vochozka y Valentina Vyacheslavovna Mantulenko. Cham: Springer International Publishing, páginas 630-638. ISBN: 978-3-030-27015-5.
- Singh, Niharika (2020). *Ganache: Personal Blockchain for Ethereum Development*. Packt Publishing.
- Skh Saad, Sheikh Munir y Raja Zahilah Raja Mohd Radzi (nov. de 2020). «Comparative Review of the Blockchain Consensus Algorithm Between Proof of Stake (POS) and Delegated Proof of Stake (DPOS)». En: *International Journal of Innovative Computing* 10.2. doi: **10.11113/ijic.v10n2.272**.
- Snider, Myles, Kyle Samani y Tushar Jain (2018). «Delegated Proof of Stake: Features and Tradeoffs». En: *Multicoins Capital* March 2, 2018.
- Spurlock, Jake (2021). *Bootstrap 5: Building Responsive Layouts with HTML and CSS*. Packt Publishing.
- Tapscott, Don y Alex Tapscott (2017). *La revolución blockchain: Descubre cómo esta nueva tecnología transformará la economía global*. Deusto.
- (2018). *Blockchain Revolution: How the Technology Behind Bitcoin is Changing Money, Business, and the World*. Portfolio.
- Thibault, Louis Tremblay, Tom Sarry y Abdelhakim Senhaji Hafid (2022). «Blockchain Scaling Using Rollups: A Comprehensive Survey». En: *IEEE Access* 10, páginas 93039-93054. doi: **10.1109/ACCESS.2022.3200051**.
- Towaha, Syed Omar Faruk (2019). *Learn Ethereum DApp Development with Remix*. Packt Publishing.
- Vigna, Paul y Michael J. Casey (2019). *The Truth Machine: The Blockchain and the Future of Everything*. Picador.

- Wang, Qin et al. (2020). «Blockchain for the IoT and industrial IoT: A review». En: *Internet of Things* 10. Special Issue of the Elsevier IoT Journal on Blockchain Applications in IoT Environments, página 100081. ISSN: 2542-6605. DOI: <https://doi.org/10.1016/j.iot.2019.100081>.
- Wang, Wenbo et al. (2019). «A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks». En: *IEEE Access* 7, páginas 22328-22370. DOI: **10.1109/ACCESS.2019.2896108**.
- Wood, Gavin (2016). *Polkadot: Vision for a Heterogeneous Multi-chain Framework*. Polkadot.
- (2024). *Ethereum: A Secure Decentralised Generalised Teansaction Ledger*. Yellow Paper Shanghai Version. Ethereum.
- Xiao, Bingbing et al. (2022). «Proof of Importance: A Consensus Algorithm for Importance Based on Dynamic Authorization». En: *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*. WI-IAT '21. Melbourne, VIC, Australia: Association for Computing Machinery, páginas 510-513. ISBN: 9781450391870. DOI: **10.1145/3498851.3499007**.
- Yakovenko, Anatoly (2017). *Solana: A new architecture for a high performance blockchain*. Solana.
- Yuan, Yong y Fei-Yue Wang (2018). «Blockchain and Cryptocurrencies: Model, Techniques, and Applications». En: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 48.9, páginas 1421-1428. DOI: **10.1109/TSMC.2018.2854904**.
- Yves-Christian, Adja Elloh et al. (2018). «Total Eclipse: How To Completely Isolate a Bitcoin Peer». En: *2018 Third International Conference on Security of Smart Cities, Industrial Control System and Communications (SSIC)*, páginas 1-7. DOI: **10.1109/SSIC.2018.8556790**.
- Zhang, Chao et al. (2011). «Unraveling the BitTorrent Ecosystem». En: *IEEE Transactions on Parallel and Distributed Systems* 22.7, páginas 1164-1177. DOI: **10.1109/TPDS.2010.123**.
- Zilliqa Team (2019). *Scilla: A Smart Contract Intermediate Level Language*. Zilliqa Research Pte. Ltd.

Glosario

acuñado

Proceso de creación de novos tokens, especialmente tokens non fungibles (NFT). O termo tamén se utiliza para referirse ao proceso de creación de novas unidades dunha criptomoeda.

algoritmo hash seguro (SHA)

Algoritmo criptográfico utilizado para xerar valores hash únicos e garantir a integridade e autenticidade dos datos na cadea de bloques.

almacenamento en frío

A práctica de manter criptomoedas fóse do mundo dixital, nunha contorna segura, para protexelas contra a piratería ou o roubo.

altcoin

Calquera criptomoeda distinta de Bitcoin.

altura

Posición dun bloque na rede blockchain, determinada polo número de bloques que se engadiron á blockchain desde o bloque xénese.

aplicación descentralizada (dApp)

Aplicación que se executa nunha rede blockchain e non está controlada por ningunha autoridade central.

