



ethereum

FUNDAMENTOS DE REDES

Memoria de la exposición

*Pedro Bonilla Nadal
Ana Peña Arnedo*

20 de noviembre de 2017



1. Introducción

Hoy en día, la información es todo. Las grandes compañías de internet han alcanzado un modelo de negocio en el cual no tienen que exigirte dinero para conseguir beneficios. Estos te cobran por una aceptación de ceder tu información para traficar con ella.

En la actualidad, con el objetivo de mantener un sistema de comercio que no tenga que ser mantenido por un organismo que, por lo tanto mantenga poder sobre sus usuarios. Este tipo de comercio 'tradicional' incluye organizaciones como, por ejemplo, amazon, que facilita la compra-venta respaldada por una compañía, o los bancos centrales, los cuales mantienen las divisas y son los responsables del cuidado y mantención de la divisa. Como sabemos que una compañía esté al cargo del cuidado del sistema, (es decir, que tenga un sistema centralizado) tiene una serie de beneficios e inconvenientes.

Por un lado el sistema centralizado con control por una compañía, como la contratación de especialistas para almacenar y proteger la seguridad, y reduce los tiempos de almacenamiento y actualización del sistema.

Por otro lado, donde hay una ventaja, hay un inconveniente. Como se ha podido ver en alguna ocasion¹ esta organización habilita a grandes compañías y estados a filtrar, robar o modificar a información sensible de los usuarios. Esta vulnerabilidad ha llegado a ser calificada como el 'pecado original' de internet, pues este siempre intento ser una plataforma de caracter descentralizado.

Satoshi Nakamoto inició en 2008 el desarrollo de bitcoin. Este hecho desencadenaría un desarrollo increíble en el area de las divisas con centralizadas, es decir un sistema monetario que no necesitase de un sistema bancario o de un valor predeterminado. También sorprendió mucho a la comunidad la aplicación de la tecnología de la cadena de bloques, como una herramienta basada en la distribución del consenso. A raíz de estas surgieron otras tecnologías basadas en la cadena de bloques, como namecoin, añadiendo. Lo que ethereum pretende es proveer una cadena de bloques con un lenguaje turing completo integrado que pueda ser usado para el desarrollo de contratos inteligentes, permitiendo a los usuarios la creación de proyectos solo escribiendo la logica de estos en unas pocas lineas de código. Algunos de estos proyectos, sacados de la propia web [ethereum.org](https://www.ethereum.org) serían la creación de un crowdfunding que no sea basado en la confianza si no en un contrato que almacenará el dinero de los contribuyentes, una organización autónoma democrática.

En resumen, nuestra moneda lo que intenta es crear un 'ordenador global' que descentralice el actual cliente-servidor. Con Ethereum en lugar de tener un servidor tendríamos un conjunto de nodos, almacenados por voluntarios alrededor del mundo . Sobre este sistema, la comunidad podrá competir por ofrecer servicios, además de consumirlos.

¹<https://www.theguardian.com/world/2013/jun/06/nsa-phone-records-verizon-court-order>

Ejemplifiquemos la diferencia: navegar por una app-store cualquiera nos ofrecerá una serie de aplicaciones en las cuales contactaremos con un servidor que nos proveerá del servicio, generalmente gestionado por un tercero no relacionado (de manera directa) con la transacción.

Ethereum, si todo funciona, devolvería la propiedad de la información a su dueño, de modo que solo el usuario puede modificar la información, y no ninguna entidad externa.

2. Ethereum

2.1. Historia.

En 2012, un joven de 19 años propuso una nueva plataforma, con el objetivo de transformar por entero internet. Vitalik Buterin, un programador de Toronto empezó su investigación en la cadena de bloques en 2011. Co-fundó el portal web Bitcoin Magazine y trabajó para compañías de la materia. En el camino pensó en una plataforma que fuera más allá de las posibilidades del bitcoin.

Con esta idea nació ethereum, una plataforma para la creación de contratos inteligentes (smart contract). Después de publicar en 2014 el white paper, otros desarrolladores se unieron al proyecto.

