REVISTA CÁTEDRA

Registro de títulos académicos mediante una aplicación basada en Blockchain y Smart Contracts

Recordkeeping of academic degrees through an application based on Blockchain and Smart Contracts

Luis Rosero-Correa
Universidad Central del Ecuador, Quito, Ecuador
erosero@golden-companies.com
https://orcid.org/0000-0001-7938-768X

Mario Morales-Morales
Universidad Central del Ecuador, Quito, Ecuador
mmoralesm@uce.edu.ec
https://orcid.org/0000-0002-7493-8072

Santiago Morales-Cardoso
Universidad Central del Ecuador, Quito, Ecuador
smorales@uce.edu.ec
http://orcid.org/0000-0002-3833-9654

(Recibido: 29/04/2020; Aceptado: 1/05/2020; Versión final recibida: 15/05/2020)

Cita del artículo: Rosero-Correa, L., Morales-Morales, M. y Morales-Cardoso, S. (2020). Registro de títulos académicos mediante una aplicación basada en Blockchain y Smart Contracts. *Revista Cátedra, 3*(2), 73-98.

Resumen

La implementación de nuevas tecnologías en cualquier tipo de institución surge de la necesidad de generar mejoras en los procesos que éstas realizan con el fin de ofrecer mejores productos y servicios. En este artículo se analiza la propuesta de factibilidad de una

aplicación basada en la tecnología *Blockchain* y en los contratos inteligentes para reproducir el proceso de asignar títulos académicos a estudiantes sin necesidad de un ente central, terceras personas y procesos burocráticos mientras se aprovecha las características de estas tecnologías como la transparencia, la seguridad y la inmutabilidad. Así, se desarrolló dos contratos inteligentes complementarios entre sí aprovechando las características que existen actualmente para crear estructuras que representan objetos de la vida real y funciones que manejen estas estructuras como parámetros. Estos contratos se ejecutaron en un entorno virtualizado el que se simuló una cadena de bloques de *Ethereum* con el conjunto de herramientas de *Truffle*. Se evaluó los contratos inteligentes ingresando datos de prueba y con estos registros almacenados en la cadena de bloques se ejecutó el proceso de asignar los títulos académicos a los estudiantes a través de una función dentro del contrato inteligente principal. Para validar que el proceso se ejecutó correctamente, se realizó consultas a la cadena de bloques y se verificó que los registros de asignaciones de títulos se generaron y almacenaron en la cadena de bloques con éxito. De esta manera se pudo concluir que es factible el modelo propuesto basado en tecnología *blockchain* y contratos inteligentes.

Palabras clave

Aplicaciones descentralizadas, *blockchain,* contratos inteligentes, *Ethereum,* títulos académicos.

Abstract

The implementation of new technologies in any type of institution arises from the need to generate improvements in the processes they execute in order to offer better products and services. This article analyzes the feasibility proposal of an application based on Blockchain technology and smart contracts to execute the process of assigning academic degrees to students without the need for a central entity, third parties and bureaucratic processes while taking advantage of the characteristics of these technologies such as transparency, security and immutability. Thus, two complementary smart contracts were developed, taking advantage of the features that currently exist to create structures that represent real-life objects and functions that handle these structures as parameters. These contracts were executed in a virtualized environment in which an Ethereum blockchain was simulated with the Truffle toolset. Smart contracts were evaluated by entering test data and with these records stored in the blockchain, the process of assigning academic titles to students through a function within the main smart contract was executed. To validate that the process ran successfully, the blockchain was queried, and it was verified that the title assignment records were successfully generated and stored on the blockchain. In this way, it was possible to conclude that the proposed model based on blockchain technology and smart contracts is feasible.

Keywords

Academic degrees, Blockchain, decentralized applications, Ethereum, smart contracts.

1. Introducción

Este estudio resume los componentes más importantes que fueron desarrollados en su totalidad en el trabajo de tesis de Rosero-Correa (2019). La aparición de nuevas tecnologías suele por lo general traer consigo oportunidades de negocio así como también opciones para mejorar los ya existentes, además como se menciona en la revista BBVA *Research* el surgimiento de nuevas tecnologías tiende a generar impacto en la sociedad (BBVA Research, 2016, p. 14). Este impacto puede darse en mayor o menor escala dependiendo de las funcionalidades y la utilidad que dicha tecnología ofrezca y tomando en cuenta estos aspectos se puede entender por qué algunas innovaciones suelen pasar desapercibidas mientras que otras por el contrario tienen mucha acogida como es el caso de *Blockchain* que nació en el 2008 y según *Antonopoulos* (2017) "es una tecnología basada en dos grandes pilares, el primero, los algoritmos criptográficos para cifrar los datos y el segundo la computación distribuida como soporte para el procesamiento de grandes cantidades de información" (p. 22) y gracias a su versatilidad se han generado una gran variedad de aplicaciones como lo indican los ejemplos en Gómez *et al* (2017) en los que se menciona "la logística y transporte , registros de propiedades e internet de las cosas" (p. 6) sin olvidar por supuesto las criptomonedas.