árbore de Merkle

Estrutura de datos utilizada na cadea de bloques para organizar e resumir as transaccións ou datos dun bloque, baseándose nunha serie de cálculos hash.

ataque de adianto

Tipo de ataque no que un atacante ten coñecemento avanzado dunha transacción pendente e pode manipular a orde das transaccións no seu beneficio.

ataque de eclipsado

Tipo de ataque no que un atacante toma o control da conexión dunha vítima á rede para manipular a súa visión da rede.

ataque de longo alcance

Tipo de ataque no que un atacante faise co control de todo ou gran parte do historial dunha blockchain, o que lle permite reescribir transaccións.

ataque de reentrada

Tipo de ataque no que un atacante explota un contrato intelixente que fai múltiples chamadas a outros contratos, permitíndolle executar o seu propio código múltiples veces e drenar os fondos do contrato.

ataque de Sybil

Tipo de ataque no que un atacante crea múltiples identidades falsas nunha rede co fin de obter o control sobre ela.

ataque do 51 %

Tipo de ataque a unha rede blockchain no que unha soa entidade ou grupo de entidades controla máis do 50 % dos recursos da rede (por exemplo, potencia de cálculo, nodos, criptomoeda...). Con este control maioritario, o atacante pode potencialmente manipular a rede impedindo confirmacións de transaccións, cometendo dobre gasto ou reescribindo o historial de transaccións.

autoridade de certificación

Terceiro de confianza que emite e verifica os certificados dixitais utilizados para a comunicación segura.

BaaS

Blockchain as a Service, un modelo de servizo que permite a empresas e desenvolvedores usar infraestrutura blockchain na nube sen necesidade de xestionala directamente.

Bitcoin

A primeira e máis coñecida criptomoeda, creada en 2009. O seu símbolo é BTC.

blockchain

Tecnoloxía descentralizada e distribuída que permite transaccións seguras e transparentes entre pares.

bloque

Conxunto de transaccións verificadas que se engaden á cadea de bloques.

bloque canónico

Bloque máis recente que é aceptado como válido por todos os nodos da rede.

bloque xénese

Primeiro bloque da cadea de bloques, creado polo fundador ou fundadores da rede e que contén unha transacción especial que xera a subministración inicial da criptomoeda.

cadea lateral

Ver sidechain.

carteira

Aplicación que permite aos usuarios almacenar, xestionar e transferir criptomoedas nunha rede blockchain.

circuíto integrado de aplicación específica (ASIC)

Chip informático especializado, deseñado especificamente para minar criptomoedas.

clave pública

Clave criptográfica utilizada para cifrar datos e verificar firmas dixitais na cadea de bloques, a miúdo asociada cun moedeiro ou dirección específica.

coin

Unidade de criptomoeda.

coinbase

A primeira transacción nun bloque que crea nova criptomoeda e recompensa ao mineiro que resolveu o bloque.

colisión hash

Situación na que dúas entradas diferentes producen o mesmo valor hash, o que é moi raro e considérase un risco de seguridade na cadea de bloques.

confirmación

O proceso de verificar unha transacción na cadea de bloques para asegurar que é válida e está rexistrada no libro maior.

consenso

Acordo xeral entre todos os nodos da cadea de bloques sobre o estado do libro maior e a validez das transaccións.

consorcio

Grupo de individuos ou organizacións que se unen para compartir recursos e colaborar nunha rede blockchain.

contorna de desenvolvemento integrado (IDE)

Aplicación que proporciona amplas facilidades aos programadores informáticos para o desenvolvemento de software.

contrato intelixente

Código autoexecutable que aplica automaticamente os termos dun contrato entre partes nunha rede blockchain.

criptografía

A práctica de usar algoritmos matemáticos e protocolos para asegurar e verificar transaccións na cadea de bloques.

criptomoeda

Moeda dixital ou virtual que utiliza criptografía para a seguridade e opera independentemente dun banco central.

custodia

Responsabilidade que unha entidade ten sobre o almacenamento seguro de criptomoedas ou activos dixitais en nome doutra persoa ou entidade, habitualmente mediante a protección proporcionada polas claves privadas dun sistema de criptografía de clave pública.

código aberto

Tipo de software que está dispoñible libremente e pode ser modificado e distribuído por calquera, a miúdo utilizado para proxectos blockchain para fomentar a colaboración e a innovación.

dApp

Ver aplicación descentralizada.

DeFi

Sistemas financeiros construídos sobre tecnoloxía blockchain que operan de maneira descentralizada sen intermediarios tradicionais.

denegación de servizo (DOUS)

Tipo de ataque no que un atacante asolaga unha rede ou servidor con tráfico, facendo que non estea dispoñible para os usuarios lexítimos.

desbordamento de enteiro

Condición que ocorre cando o resultado dunha operación aritmética excede o tamaño máximo que pode ser representado por un tipo enteiro.

descentralización

O principio de distribuír o control e a autoridade sobre unha rede blockchain entre moitos nodos diferentes, en lugar de depender dunha autoridade central.