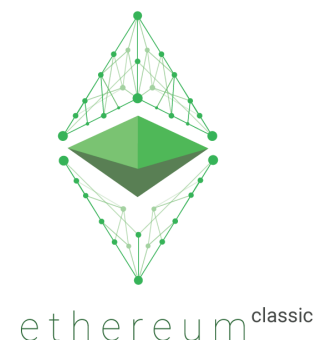
Para lanzar el proyecto se inició un crowdfunding en julio de 2014, donde los participantes compraban ether. Después de reunir más de 18 millones de dolares, se inició y en 2015 se lanzó una plataforma, no demasiado user-friendly, pero con comandos que permitía la creación de aplicaciones descentralizadas.

Este nuevo tipo de contratos caló entre el público llamando la atención de gigantes tecnológicos como IBM y gran cantidad de desarrolladores. El dinero recaudado inicialmente está gestionado por Ethereum foundation², una compañía sin ánimo de lucro ubicada en Suiza.

2.1.1. Hardfork y el cisma de ethereum

En 2016 una organización autónoma descentralizada llamada The DAO, un conjunto de contratos inteligentes reunieron un total de USD \$150 millones en una crowd-sale. Al final DAO explotó cuando en junio USD \$50 millones en Ether fueron reclamados de manera anónima. El suceso inició un debate sobre si se debía hacer un hardfork y como resultado de la disputa, la red se dividió en dos: Ethereum, el objetivo de este trabajo, que continuó la cadena modificada, y Ethereum Classic, que continuó la cadena original. Desde entonces se generó un conflicto entre ambas comunidades.

Figura 1:



²<https://www.ethereum.org/foundation>

2.2. Funcionamiento.

Una vez sabemos lo que es ethereum, profundizamos en el funcionamiento de la plataforma. Al usar ethereum, la ‘app’ no requiere ninguna entidad para almacenar y controlar sus datos. Para conseguirlo, ethereum hace uso del protocolo de bitcoin y su diseño de la cadena de bloques, aunque lo ajusta de manera que puede respaldar aplicaciones además del dinero. Sin embargo, ethereum persigue abstraerse del diseño de bitcoin con el fin de que los desaprobadores puedan crear aplicaciones o acuerdos que incluyan medidas adicionales, nuevas normas de propiedad, formatos alternativos de transacciones o diferentes maneras de transferir estados.

El objetivo del lenguaje de programación ‘Turing-completo’ de ethereum es permitir a los desarrolladores escribir más programas (que los que permite la línea de comandos de bitcoin) en los que las transacciones en la cadena de bloques puedan gobernar y automatizar resultados específicos. Tengamos en cuenta que un lenguaje turing-completo es capaz de generar (en teoría) los mismos programas que son desarrollables c++. Esta flexibilidad que ofrece ethereum es la innovación fundamental de ethereum sobre otras plataformas basadas en protocolo bitcoin.

2.2.1. Bitcoin como sistema de transición de estados

Aquí van la página 5 y 6 del pdf que te pasé (el whitepaper) http://www.the-blockchain.com/docs/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf

2.2.2. Cadena de Bloques de Ethereum

La estructura de la cadena de bloques de ethereum es muy similar a la de bitcoin, dado que se trata de un registro compartido de la historia de transacciones completa. Cada nodo en la red almacena una copia de este historial.

La gran diferencia con ethereum es que sus nodos almacenan también el estado más reciente de cada contrato inteligente, además de todas las transacciones de ether. Para cada aplicación de ethereum, la red tiene que mantener un seguimiento del ‘estado’ o información actual de todas estas aplicaciones, incluyendo el saldo de cada usuario, todo el código del contrato inteligente y dónde se almacena todo. Bitcoin usa salidas de transacción no utilizadas para rastrear quién tiene cuánto bitcoin. Aunque suena complejo, la idea es bastante simple. Cada vez que se hace una transacción de bitcoin, la red ‘rompe’ la cantidad total como si fuera dinero impreso, emitiendo bitcoins de vuelta de una forma que hace que la información manejada se comporte como las monedas o cambio físico.

Para efectuar futuras transacciones, la red de bitcoin tiene que añadir todas las piezas de cambio, que se clasifican en ‘gastadas’ o ‘no gastadas’. Ethereum, por otro lado, utiliza cuentas. Como en fondos de cuentas bancarias, las ‘fichas’ de ether aparecen en una cartera, y pueden ser portados (por así decirlo) a otra cuenta. Los fondos siempre están en algún sitio, pero no tienen lo que podría llamarse una relación continua.

Extender.