Más tarde, para dar mayor acogida y realce a *Blockchain* aparece el proyecto *Ethereum* el cual desde la perspectiva de Buterin se lo puede entender como un protocolo alternativo que facilita la construcción de aplicaciones descentralizadas (Buterin, 2009, p. 13) gracias al nacimiento de otro concepto conocido como contratos inteligentes que se define como aquellos "programas informáticos inmutables que se ejecutan de manera determinista en el contexto de una Máquina Virtual Ethereum como parte del protocolo de red Ethereum" (Antonopoulos y Wood, 2018, p. 127). Estos programas son capaces de administrar activos que están incluidos en dicha red y una de las características más importantes es que para que todo esto suceda no se requiere de la participación de un mediador ya que como explica Mendoza-Tello et al.(2018) "la verificación de la validez de las transacciones, se distribuye entre todos los nodos que conforman la red de Ethereum, garantizando de esta forma la seguridad y la integridad de éstas ya que se organizan dentro bloques inmutables" (p. 6).

Estas ventajas y características que han permitido a *Blockchain* y a los contratos inteligentes empezar a formar parte de diversas áreas, los han llevado también a incursionar en el ámbito académico como menciona Arenas y Fernandez (2018) al proponer *Blockchain* para usarlo como "un sistema transparente y confiable para asegurar, compartir y verificar las credenciales académicas" (p. 2) con el fin de contar con un repositorio libre e inmutable de documentos que han sido emitidos por una institución académica para que puedan ser consultados por personas interesadas en verificar la validez de dichos documentos.

Tomando estas ideas como premisa, el presente trabajo investigativo consiste en analizar la factibilidad de hacer uso de la tecnología de *Blockchain* sobre la plataforma de *Ethereum* y los contratos inteligentes para aprovechar sus características como la transparencia, la seguridad, la inmutabilidad, la descentralización y herramientas inherentes como la criptografía para a través de todas ellas ejecutar el procedimiento de registro de títulos académicos de tal forma que estos queden almacenados en este registro inmutable que permita verificar su originalidad y validez por parte de personas interesadas en estos aspectos.

La idea de realizar este trabajo surge con la intención de brindar una manera alternativa para realizar el registro de títulos académicos en las instituciones educativas. Esto a propósito de agilizar el proceso de registro, automatizarlo, evitar la dependencia de un ente central y omitir procesos burocráticos que por lo general ocupan mucho tiempo y causan dificultades a estudiantes, profesores y personal administrativo. En consecuencia, los objetivos de este trabajo se plantean como sigue: i) elaborar una propuesta de aplicación basada en la tecnología *blockchain* para el registro de títulos, ii) comprobar la factibilidad de que las herramientas actuales permiten el desarrollo y despliegue de contratos inteligentes, y, iii) seguir una metodología experimental para el desarrollo de contratos inteligentes propuesta por los autores y probar su validez.

Dado que para el desarrollo y ejecución de los contratos inteligentes se requiere de una cadena de bloques de *Ethereum* y como el proceso de crear una cadena de bloques privada resulta demasiado complejo para lo que se pretende probar, se usará como alternativa las herramientas que proporciona *Truffle* que entre otras cosas permiten simular una cadena de bloques de *Ethereum* en la que se van a ejecutar los contratos inteligentes y realizar las respectivas pruebas.

Con estas consideraciones el trabajo se ha estructurado de forma que se presenta en un inicio el estado del arte en el que se habla acerca de diferentes áreas en las que se ha aplicado con éxito la cadena de bloques y los contratos inteligentes, posterior a ello se presenta una sección de preliminares en la que se aborda algunos de los temas más importantes sobre los que se desarrolla el trabajo para generar las bases que permitan entender el contexto general de lo que se desea realizar. La cuarta sección contiene el esquema metodológico experimental que proponen los autores y en que se ha basado esta investigación. Seguidamente, la quinta sección presenta el desarrollo de la propuesta en donde se explica los puntos principales que conformarán la estructura de los contratos inteligentes que se van a desarrollar, así como la ejecución en la cadena de bloques, y, finalmente se presentan las conclusiones obtenidas.

2. Estado de la cuestión

Debido a las grandes ventajas y características que ofrece tanto la cadena de bloques como los contratos inteligentes, en los últimos años han sido varios los escenarios en los cuales han hecho su aparición y siguen extendiéndose cada vez más a nuevos campos en donde la innovación da paso a nuevas aplicaciones; entre las más comunes se pueden destacar las siguientes:

2.1 Gestión de la cadena de suministro

Al hablar de cadena de suministro se debe considerar dos puntos fundamentales, el primero consiste en todo el proceso al que se refiere en sí la cadena de suministro para pasar desde la materia prima hasta los productos elaborados que se venden al por menor; el segundo consiste en garantizar que esos productos estén siempre disponibles para los consumidores y que sean de calidad generando de esta manera confianza en los compradores y prestigio para el producto y la marca.

Las aplicaciones de *Blockchain* en la cadena de suministro tienen muy buena acogida, ya que como se menciona en la revista de Microsoft (2018) gracias a *Blockchain* "las organizaciones rastrean los productos desde la franja de tierra donde crecen hasta la entrega al por menor"

(p. 5). El poder generar registros de todo cuanto sucede en el trayecto de los productos hasta el consumidor final incrementaría la confianza y aceptación porque como explica Galvez de esta manera se estaría otorgando la capacidad de que los consumidores puedan acceder a la historia completa de los productos que adquieren (Galvez 2018, p. 230). Aunque si bien la completa aceptación de *Blockchain* como una herramienta para el mejoramiento de la seguridad y las prácticas de la cadena de suministro puede tomar algún tiempo, este beneficio está siendo respaldado por varias historias de éxito como es el caso de Skuchain que en conjunto con la japonesa NTT Data ha podido construir una plataforma basada en Blockchain para la cadena de suministro y gestión logística (Bermingham, 2018).