DEX

Intercambio descentralizado. Plataforma de intercambio que opera sen unha entidade central e permite aos usuarios comerciar directamente entre si.

dificultade

Medida dos recursos necesarios para minar un novo bloque na cadea de bloques. Axústase periodicamente para manter unha taxa constante de produción de bloques.

dirección

Identificador único nunha blockchain que representa a localización dunha carteira, NFT ou contrato intelixente.

dobre gasto

Acto de tentar gastar a mesma criptomoeda máis dunha vez, o que se impide na cadea de bloques mediante consenso e protocolos criptográficos.

entre pares (P2P)

Tipo de arquitectura de rede na que os nodos comunícanse e realizan transaccións directamente entre si, sen necesidade de intermediarios ou autoridades centrais, a miúdo utilizada en redes blockchain.

escalabilidade

Capacidade da rede blockchain para manexar un crecente volume de transaccións e novos usuarios sen comprometer o rendemento ou a seguridade. Un reto crave para os desenvolvedores de blockchain.

escrow

Contrato de depósito no cal un terceiro mantén e regula o pago dos fondos necesarios para as dúas partes involucradas nunha transacción, liberándoos só cando se cumpren certas condicións.

Ether

criptomoeda nativa da cadea de bloques Ethereum, utilizada para pagar as taxas das transaccións e a execución de contratos intelixentes. O seu símbolo é ETH.

Ethereum

Plataforma blockchain descentralizada que permite a creación de contratos intelixentes e aplicacións descentralizadas (dApps).

exchange

Plataforma onde os usuarios poden comprar, vender e intercambiar criptomoedas con outros usuarios ou coa propia plataforma.

finanzas descentralizadas (DeFi)

Uso da tecnoloxía blockchain para crear aplicacións financeiras descentralizadas abertas, transparentes e accesibles a todo o mundo.

firma dixital

Esquema matemático utilizado para verificar a autenticidade e integridade dunha mensaxe, transacción ou documento na cadea de bloques.

fork

División na rede blockchain que crea dúas versións separadas do libro maior. Isto pode deberse a condicións de carreira na creación de bloques, diferenzas nas regras de consenso ou actualizacións do protocolo.

fragmentación

Método de partición de datos utilizado en redes blockchain para mellorar a escalabilidade, dividindo a rede en fragmentos (shards) que procesan transaccións en paralelo.

gas

Unidade de medida utilizada para calcular o custo de execución dunha transacción ou contrato intelixente na blockchain de Ethereum.

gas fee

Ver tarifa de transacción.

gobernanza

Proceso mediante o cal os titulares de tokens participan na toma de decisións sobre o desenvolvemento e cambios no protocolo.

grafo acíclico dirixido (DAG)

Estrutura de datos utilizada na tecnoloxía blockchain e nos sistemas de arquivos descentralizados para organizar e resumir conxuntos de datos.

gráfico acíclico dirixido de Merkle (DAG)

DAG utilizado en IPFS no que cada nodo está etiquetaxe co hash dos seus datos e o hash dos seus fillos, formando unha estrutura similar á árbore de Merkle.

halving

Redución programada da recompensa por minar un novo bloque na cadea de bloques, que se produce periodicamente para manter unha oferta finita e deflacionaria da criptomoeda.

hard fork

División permanente na rede blockchain que crea unha nova versión do libro maior, a miúdo causada por cambios fundamentais nas regras de consenso ou actualizacións de protocolo.

hash

Función matemática que toma unha entrada (datos) e produce unha saída de tamaño fixo (valor ou clave hash), utilizada para identificar datos de forma única na blockchain.

hashrate

Ver taxa de hash.

holdear

Estratexia consistente en manter unha criptomoeda durante un longo período de tempo, normalmente coa esperanza de obter importantes ganancias nun futuro.

Hyperledger

Plataforma blockchain de código aberto mantida pola Fundación Linux, deseñada para uso empresarial e centrada na privacidade, a escalabilidade e a interoperabilidade.

infraestrutura de clave pública (PKI)

Sistema que utiliza claves privadas e públicas para asegurar e autenticar transaccións e datos na cadea de bloques, a miúdo utilizado para a xestión da identidade dixital.

inmutabilidade

Propiedade dunha blockchain de que unha vez que unha transacción ou dato engadiuse a mesma, non pode ser modificado ou borrado, asegurando a integridade e autenticidade do libro maior.

interoperabilidade

Capacidade que teñen redes de blockchain diferentes para interactuar e compartir información de forma segura e eficiente entre si.

latencia

Tempo que transcorre entre o envío dunha transacción ou petición e a recepción dunha resposta na rede blockchain, afectando a velocidade de confirmación de transaccións.

libro maior

Sistema de rexistro que rastrea as transaccións e os datos na cadea de bloques, mantido polos nodos da rede.

liquidez

Facilidade coa que un activo DeFi pode ser comprado ou vendido no mercado sen afectar significativamente o seu prezo.

mainnet

Versión viva e operativa da rede blockchain que está aberta ao público e utilízase para transaccións reais e almacenamento de datos.

manipulacións, a proba de

Propiedade dun sistema que deixa claro cando se tentou modificalo ou manipulalo.

manipulacións, resistente a

Propiedade dun sistema que impide a modificación ou manipulación non autorizada.

memecoin

Ver mememoeda.

mememoeda

Moeda dixital creada principalmente como unha broma ou para seguir unha tendencia viral, baseada en memes ou referencias da cultura popular na internet.

mempool

Espazo onde as transaccións non confirmadas almacénanse temporalmente polos nodos, ata que son incluídas nun bloque.

mercado secundario

Plataforma ou lugar onde tokens ou criptomoedas previamente emitidos mércanse ou véndense entre investidores, sen a intervención directa do emisor orixinal.