2.2.3. Máquina Virtual de Ethereum

Con ethereum, cada vez que se usa un programa, una red de miles de computadores lo procesa. Los contratos escritos en un lenguaje de programación específico de contrato inteligente se com-

pilan en ‘bytecode’, lo que una prestación llamada ‘ethereum virtual machine’ (EVM) puede leer y ejecutar. Todos los nodos ejecutan este contrato usando sus EVMs. Cada vez que un usuario realiza alguna acción, todos los nodos de la red tienen que estar de acuerdo en que ese cambio se ha efectuado. El objetivo aquí es que la red de mineros y nodos tomen la responsabilidad de transferir el cambio de estado a estado, en lugar de cualquier autoridad como PayPal o un banco. Los mineros de bitcoin validan el cambio de propiedad de bitcoins de una persona a otra. La EVM ejecuta un contrato con las reglas que el desarrollador programó inicialmente.

El cálculo real en la EVM se consigue mediante un lenguaje ‘bytecode’ basado en pilas, pero los desarrolladores pueden escribir contratos inteligentes en lenguajes de alto nivel como Solidity o Serpent, más fáciles de leer y escribir para las personas, dado que bytecode es un lenguaje poco legible, pero mmás interpretable a un ordenador, análogo al lenguaje máquina. Los mineros son los que evitan un mal comportamiento. A diferencia de en bitcoin, la cantidad de nodos es escalable, por lo que la ausencia de mineros no será un problema a largo plazo. Por ejemplo, deben asegurarse de que nadie gasta dinero más de una vez, y rechazar contratos inteligentes que no se han pagado. Existen varios miles de nodos de ethereum ahí fuera, y cada uno de ellos está compilando y ejecutando el mismo código.

Podemos pensar que todo esto tiene un coste mucho mayor al de cálculos ordinarios, lo cual es cierto. Por ello la red solo debe usarse para casos de uso particular. El tutorial oficial de desarrollo de ethereum reconoce esta ineficacia, declarando: “.A grandes rasgos, una buena heurística para usar es que no podrás hacer nada en la EVM que no puedas hacer en un teléfono inteligente de 1999”.

Este lenguaje será fundamental para la implementación de los contratos inteligentes o smart contract, pues es lo que permite generar los entes de funcionamiento autónomo para el desarrollo de estos.

2.2.4. mensajes, transacciones y estado de transición de ethereum

Los mensajes en ethereum son parecidos de cierto modo a las transacciones de en otros sistemas de cadena de bloques, pero con algunas características llamativas. Un mensaje puede ser creado tanto por una entidad externa como por un contrato, los mensajes pueden contener datos o, si el mensaje es recibido por un contrato, este tiene la opción de responder. Esto implica que un mensaje en etherum puede toar el aspecto de función.

El termino transacción es usado en ethereum para referirse a un paquete firmado que coniene un mensaje para ser enviado por una cuenta externa. Las transacciones serian el recipiente del mensaje, una firma identificando el remitente, la cantidad de ether que mandar, y además los dos valores importantes de Gasprice y Startgas.

Para prevenir el uso de bucles infinitos, cada transacción requiere marcar cuantos pasos de computación pueden ser realizados. Startgas es el límite, y Gasprice es lacuotaa pagar al minero por iteración. Si la transacción se queda sin ”gas” todos los cambios de estado en la cadena se revierten. Si llega a un halts con algun gas restante, este se le envia al contratante.

Otra cosa importante de Ethreum es que An important consequence of the message mechanism is the ”first class citizen”property of Ethereum - the idea that contracts have equivalent powers to external accounts, including the ability to send message and create other contracts. This allows contracts to simultaneously serve many different roles: for example, one might have a member of a decentralized organization (a contract) be an escrow account (another contract) between an paranoid individual employing custom quantum-proof Lamport signatures (a third contract) and a co-signing entity which itself uses an account with five keys for security (a fourth

contract). The strength of the Ethereum platform is that the decentralized organization and the escrow contract do not need to care about what kind of account each party to the contract is.