2.2 Internet de las cosas (IoT)

Un término que últimamente ha estado causando revuelo es el internet de las cosas. Si bien el internet de las cosas no es en sí una nueva tecnología ha generado un gran impacto en la sociedad ya que debido a su utilidad se lo ha aplicado en una inmensa variedad de campos. Actualmente se sigue investigando de qué manera ampliar sus horizontes y expandirse más, pues como menciona Reyna *et al* (2018) "el internet de las cosas busca un mundo totalmente conectado, donde las cosas pueden intercambiar datos e interactuar entre sí de modo que se pueda representar el mundo real de manera digital" (p. 173). De momento el tener un mundo interconectado generando e intercambiando inmensas cantidades de datos está solventado, pero a esta compartición de datos entre dispositivos heterogéneos le hace falta ciertos aspectos como por ejemplo un alto nivel de seguridad. En este punto la cadena de bloques juega un papel importante como lo explica Hammi *et al* (2018) ya que "proporciona al internet de las cosas aspectos como la integridad, la disponibilidad, escalabilidad, el no repudio, así como también la identificación y la autenticación mutua" (p. 130), y lo corrobora Makhdoom *et al* (2018) mencionando que "Blockchain con su arquitectura descentralizada y sus beneficios clave proporciona una solución ideal para sistemas internet de las cosas especialmente en un entorno no confiable" (p. 260).

El enorme potencial que presenta el internet de las cosas combinado adecuadamente con la cadena de bloques propone la formación de sistemas robustos y confiables en los que se puede tener un registro de todo cuanto sucede en los entornos que se está controlando gracias al apoyo de los dispositivos de internet de las cosas. En este sentido (Christidis y Devetsikiotis, 2016) nos da la pauta de "utilizar estos sistemas robustos dentro de las fábricas, de tal forma que se automaticen procesos y se disminuya la interacción del usuario" (p. 7) teniendo a la vez una base de datos compartida con la cual se pueda realizar un seguimiento de los procesos gracias a las actualizaciones provenientes de los dispositivos de internet de las cosas que se propagan a lo largo de toda la red automáticamente. Un claro ejemplo de la combinación de la cadena de bloques con el internet de las cosas puede ser el caso antes mencionado de Skuchain que como explica Bermingham (2018) "busca controlar la cadena de suministro y la gestión logística mediante la conjunción de la cadena de bloques e internet de las cosas basado en radiofrecuencia (RDIF)" (p. 2).

2.3 Sistemas distribuidos de energía

En los tiempos actuales por el avance a pasos agigantados de la tecnología y el fácil acceso a la información se ha logrado que para las personas sea un poco más sencilla la investigación y creación e sus propios productos, tal es el caso que las personas han comenzado a crear sus propias fuentes alternativas de energía, en parte para tener un sustento ante una falla eléctrica general y en parte para ser más autónomos y no depender

tanto de los servicios públicos, reduciendo así su consumo y los montos que se pagan por el servicio. Pero esta generación autónoma de energía ha resultado tan productiva que algunos de quienes lo hacen incluso han llegado a generar excedentes por lo que aparte de desvincularse del servicio público de energía han encontrado un nuevo modelo de negocio dando paso a los sistemas distribuidos de energía que según Kumar hacen referencia a la generación de manera descentralizada permitiendo de esta manera mejorar la eficiencia general de los sistemas en lo que se refiere a la generación de energía, economía y medio ambiente (Kumar, 2018, p. 5).

Dentro de toda esta idea de generar energía eléctrica de manera autónoma y descentralizada hace su aparición Blockchain que como indica Andoni *et al* (2019) "debido a su naturaleza, las cadenas de bloques podrían proporcionar una solución prometedora para controlar y gestionar sistemas de energía complejos y microrredes cada vez más descentralizados" (p. 151). Teniendo como premisa que estos excedentes de energía se pueden comercializar mediante plataformas sobre una base entre pares que es lo que proporciona la cadena de bloques en la que no intervienen terceros ni intermediarios, solo restaría llegar a un acuerdo sobre cómo se realizará esta comercialización y esto como propone Mylrea y Gourisetti (2017) se puede controlar mediante contratos inteligentes ya que "facilitan los intercambios de energía entre pares al permitir que los productores y consumidores se vendan esta energía entre sí, en lugar de realizar transacciones a través de un sistema de múltiples niveles" (p. 17).