MetaMask

Extensión do navegador e aplicación móbil que permite aos usuarios interactuar con aplicacións descentralizadas (dApps) na cadea de bloques Ethereum e outras redes compatibles. Tamén serve como moedeiro para esas redes.

metaverso

Universo ou mundo virtual, normalmente creado e xestionado por unha plataforma baseada en blockchain. O metaverso é un espazo virtual compartido no que os usuarios poden interactuar entre si, explorar contornas dixitais e participar en diversas actividades como xogos, socialización e comercio. O metaverso concíbese a miúdo como unha versión inmersiva e descentralizada da internet, coa súa propia economía e sistemas de gobernanza.

micropago

Transacción de moi pequeno valor que se realiza a través da blockchain, xeralmente usada para pagar servizos de baixo custo.

mineiro

Participante na rede blockchain que utiliza potencia computacional para validar transaccións, crear novos bloques e obter recompensas en forma de criptomoeda.

minería

Proceso de utilizar potencia computacional para validar transaccións, crear novos bloques e gañar recompensas en forma de criptomoeda nunha rede blockchain.

minería de liquidez

No contexto DeFi, refírese á obtención de tokens mediante a participación en protocolos que distribúen recompensas.

moeda dixital de banco central (Central Bank Dixital Currency, CBDC)

Moeda dixital emitida e apoiada polo banco central dun país. Polo tanto, representa unha forma dixital de diñeiro fiduciario.

multifirma

Tipo de firma dixital que require que varias partes asinen unha transacción ou autoricen unha transferencia de fondos na cadea de bloques, utilizada con fins de seguridade e xestión de riscos.

máquina de estados

Modelo matemático utilizado para describir o comportamento dun sistema, como unha rede blockchain.

máquina virtual

Contorna de software que emula un sistema informático e permite a execución de contratos intelixentes nunha rede blockchain.

nodo

Participante na rede blockchain que valida as transaccións, mantén unha copia do libro de contabilidade e comunícase con outros nodos para garantir a seguridade e integridade da rede.

nodo completo

Nodo na rede blockchain que almacena unha copia completa do libro maior e participa na verificación e validación de transaccións.

nodo lixeiro

Nodo na rede blockchain que almacena só unha copia parcial do libro maior, confiando nos nodos completos para a verificación e validación das transaccións.

nonce

Número incremental ou aleatorio xerado por un mineiro durante o proceso de minado dun bloque na blockchain, utilizado para crear un valor hash único que cumpra o obxectivo de dificultade.

oferta de token de seguridade (Security Token Offer, STO)

Lanzamento de tokens que representan valores financeiros, como accións ou bonos, e que están suxeitos a regulacións financeiras.

oferta inicial de exchange (Initial Exchange Offer, IEO)

Método de recadación de fondos onde un exchange ou portal de criptomonedas actúa como intermediario para lanzar tokens e facilitar a venda destes aos investidores. Úsase para obter fondos para financiar novas empresas ou negocios.

oferta inicial de moedas (Initial Coin Offer, ICO)

Proceso polo cal unha nova criptomoeda ou token lánzase ao público para recadar fondos, xeralmente nunha etapa temperá do desenvolvemento dun proxecto blockchain.

organización autónoma descentralizada (DAO)

Unha organización dirixida por código informático nunha rede blockchain, en lugar de por xestores ou líderes humanos.

oráculo

Servizo ou mecanismo de terceiros que proporciona datos do mundo real á blockchain, a miúdo utilizado para facilitar a execución de contratos intelixentes e a toma de decisións na blockchain.

perda impermanente

A perda de valor que os provedores de liquidez poden experimentar debido á volatilidade nos prezos dos activos.

pool de transaccións

Conxunto de transaccións non confirmadas á espera de ser procesadas e engadidas a unha rede blockchain.

prezo do gas

A cantidade de Ether necesaria para pagar cada unidade de gas utilizada nunha transacción ou execución de contrato intelixente na blockchain de Ethereum.

privada (blockchain)

Tipo de rede blockchain na que o acceso e a participación están restrinxidos a participantes ou entidades autorizados, a miúdo utilizada para aplicacións empresariais e gobernamentais.

proba de autoridade (PoA)

Mecanismo de consenso no que os validadores selecciónanse para validar transaccións baseándose na súa identidade ou reputación, a miúdo utilizado en blockchains privadas ou empresariais.

proba de cobertura (PoC)