2.2.5. Contratos inteligentes.

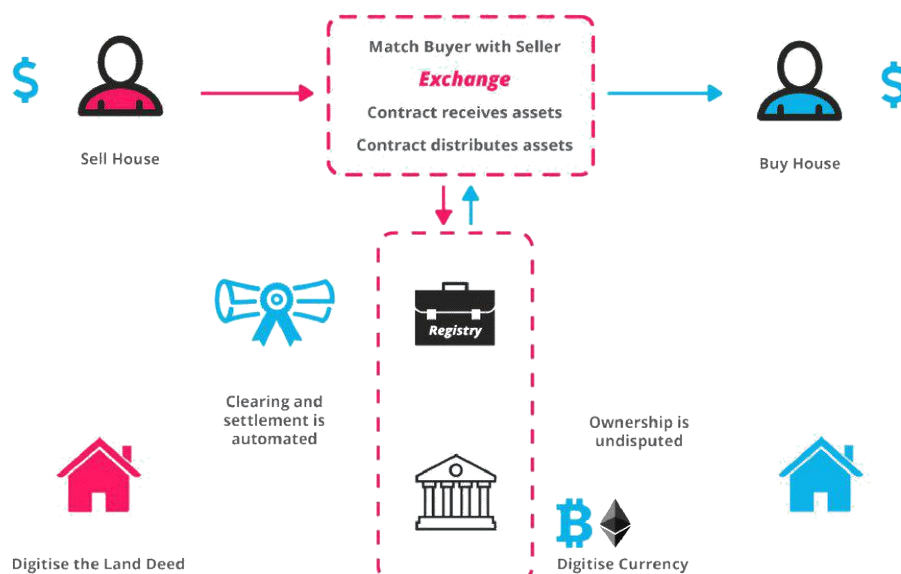
Un contrato inteligente (en inglés Smart contract) es un programa informático que facilita, asegura, hace cumplir y ejecuta acuerdos registrados entre dos o más partes (por ejemplo personas u organizaciones). Como tales ellos les ayudarían en la negociación y definición de tales acuerdos que causarían que ciertas acciones sucedan como resultado de que se cumplan una serie de condiciones específicas.

Un contrato inteligente es un programa que vive en un sistema no controlado por ninguna de las partes, o sus agentes, y que ejecuta un contrato automático el cual funciona como una sentencia if de cualquier otro programa de ordenador. Con la diferencia de que se realiza de una manera que interactúa con activos reales. Cuando se dispara una condición pre-programada, no sujeta a ningún tipo de valoración humana, el contrato inteligente ejecuta la cláusula contractual correspondiente.

Tienen como objetivo brindar una seguridad superior a la ley de contrato tradicional y reducir costos de transacción asociados a la contratación. La transferencia de valor digital mediante un sistema que no requiere confianza abre la puerta a nuevas aplicaciones que pueden hacer uso de los contratos inteligentes, como es el caso de ethereum.

Bitcoin inició este proceso antes, pero su uso está limitado a el uso como divisa únicamente. Sin embargo, el lenguaje turing completo de ethereum que permite a los desarrolladores escribir sus propios programas, soporta que estos desarrollen sus propios contratos inteligentes, es decir, agentes que que controlen todo el proceso del negocio.

How Smart Contracts Works



Estos agentes, no controlados por ninguna de las partes, son los que, (una vez aceptado el contrato) fuerzen a su cumplimiento o devuelvan una vez el contrato se ha terminado de forma fallida al estado original la cadena de bloques.

Un punto importante a considerar como fallo de los contratos inteligentes es que ambas partes han de tener acceso al código del smart contract, y comprender activamente (no porque se lo haya explicado alguien) que hace y porque lo hace el programa brindado. Si estas condiciones se cumplen, y ambas partes saben lo que están aceptando su cumplimiento es obligado, de modo que si no se cumpliera el contrato no tendría ningún efecto.

Los contratos inteligente pueden, entre otras muchas cosas:

- Sevir como un programa de firma de comunidad, en el cual los fondos se invierten solo cuando hay una mayoría prefijada de personas que aceptan la inversión. Del mismo modo este contrato podría ser reformado en el código que lo constituye cuando la misma mayoría (por ejemplo, del 80 %) lo acpeta.
- Controlar acuerdos entre usuarios, por ejemplo, cuando uno contrata un seguro con el otro.
- Gestionar el funcionamiento de otros contratos (recordemos que en ethereum un contrato puede realizar las mismas acciones que un usuario).