Como ejemplo de la aplicación de la cadena de bloques en sistemas distribuidos de energía se puede mencionar a *Power Ledger* que es una plataforma desarrollada para gestionar el intercambio de energía entre pares y que se ejecuta sobre la tecnología *Blockchain.*

3. Preliminares

3.1 Cadena de bloques

Como primera idea Galvez *et al.* (2018) entre otras cosas menciona que "la cadena de bloques se trata esencialmente de una base de datos distribuida que almacena registros en forma de bloques encriptados que pueden ser verificados en cualquier momento del futuro" (p. 222) y es por esta razón por la que esta tecnología adopta el nombre de cadena de bloques o *Blockchain* puesto que un conjunto de datos se reúnen de manera encriptada en una estructura que se denomina bloque el que a su vez está relacionado con los bloques predecesores a manera de cadena. Otra forma de ver la cadena de bloques que resulta más sencilla e intuitiva es la que nos presenta Crosby *et al* (2015) que nos dice que "a la cadena de bloques se la puede mirar como un libro mayor público de todas las transacciones o eventos digitales que se han ejecutado y compartido entre las partes participantes dentro de una red de Blockchain" (p. 3).

Para entender mejor lo que es la cadena de bloques es importante saber que los bloques son la unidad fundamental de esta cadena y se componen de un conjunto de transacciones que como explica Singh y Kim (2018) "fueron realizadas en un periodo determinado de tiempo" (p. 220). Pero estos bloques por sí solos no representan mucho sino que requieren de un nexo que los una en lo que se denomina la cadena y esto se logra de acuerdo con lo expuesto por Makhdoom *et al* (2018) quien dice que "los bloques se conforman de tal manera que cada bloque nuevo está criptográficamente conectado al bloque anterior" (p. 255) logrando

de esta manera el enlace simbólico que hace que estos bloques conformen un solo grupo secuencial inseparable a manera de cadena.

Dentro de los bloques al ser observados de una manera un tanto más técnica como lo hace Grewal-Carr y Marshall (2016) se pueden observar dos partes importantes que lo conforman (p. 5):

> El encabezado, que incluye metadatos como, un número de referencia de bloque único, la hora en que se creó el bloque y un enlace al bloque anterior.
>
> El contenido, generalmente una lista validada de activos digitales y declaraciones de instrucciones, como las transacciones realizadas, sus montos y las direcciones de las partes en esas transacciones (p. 5).

Una vez definidos estos aspectos sobre cómo se conforma la cadena de bloques se pueden vislumbrar dos perspectivas para ésta, la primera resulta clara de ver puesto que cada bloque tiene información y ocupa espacio en disco, por lo tanto, a medida que se vayan aumentando bloques a la cadena el almacenamiento en disco también irá aumentando, la segunda y que es un poco más complicada de entender aunque presenta una perspectiva más alentadora es la que indica Singh y Kim (2018) y expresa que "mientras más datos tenga la cadena de bloques, ésta se vuelve más fuerte" (p. 220).

3.1.1 Arquitectura de la cadena de bloques

La cadena de bloques rompe el paradigma de un servidor central que gobierna la red proporcionando con eso la característica de la descentralización, por esta razón, Min (2018) explica que:

> Dado que la arquitectura sobre la que se basa una cadena de bloques es una red de malla descentralizada de computadoras, conectadas entre sí en lugar de un solo servidor central, hay una serie de capas que gobiernan las operaciones de la cadena de bloques y crean los protocolos para las aplicaciones de la tecnología Blockchain (p. 3).

Dichas capas se las puede entender como módulos de la arquitectura que, según Min conforman la cadena de bloques (Min, 2018, p. 3). El primer módulo corresponde al módulo de fuente de datos que Min describe como la base para crear una cadena de bloques en la que la base de datos es distribuida, puesto que, no se basa en una arquitectura cliente-servidor, además, no requiere que los usuarios se identifiquen para validar credenciales que pueden ser manipuladas o alteradas. En una segunda capa se encuentra el módulo de transacción, el cual es encargado de validar y crear nuevas transacciones, creando en primer lugar un acuerdo entre las dos partes involucradas y luego enviando la transacción a la red para que ésta sea validada por los mineros. La tercera capa corresponde al módulo de creación de bloques, que es el encargado de agregar a la cadena el nuevo bloque que ha sido minado de modo que cada nuevo bloque se ubique seguido del anterior y esté enlazado a éste. En la cuarta capa se encuentra el módulo de consenso que se encarga de verificar que las transacciones sean válidas mediante un algoritmo de consenso evitando así que se manipulen o corrompan los datos. Finalmente se tiene el módulo de conexión e interfaz, este módulo se encarga de proporcionar interfaces web entre los usuarios a la vez que permite

conocer el estado en tiempo real de la cadena de bloques y las transacciones (Min, 2018, pp. 3–5).

3.1.2 Árboles de *Merkle*

Como ya se mencionó antes, los bloques constan básicamente de dos partes, el encabezado y el cuerpo el cual contiene las transacciones, pero para que éstas no puedan ser alteradas y estén seguras dentro del bloque se las encripta de manera recursiva a través de una estructura de datos de varios niveles que se conoce como árbol de *Merkle*. Las transacciones se ensamblan en dichos bloques de tal manera que cada bloque consecuente se conecte al bloque anterior a través de un valor *hash*.

Un árbol *Merkle*, también conocido como un árbol *hash* binario, como define *Antonopoulos* (2017) "es una estructura de datos utilizada para resumir y verificar de manera eficiente la integridad de grandes conjuntos de datos" (p. 284). La estructura de este árbol como se muestra en la Figura 1 y como explica Buterin (2009):

> Está compuesto por un conjunto de nodos con un gran número de nodos hoja en la parte inferior del árbol que contiene los datos subyacentes, un conjunto de nodos intermedios donde cada nodo es el *hash* de sus dos hijos, y, finalmente, un solo nodo raíz, también formado a partir del *hash* de sus dos hijos, que representa la 'parte superior' del árbol" (p. 9).