Mecanismo de consenso no que os validadores demostran que teñen a cobertura de seguro necesaria para apoiar as súas transaccións ou actividades na blockchain, utilizado nas industrias financeira e de seguros.

proba de custodia

Mecanismo que asegura que un nodo nunha rede de almacenamento descentralizada, como Swarm, segue almacenando e mantendo os datos de forma correcta e segura durante un período de tempo.

proba de espazo e tempo (PoSaT)

Mecanismo de consenso no que os validadores demostran o seu compromiso de espazo de almacenamento e recursos de rede, utilizado na cadea de bloques Chia.

proba de importancia (PoI)

Mecanismo de consenso no que os validadores demostran a súa importancia e contribución á rede baseándose no seu historial de transaccións e outros factores, utilizado na cadea de bloques NEM.

proba de liquidez (PoL)

Mecanismo de consenso no que os validadores demostran a súa liquidez e capacidade para achegar fondos ou activos suficientes á rede, utilizado en mercados dixitais descentralizados e fondos de liquidez.

proba de Merkle

Proba criptográfica que permite a un nodo da rede blockchain verificar a inclusión dunha transacción ou datos nun bloque sen ter que descargar e validar o bloque completo.

proba de participación (PoS)

Mecanismo de consenso no que os validadores avalan con criptomoeda a súa capacidade para validar transaccións, utilizada en Ethereum 2.0 e outras cadeas de bloques.

proba de participación delegada (DPoS)

Algoritmo de consenso utilizado nalgúñas redes blockchain, onde os posuidores de criptomoeda poden votar para seleccionar un conxunto de delegados que son responsables de verificar as transaccións e engadir novos bloques á cadea.

proba de peso (PoW)

Mecanismo de consenso no que os validadores demostran a súa reputación e importancia na rede en función da súa contribución e actividade, utilizado na cadea de bloques Algorand.

proba de replicación

Mecanismo de verificación utilizado nalgunhas redes de almacenamento descentralizado, como Filecoin, que proba que un proveedor ten replicado un arquivo no seu sistema.

proba de traballo (PoW)

Mecanismo de consenso no que os validadores resolven complexos problemas matemáticos para validar transaccións e obter recompensas, utilizado en Bitcoin e outras cadeas de bloques.

profundidade

Número de bloques que se engadiron á cadea de bloques despois dunha transacción concreta, utilizado para medir o nivel de confirmación ou seguridade das transaccións.

protección contra repeticións

Mecanismo para evitar que actores maliciosos reproduzan ou dupliquen transaccións válidas na cadea de bloques, a miúdo utilizado en hard forks e actualizacións da rede.

préstamo instantáneo

Un tipo de préstamo DeFi que permite aos usuarios pedir prestado e devolver unha cantidade de fondos nunha soa transacción.

pública (blockchain)

Tipo de rede blockchain na que o acceso e a participación están abertos a calquera, sen necesidade de aprobación ou autenticación, a miúdo utilizada para aplicacións de criptomoedas.

raíz de Merkle

Hash do nodo principal dunha árbore de Merkle, que resume todas as transaccións ou datos dun bloque e inclúese na cabeceira do bloque.

recompensa

Incentivo ou compensación que obteñen os validadores pola súa contribución á rede, a miúdo en forma de criptomoeda ou tokens.

rede

Conxunto de nodos e participantes conectados á cadea de bloques, que xuntos validan as transaccións, manteñen o libro maior e garanten a seguridade e integridade da rede.

reversión

A reversión dun ou máis bloques na cadea. Isto pode ocorrer debido a un erro de software, un ataque á rede, ou outros eventos inesperados que causan unha diverxencia entre o estado real da cadea de bloques e o que a maioría dos nodos cren. As reversións poden ter graves consecuencias para a integridade e fiabilidade da cadea de bloques.

security token

Token criptográfico que representa un activo, como accións dunha empresa, e está suxeito á normativa sobre valores.

sen confianza

Propiedade dunha rede blockchain que permite ás partes realizar transaccións entre si sen depender da confianza nunha autoridade central ou intermediario.

sharding

Ver fragmentación.

sidechain

Blockchain paralela a unha blockchain principal que se conecta a ela para transferir activos entre ambas, permitindo maior escalabilidade e funcionalidade.

sistema de arquivos interplanetario (Interplanetary File System, IPFS)

Rede de almacenamento e compartición de arquivos peer-to-peer descentralizada que permite almacenar e acceder a contido de maneira distribuída.

Solidity

Linguaxe de programación utilizado para escribir contratos intelixentes na blockchain Ethereum.

stablecoin

Token estable. Tipo de criptomoeda deseñada para manter un valor estable fronte a unha moeda fiduciaria ou outro activo (por exemplo, o euro ou o dólar estadounidense).

stacking

Proceso polo cal os posuidores de criptomoedas bloquean unha cantidade dos seus fondos para apoiar a seguridade e operacións dunha rede blockchain, recibindo recompensas a cambio.

subdesbordamiento de enteiros

Condición que ocorre cando o resultado dunha operación aritmética é menor que o tamaño mínimo que pode ser representado por un tipo enteiro.

swap

Intercambio directo dunha criptomoeda ou token por outro entre dúas partes, xeralmente a través dun contrato intelixente.

swap atómico

Intercambio descentralizado de criptomoedas ou tokens entre dúas blockchains diferentes sen a necesidade dun intermediario.