Pongamos un ejemplo de este último:

Cuando en una compañía una mayoría acepte bajar la temperatura del AC, otro contrato comproabría la previsión meteorológica para ver si está acción debe tener caracter permanente, y otro contactaría con la base de datos para enviar la información recogida en los anteriores a la base de datos. transaction fees, which depend on the amount of computational power required.

2.2.6. Minería

<https://www.coindesk.com/information/ethereum-mining-works/> <https://www.coindesk.com/information/how-to-mine-ethereum/>

2.2.7. Hardfork

<https://es.cointelegraph.com/news/hard-fork-y-soft-fork-en-qu%C3%A9-consisten-y-cu%C3%A1les-son-sus-diferencias>

3. Aplicaciones.

En general, hay tres tipo de aplicaciones montadas encima de ethereum. La primera categoría es aplicaciones de financiación, las cuales habilitan una vía más amplia de gestionar los contratos. La segunda categoría son aplicaciones semifinancieras, donde el dinero está involucrado pero hay una gran parte de importancia en valores no monetarios. Por último estarían las plataformas para organizaciones de gobieron descentralizado y voto, no monetarias en ningún sentido.

3.1. Sistemas de Token.

Los sistemas de tokens basados en cande de bloques tienen multitud de aplicaciones, desde representar alguna divisa o materia relacionada con el mundo real como el precio del venta de oro o intercambio de USD. Otros usos serían tokens en relación a conceptos como propiedad

intelectual o sistemas no relacionados en ningún punto, no financieros en ningún aspecto.

Los sistemas de token son muy fáciles de implementar con ethereum. El punto clave para entender el funcionamiento es darse cuenta de que cualquier sistema de token es, fundamentalmente, una base de datos con una operación: Quita x unidades a A y dáselas a B , con condición de que:

- A tiene al mmenos X unidades antes de la transacción.
- La transacción está aprobada por A .

Todo lo restante sería implementar esta lógica en un contrato. Un código básico sería:

Algorithm 1 Contrato de Tokens.

```
1: procedure TOKEN( $msg, contract$ )      ▷  $msg$  contiene la información de la comunicación
2:                                     ▷  $contract$  contiene información del contrato
3:    $from = msg.sender$ 
4:    $to = msg.data[0]$ 
5:    $value = msg.data[1]$ 
6:   if  $contract.storage[from] \geq value$  then
7:      $contract.storage[from] = contract.storage[from] - value$ 
8:      $contract.storage[to] = contract.storage[to] + value$ 
9:   end if
10: end procedure
```

Esto sería una implementación en pseudo-código de la lógica básica del sistema bancario ya descrito.

3.2. Sistema de comprobación de identidad y reputación.

INVESTIGAR Y DESARROLLAR

3.3. Decentralized File Storage

En el último tiempo han surgido..

3.4. Decentralized Autonomous Organizations

4. Particularidades.

4.1. Ether.

AQUÍ VA LA PAGINA QUE HAY QUE TRADUCIR.

<https://www.coindesk.com/information/what-is-ether-ethereum-cryptocurrency/>
si pinchas aquí te lleva.

4.2. DoS attack.

Dado el incidente del DAO, hemos de explicar en que consiste este ataque hecho a la red.

https://en.wikipedia.org/wiki/Denial-of-service_attack

Esto se solvento hipotéticamente a finales de 2016 al mejorar la denfensa en este tipo de ataques.
[investigar como lo han hecho](#)

4.3. Escalabilidad

4.4. Minería centralizada



Índice

1. Introducción	1
2. Ethereum	2
2.1. Historia.	2
2.1.1. Hardfork y el cisma de ethereum	2
2.2. Funcionamiento.	3
2.2.1. Bitcoin como sistema de transición de estados	3
2.2.2. Cadena de Bloques de Ethereum	3
2.2.3. Máquina Virtual de Ethereum	3
2.2.4. mensajes, transacciones y estado de transición de ethereum	4
2.2.5. Contratos inteligentes.	5
2.2.6. Minería	7
2.2.7. Hardfork	7
3. Aplicaciones.	7
3.1. Sistemas de Token.	7
3.2. Sistema de comprobación de identidad y reputación.	8
3.3. Decentralized File Storage	8
3.4. Decentralized Autonomous Organizations	8
4. Particularidades.	8
4.1. Ether.	8
4.2. DoS attack.	8
4.3. Escalabilidad	8
4.4. Minería centralizada	8