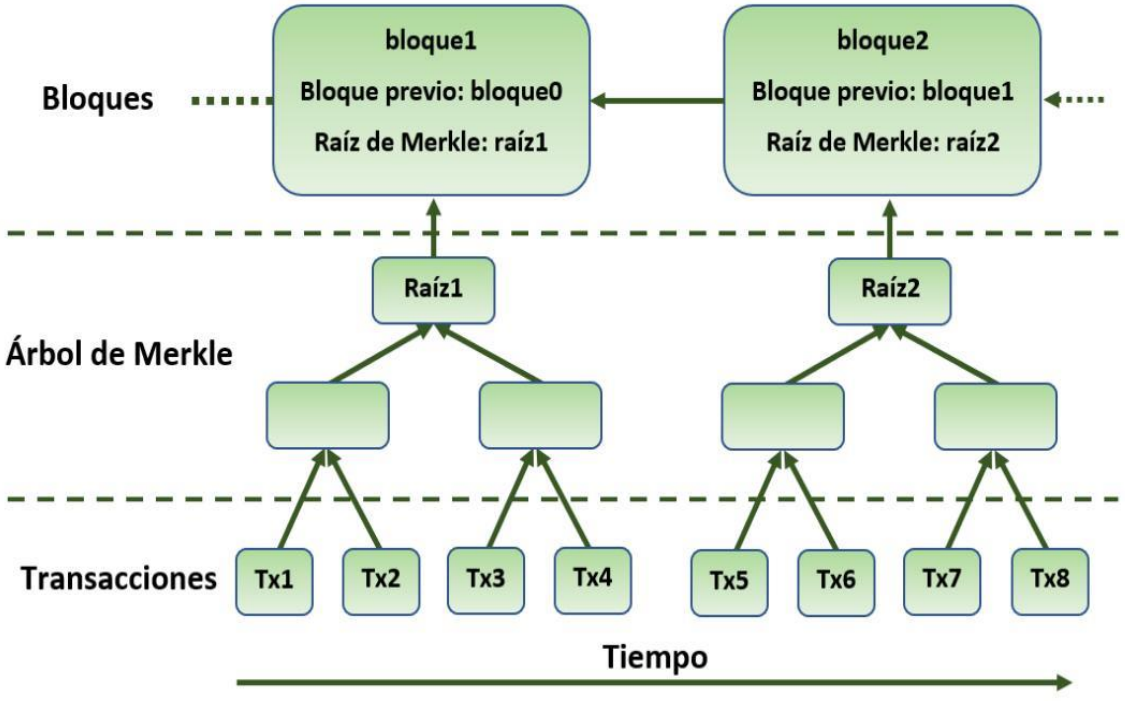

Figura 1. Diagrama de una cadena de bloques, formada a partir de transacciones en una estructura de árboles de Merkle (Antonopoulos, 2017, p. 429)

Para formar la raíz de *Merkle*, se inicia desde el conjunto de transacciones, se obtiene el valor *hash* de cada una y se las va agrupando de a dos, de modo que en el siguiente paso se

calcula el *hash* de la pareja y se repite el proceso hasta que queda un solo nodo. En el caso de que el conjunto de transacciones resulte en un número impar, la estrategia que se toma es duplicar el valor hash de una de las transacciones para poder seguir el proceso recursivo y conseguir la raíz de *Merkle.*

3.1.3 Funciones *Hash*

Las funciones criptográficas *hash* son un elemento importante de la cadena de bloques ya que son usadas al momento de la construcción de los bloques a través de la estructura de los árboles de *Merkle*. Una función *hash* es básicamente una función que transforma cualquier mensaje de una longitud variable a un conjunto de caracteres de longitud fija, independientemente de la longitud que tengan los datos de entrada.

Las funciones hash se usan en muchos algoritmos y protocolos criptográficos de los cuales existen una gran variedad de aplicaciones en el área de la seguridad de la información y en la actualidad las funciones *hash* son de trascendental importancia en aplicaciones donde se requiere eficiencia para implementar verificación de integridad y autenticación como es el caso de aplicaciones basadas en *Blockchain.* Entre los algoritmos más comunes en los que se usan las funciones *hash* Medina (2016) menciona que "están entre otros el SHA-256, que en realidad es procedente de SHA-1, RIPEMD, BLAKE, Skein" (p. 5).

Las funciones *hash* son también ampliamente utilizadas en la criptografía y como ejemplo se puede tomar el de Álvarez *et al* que en su trabajo explica cómo se puede utilizar el Estándar de Cifrado Avanzado o AES por sus siglas en inglés, como un generador de números pseudo aleatorios para servir como base a funciones de *hash* de contraseñas las cuales resultan de mucha utilidad para cifrar contraseñas de usuario, que por lo general son de longitud variable y no se pueden utilizar directamente como claves de cifrado de tamaño fijo (Álvarez et al., 2018, p. 1).