Tangle

Tecnoloxía de libro maior distribuído utilizada pola rede IOTA que utiliza un grafo acíclico dirixido (directed acyclic graph, DAG) en lugar dunha cadea de bloques.

tarifa de gas

Ver tarifa de transacción.

tarifa de transacción

Taxa paga polo remitente dunha transacción para incentivar aos participantes da rede para procesar e validar a transacción.

taxa de hash

Velocidade á que un mineiro pode realizar cálculos hash no proceso de minería de criptomoeda na cadea de bloques.

testnet

Rede blockchain utilizada para probas e desenvolvemento, separada da rede blockchain principal.

token

Unidade de valor que se crea e xestiona nunha rede blockchain (NFT, criptomoeda....

token fungible

Activo dixital que pode ser intercambiado por outro activo do mesmo tipo sen perder valor, xa que todos os tokens desa clase son idénticos. Un exemplo típico é unha criptomoeda.

token non fungible (NFT)

Tipo de activo dixital na blockchain que é único e non pode ser intercambiado por outros tokens ou activos un polo outro, a miúdo utilizado para colecciónables, xogos e arte.

tokenización

Proceso de conversión dun activo físico ou dixital nun token nunha rede blockchain.

tokenómica

Estudo dos aspectos económicos e financeiros dos tokens nunha rede blockchain, incluíndo a emisión, distribución e factores que afectan o seu valor.

tolerancia a fallos bizantinos (BFT)

Capacidade dun sistema de seguir funcionando aínda que algúns dos seus compoñentes fallen ou se comporten de forma inesperada. Tamén, un algoritmo de consenso que é tolerante a fallos e seguro contra un certo número de nodos bizantinos (é dicir, maliciosos).

tolerancia práctica a fallos bizantinos (PBFT)

Algoritmo de consenso que permite a unha rede distribuída de nodos poñerse de acordo sobre o estado dun sistema, mesmo se algúns nodos fallan ou se comportan maliciosamente. Unha variación máis eficiente da tolerancia a fallos bizantinos na que os nodos se turnan para crear e validar bloques.

transacción

Transferencia de valor entre dous partes nunha rede de cadea de bloques.

táboa de hash distribuída (DHT)

Sistema descentralizado para almacenar e recuperar pares crave-valor. Nunha DHT, os nodos organízanse nunha rede P2P, e cada nodo é responsable de almacenar e servir un subconxunto de claves. Cando un nodo necesita recuperar un valor para unha clave dada, utiliza unha función hash para determinar que nodo é responsable desa clave e, a continuación, envía unha solicitude a ese nodo.

unidade de procesamento gráfico (GPU)

Chip informático especializado deseñado para o procesamento eficiente e de alta velocidade de gráficos e outros datos complexos, a miúdo utilizado na minería de criptomoeda.

validación

Proceso mediante o cal as transaccións nunha blockchain verifícanse polos nodos da rede para asegurar a súa lexitimidade antes de ser engadidas ao libro maior da cadea de bloques.

wallet

Ver carteira.

web3

Web 3.0, a próxima xeración da internet, que se concibe como unha rede descentralizada e distribuída de aplicacións e servizos que utilizan a tecnoloxía blockchain e outros protocolos descentralizados.

wei

A unidade máis pequena de Ether (ETH), a criptomoeda utilizada na blockchain de Ethereum. 1 Ether equivale a 10^{18} wei. Esta unidade utilízase para representar cantidades moi pequenas de Ether. Por exemplo, se o prezo do gas para unha transacción é de 20 gwei, significa que o custo de executar esa transacción é de 20×10^6 wei ou 0,00000002 ETH.

xerais bizantinos

Termo utilizado para describir un escenario no que un grupo de xerais debe coordinar un ataque, pero algúns deles poden ser traidores que enviarán mensaxes falsas.

yield farming

Ver minería de liquidez.