3.1.4 Protocolos de consenso

Una de las características de *Blockchain* es que dentro de toda la red no existe un nodo central encargado de orquestar y gestionar a los demás nodos la información que cada uno de ellos almacena en su copia de la cadena de bloques, o visto de otra manera, en la cadena de bloques no hay un nodo central que asegure que los libros mayores en todos los nodos distribuidos sean todos iguales. Además como explica Zheng *et al.* (2017) "los nodos no necesitan confiar en otros nodos por lo que algunos protocolos son necesarios para garantizar que los registros en diferentes nodos sean consistentes" (p. 358). La idea es que en una red distribuida en donde los participantes son desconocidos y poco confiables, las transacciones pueden ser verificadas a través del consenso, siendo este consenso el mecanismo o conjunto de reglas que permite a todos los nodos llegar a un acuerdo sobre el orden de las transacciones.

Existen muchos algoritmos de consenso sobre *Blockchain* en el sector de la energía (Andoni *et al*., 2019, pp. 148–150). Se presentan tres de los que a criterio de los autores son los más importantes:

- Prueba de trabajo o *Proof of Work* (PoW): La prueba de trabajo es un algoritmo que consiste en resolver una tarea computacionalmente intensiva y compleja para poder de esta manera agregar un bloque a la cadena. De forma más específica mediante este algoritmo se debe calcular un valor *hash* para el encabezado del bloque de tal manera que como explica Makhdoom *et al* (2018) "este *hash* criptográfico

computado debe tener un número específico de ceros al inicio según lo que se haya definido en el nivel de dificultad" (p. 258). Cuando un nodo obtiene el valor objetivo transmite el bloque a los demás nodos en la red y éstos deben confirmar mutuamente la exactitud del valor *hash*, de modo que si se valida el bloque los demás nodos deben agregarlo a su copia local de *Blockchain* y como recompensa por el trabajo computacional realizado para el cálculo del valor *hash* el nodo que resolvió la tarea es recompensado.

- Prueba de participación o *Proof of Stake* (PoS): Este mecanismo de consenso en lugar de que sea el nodo quien declare el resultado, es el sistema quien elige a un nodo de la red para que lo calcule mediante lo que Singh y Kim (2018) denomina "un sistema de lotería" (p. 220) en la que se toman en cuenta a los nodos que cuentan con más "capital" en criptomonedas; así mientras más monedas posee un nodo, tiene más probabilidades de ser elegido para calcular el valor *hash* del siguiente nodo que se va a agregar a la cadena. Este mecanismo trae consigo dos ventajas, la primera es que mejora la latencia, las grandes cantidades de cálculo y los altos consumos de energía que son propios del mecanismo de consenso de prueba de trabajo; la segunda se refiere a que existe una menor posibilidad de que la cadena de bloques sufra un ataque porque se requeriría al menos el 51% de capacidad de procesamiento para poder lograr un ataque exitoso. Esta idea se basa según menciona Zheng *et al* (2017) en que "se cree que las personas que poseen más monedas tienen menos interés en atacar la red" (p. 560).
- Algoritmo práctico de tolerancia a fallos bizantinos (PBFT): Este mecanismo está diseñado para resolver conflictos entre nodos informáticos participantes de una red distribuida cuando de entre un conjunto de nodos, alguno genera una salida diferente a la de los demás. Este algoritmo como menciona Andoni *et al* (2019) "requiere que al menos 2/3 de la red se comporte con honestidad y la sobrecarga de mensajes puede aumentar significativamente a medida que aumenta el tamaño de la red, lo que afecta tanto la velocidad como la escalabilidad" (p. 150), lo que se traduce en que este mecanismo puede tolerar hasta un 33% de nodos maliciosos para seguir siendo consistente y seguro. A pesar de ser más eficiente que otros protocolos se lo considera costoso por la cantidad de mensajes que son necesarios para lograr el consenso ya que como explica Liang *et al* (2012) "se requiere que todos y cada uno de los nodos envíen sus resultados utilizando su propio estado interno e información de la que dispone" (p. 4). Todo el proceso para que se pueda lograr el consenso por medio de estos mensajes según Zheng *et al* (2017) "podría dividirse en tres fases: preparación previa, preparación y compromiso" (p. 560).

3.2 Contratos inteligentes

En los últimos años se ha aprovechado la capacidad de *Blockchain* para ejecutar scripts autónomos mediante los cuales los desarrolladores han creado nuevas versiones de la cadena de bloques que pueden realizar cálculos arbitrarios distintos de la transferencia de monedas. Es así como nacen los contratos inteligentes o *Smart Contracts*, que como indica Xu *et al* (2016) "se introdujeron como programas autónomos que se ejecutan en toda la red de *Blockchain* y puede expresar disparadores, condiciones y lógica de negocios para permitir transacciones complicadamente programables" (p. 1), es así que estos contratos inteligentes pueden ejecutar cualquier algoritmo codificado en ellos.

Se debe considerar que los contratos inteligentes pueden ser un programa completo almacenado en una plataforma de cadena de bloques y distribuido por todos los nodos de la red. Para que este contrato exista dentro de la red se debe proceder como indica Destefanis *et al,* es decir, almacenar el contrato inteligente en la cadena de bloques mediante una transacción de creación de contrato, al cual se le asigna una dirección para identificarlo que se genera siempre y cuando la transacción de creación se haya ejecutado con éxito (Destefanis *et al*., 2018, p. 21). Una vez que estos contratos se ejecutan sobre la cadena de bloques se encargan de administrar los activos que dicha plataforma incluye mediante transacciones que van más allá de las simples transacciones de compra/venta de divisas, y pueden tener instrucciones más extensas incorporadas en ellas, todo esto sin depender como explica Gürkaynak *et al* (2018) "de una parte intermediaria, como un banco o un organismo gubernamental para transferencias de valor, al tiempo que proporciona a las partes involucradas en la transacción de absoluta confianza en la validez y seguridad de la transacción" (p. 848).

3.2.1 Estructura de los contratos inteligentes

Como se muestra en la Figura 2, un contrato inteligente básicamente se compone de un saldo de cuenta en monedas virtuales, en este caso Ether que es la criptomoneda de Ethereum, un almacenamiento privado y un código ejecutable. Este código al ser almacenado por medio de una transacción como menciona Luu *et al* (2016) "es un 'agente autónomo' almacenado en la cadena de bloques, codificado como parte de la transacción de 'creación' que introduce el contrato en la cadena de bloques" (p. 256), es por esta razón que se lo identifica mediante la asignación de una dirección única de 20 bytes y una vez introducido en la cadena de bloques no puede ser modificado.

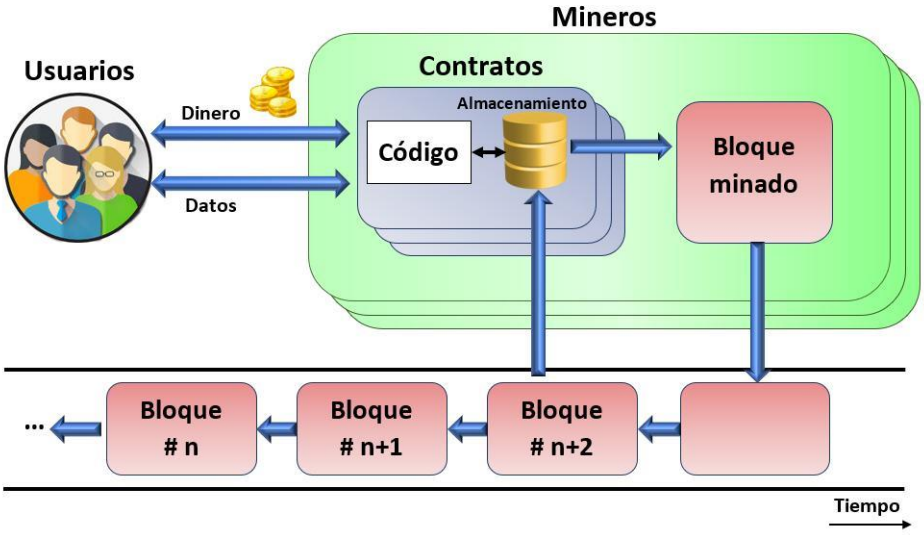

Figura 2. Estructura de los elementos que conforman un contrato inteligente (Alharby y Moorsel, 2017).

El paso adicional que se requiere para que el contrato quede insertado en la cadena de bloques es que la transacción de creación esté dentro del conjunto de transacciones que van a conformar el árbol de Merkle y se agregue en el bloque que va a ser minado e incluido en la cadena de bloques.

3.2.2 Funcionamiento de los contratos inteligentes

De manera general, las transacciones enviadas a un contrato inteligente atraviesan por tres fases, la primera la de las entradas, la segunda correspondiente al intérprete del contrato y la última que son las salidas como se muestra en la Figura 3 y se detallan a continuación.

- **Entradas:** en esta fase se especifican el identificador del contrato, la solicitud de transacción, las dependencias que puedan existir y el estado actual del libro mayor.
- **Intérprete del contrato:** esta fase se carga con el estado actual del libro mayor y el código de contrato inteligente. Para procesar estas transacciones se sigue el procedimiento destacado en Hyperledger (2018) que indica que "cuando el intérprete del contrato recibe una solicitud, la comprueba inmediatamente y luego rechaza cualquier solicitud no válida" (p. 4). El contrato puede, según la transacción que recibe, leer/escribir en su almacenamiento privado, almacenar dinero en el saldo de su cuenta, enviar/recibir mensajes o dinero de usuarios/otros contratos, o incluso crear nuevos contratos.
- **Salidas:** si la solicitud es válida se generan las salidas que incluyen, un nuevo estado y cualquier efecto lateral. Cuando se completa todo el procesamiento, el intérprete empaqueta el nuevo estado, una declaración de corrección y cualquier sugerencia de pedido requerida para los servicios de consenso. Ese paquete se envía al servicio de consenso para el compromiso final con la cadena de bloques.