Índice de materias

- Aave, 61, 162
- ABI, 83
- activo
 - dixital, 51
 - sintético, 123, 163
- Alchemy, 85, 146
- almacenamiento descentralizado, 177
- Amazon Managed Blockchain, 174
- Angular, 88
- anonimato, 68
- aplicación descentralizada, 76
- Application Binary Interface, 83
- árbore
 - de Merkle, 189
 - hash, 189
- arquitecturas de oráculos, 122
- ataque
 - de denegación de servicio, 48, 103
 - de eclipsado, 48
 - de largo alcance, 46
 - de reentrada, 101
 - de secuestro de marca de tiempo, 48
 - de Sybil, 47
 - do 51 %, 46
 - dos nodos bizantinos, 47
- auditoría de código, 103
- Axie Infinity, 137
- BaaS, 169
- Band Protocol, 132
- Basic Attention Token, 61
- beacon chain, 195
- bibliotecas maliciosas, 103
- bifurcación, 41, 53
- Binance, 62
 - Smart Chain, 81, 139
 - Smart Chain Oracle, 133
- Bitcoin, 51, 169
- blockchain
 - como servicio, 169
 - de capa 2, 42
 - de proba, 81
 - empresarial, 28
 - privada, 27
 - pública, 26
- bloque, 20
- Bootstrap, 88
- bóveda, 165
- Brave, 61
- cadea
 - de bloques, 13

- de consorcio, 27
 - lateral, 43
 - sentinela, 195
- carteira, 62
- CBDC, 70
- censura, 76, 113
- chaincode, 79
- Chainlink, 61, 130, 132
- CID, 114
- clave
 - privada, 181
 - pública, 181
- Coinbase, 62
- cold wallet, 62
- Compound, 161
- Consello de Estabilidade Financeira, 67
- consenso, 17
- consumo enerxético, 68
- contorna de desenvolvemento integrada, 81
- contrato intelixente, 53, 75, 119, 139
- Corda, 169
- creación dunha criptomoeda, 60
- criptografía
 - asimétrica, 181
 - de clave pública, 181
- criptomoeda, 51
 - ecolóxica, 68
- crowdfunding, 65
- Curve Finance, 160
- custodia, 63
 - en frío, 63
 - en quente, 63
- DAG, 114, 191
- Dai, 61
- DAO, 25, 42, 76
- dApp, 76
- Decentraland, 61
- DeFi, 64, 78, 123, 157
- delito informático, 68
- desbordamento de enteiros, 101
- descentralización, 53, 109
- desenvolvedor de software, 24
- determinismo, 128
- DEX, 158
- direccionamiento por contido, 109
- dobre gasto, 52
- Dogecoin, 55
- eficiencia, 104
 - operativa, 45
- ERC1155, 148
- ERC20, 98
- ERC721, 143
- escalabilidade, 42, 45, 178
- especulación, 66
- ETH, 93
- Ether, 57, 93
- Ethereum, 53, 56, 79, 81, 139, 169, 193
- Europol, 68
- EVM, 90
- exchange, 61
- explorador de blockchain, 69
- fallback, 97
- fallos bizantinos, 30
- Filecoin, 116
- finanzas descentralizadas, 78, 157
- Flow, 139
- fork, 41
- forxado, 37
- front running, 101
- Función Aleatoria Verificable, 129
- función hash, 187
- funxibilidade, 59
- Ganache, 81
- gas, 89

- fee, 64
- Go, 79
- grafo acíclico dirixido
 - de Merkle, 114
- grafo dirixido acíclico, 191
- gwei, 90
- Hardhat, 82
- hash, 128, 189
- hashrate, 36
- hot wallet, 62
- Hyperledger Fabric, 79, 169
- IBM Blockchain Platform, 172
- ICO, 65
- IDE, 81
- impacto ambiental, 67
- inclusión financeira, 53
- índice sintético, 163
- inmutabilidade, 44, 104
- Internet das cousas, 119
- interoperabilidade, 133, 178
- IPFS, 112, 140
- JavaScript, 79
- JSON, 140
- Kraken, 62
- lavado de diñeiro, 68
- layer 2, 42
- libro maior, 13, 61
- liquidez, 162
- Litecoin, 53
- Maidsafe, 117
- mainnet, 87
- MakerDAO, 133, 162
- marco regulatorio, 70
- MD5, 187
- mecanismo de consenso, 28
- mememoeda, 55
- mempool, 18
- mercados financeiros
 - descentralizados, 158
- micropago, 65
- mineiro, 18, 22
- minería, 32, 61, 68
 - de liquidez, 162
- modelo publicación-subscrición, 122
- modelo solicitude-resposta, 122
- Monero, 65
- Máquina Virtual de Ethereum, 83
- NFT, 65, 116, 137
- Nifty Gateway, 139
- nodo, 21
 - de almacenamento, 110
 - de direccionamiento, 111
 - de procura, 111
- número pseudoaleatorio, 128
- oferta inicial de criptomoeda, 65
- opcións, 163
- OpenSea, 139
- OpenZeppelin, 89, 98
- organizacións autónomas
 - descentralizadas, 76
- oráculo, 119
 - baseado en hardware, 120
 - baseado en software, 120
 - centralizado, 121
 - descentralizado, 121
 - intercadena, 133
- P2P, 107, 177, 197
- PAT, 98
- PATACON, 60, 98
- peer-to-peer, 197
- privacidade, 65
- proba

- de custodia, 111
- de participación, 29, 68, 193
- de participación delegada, 29
- de queima, 30
- de replicación, 112
- de trabajo, 29, 55, 67
- préstamos descentralizados, 161
- randao, 129, 193
- rastrejo de transacciones, 68
- raíz de Merkle, 189
- rede
 - de oráculos, 132
 - entre iguais, 197
 - P2P estructurada, 198
 - P2P non estructurada, 197
- regulador, 24, 67, 178
- remesas, 65
- Remix, 81
- Rust, 79
- Satoshi Nakamoto, 52, 60
- Scilla, 79
- Script, 55
- Sepolia, 83, 140
- SHA-256, 55, 187
- Shiba Inu, 61
- Sia, 117
- sidechain, 43
- sistema de archivos
 - descentralizado, 107
 - interplanetario, 112
 - peer-to-peer, 107
- SOL, 57
- Solana, 57
- Solidity, 79, 140
- stablecoin, 57, 70
- staking, 163
- Storj, 117
- subdesbordamento de enteros, 101
- superposición de nodos, 198
- SuperRare, 139
- SushiSwap, 160
- sustentabilidade, 67
- Swarm, 117
- tarifa en gas, 64
- taxa de hash, 36
- testnet, 81
- Tether, 57
- token
 - apalancado, 163
 - fungible, 53, 59, 98
 - inverso, 164
 - non fungible, 137
- tokenización, 59
- tokenización de activos, 65
- tolerancia a fallos, 109
 - bizantinos, 31
- transacciones, 13
- transparencia, 157
- TRON Oracle, 133
- Truffle, 81
- Uniswap, 61, 133, 159
- USD Coin, 61
- validación, 27
 - de entradas, 103
- validador, 18
- validadores, 193
- vault, 165
- volatilidade, 66
- VRF, 130
- Vue.js, 88
- Vyper, 79
- wallet, 62
- Web3, 177

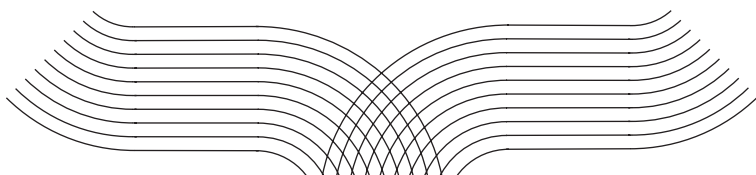
wei, 143

Wrapped Bitcoin, 61

Yearn Finance, 162

YFI, 165

Zcash, 65



Monografías

Serie Científico-Tecnolóxica

Últimas publicacións na colección

Correspondencia entre Fermat e Pascal. Tradución comentada (2024)

José Nicanor Alonso Álvarez, Miguel Ángel Mirás Calvo e Carmen Quinteiro Sandomingo

O libro que ninguén puido ler: Sobre as revolucións dos orbes celestes (2021)

José Nicanor Alonso Álvarez, Raúl Gómez Pato e Miguel Ángel Mirás Calvo

A rama atmosférica do ciclo hidrolóxico. Dende a evaporación oceánica ate a precipitación nos continentes (2021)

Marta Vázquez, Iago Algarra, Jorge Eiras-Barca e Rogert Sorí

Turbomáquinas hidráulicas (2019)

Concepción Paz Penín, Eduardo Suárez Porto, Miguel Concheiro Castiñeira e Marcos Conde Fontenla

A lista de Hilbert (2018)

José Nicanor Alonso Álvarez



Blockchain, criptomoedas, NFT e aplicacións descentralizadas

Fundamentos tecnolóxicos

A pesar do rebumbio que xeraron conceptos como as criptomoedas ou os tokens non funxibles (NFT), as cadeas de bloques representan unha revolución silenciosa que está a remodelar o tecido da nosa sociedade dixital. Esta tecnoloxía, froito da innovación e do desexo dun mundo máis transparente e equitativo, promete transformar non só as transaccións financeiras ou introducir un innovador xeito de entender o coleccionismo, senón tamén a forma en que interactuamos, confiamos e colaboramos no ámbito dixital.

Podemos imaxinar un futuro no que a confianza non dependa de intermediarios que concentran o poder, senón que estea codificada na propia estrutura das redes. Un mundo no que a transparencia e a garantía da inmutabilidade da información sexan a norma, e non a excepción. Un escenario onde artistas, emprendedores e visionarios poidan desbloquear novas formas de crear

valor e conectar con audiencias globais.

Blockchain non é só unha tecnoloxía; é unha porta de entrada a un novo paradigma tecnolóxico, un catalizador para a innovación que ten o potencial de democratizar o acceso ás novas oportunidades xurdidas por mor dos avances tecnolóxicos, e así empoderar ás persoas en todo o planeta. Ao embarcarnos nesta viaxe para entender as cadeas de bloques, non só estamos aprendendo sobre unha tecnoloxía; estamos albiscar os cimentos dun futuro máis aberto, xusto e conectado.

Con blockchain á vangarda da innovación tecnolóxica, este manual pretende ofrecer ás lectoras e lectores unha comprensión profunda dos principios, retos e do potencial transformador desta tecnoloxía, utilizando estes momentos revolucionarios como unha lente para explorar o seu papel na sociedade.

Servizo de Publicacións

Universidade de Vigo