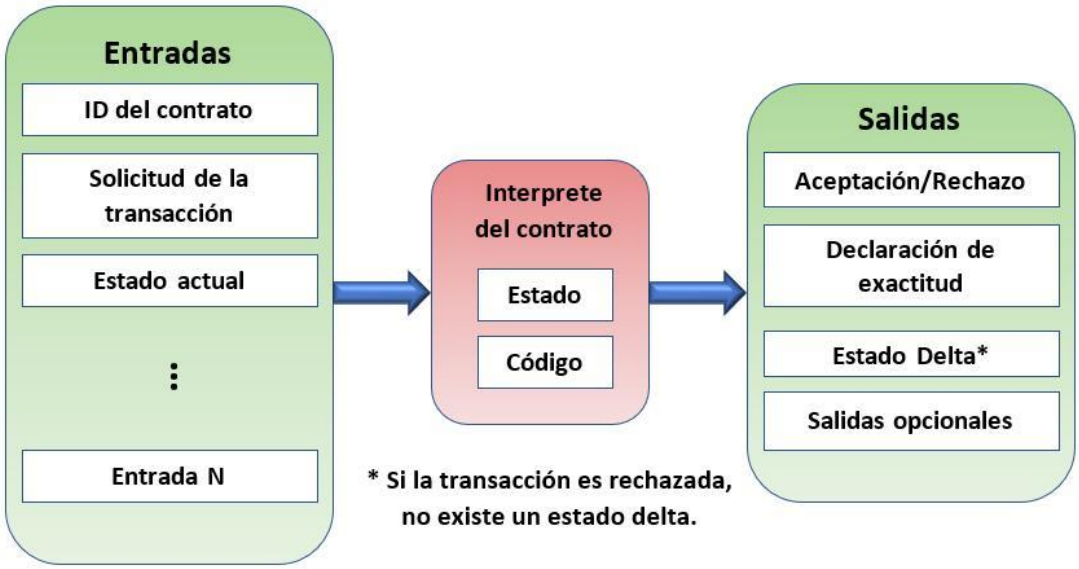

Figura 3. Fases que se deben seguir para la ejecución de un contrato inteligente (Buterin, 2009)

El estado que tiene el contrato al momento de recibir una transacción cambiará a un estado delta en el caso de que la transacción se ejecute de manera correcta, si la solicitud de transacción no se valida, no se ejecutará el código del contrato y por lo tanto no podrá existir un estado delta ya que el estado actual no ha sido alterado.

3.3 Ethereum

Ethereum se concibió en un momento en que las personas reconocían el poder del modelo de *Bitcoin* y trataban de ir más allá de las aplicaciones de criptomonedas. Es así como el

programador Vitalik Buterin que tenía cierto apasionamiento por Bitcoin le dio vida a Ethereum luego de pasar una serie de etapas que se iniciaron en el 2013. Su intención era ampliar las capacidades de Bitcoin y Mastercoin proponiendo en octubre de ese año un enfoque más general en el que concebía contratos más flexibles con los cuales reemplazar el lenguaje especializado de Mastercoin. De este modo para diciembre comenzó a compartir un documento técnico en el que se describía la idea central detrás de *Ethereum* que consistía, como menciona Antonopoulos y Wood (2018) "en una cadena de bloques que sea Turing completa y de propósito general" (p. 41).

Esta plataforma es diferente de los sistemas de *Blockchain* anteriores y a más de ser el protocolo *Blockchain* líder en términos de innovación se la conoce según explica Dika y Nowostawsky (2017) "como la computadora mundial y como el futuro de Internet con tecnología de cadena de bloques" (p. 8). Debido a su novedosa idea de procesamiento computacional distribuido de aplicaciones sin la intervención de terceros y el predominio de la transparencia, se puede mencionar también que es una plataforma de código abierto lo que representa un cambio cultural de algunos de sus predecesores.

El hecho de que *Ethereum* represente una cadena de bloques con un lenguaje de programación Turing completo incorporado, significa que según Vujičić *et al* (2018) "admite todos los tipos de cálculos, incluidos los bucles y la transición de estado, así como otras mejoras sobre la estructura de la cadena de bloques" (p. 4). Dentro de la cadena de bloques de Ethereum se maneja la criptomoneda Ether (ETH) que permite pagar las transacciones financieras y procesar aplicaciones. Dichas aplicaciones como se menciona en coinPY.net (2018) "se pueden programar en siete lenguajes diferentes entre los que están JavaScript, Go, Python y Lisp" (p. 3).

3.3.1 Cuentas *Ethereum*

Dentro de *Ethereum* como expone Buterin existen dos tipos de cuentas: y aquellas que son cuentas de contrato, creadas específicamente para ejecutar el código de los contratos inteligentes que alojan en su almacenamiento interno (Buterin, 2009, p. 13).

Estas cuentas como explica Buterin (2009) se componen de cuatro campos:

- El *nonce*, un contador que se utiliza para garantizar que cada transacción solo pueda procesarse una vez.
- El saldo de ether actual de la cuenta.
- El código de contrato de la cuenta, si es que lo tiene.
- El almacenamiento de la cuenta que está vacío por defecto (p. 13).

Las cuentas que no son de contrato también cuentan con su almacenamiento interno el cual se mantendrá vacío puesto que, está destinado a almacenar el código de los contratos inteligentes, y como las cuentas de propiedad externa no manejan contratos, no tienen nada que guardar en su almacenamiento interno.

3.3.2 Mensajes y transacciones

En Ethereum se manejan los conceptos de mensajes y transacciones y como explica Buterin (2009):

> Los 'mensajes' en Ethereum tienen cierta similitud con las transacciones en Bitcoin, aunque con tres características que los diferencian. La

> primera es que un mensaje Ethereum puede ser creado por una entidad externa o por un contrato, mientras que una transacción de Bitcoin solo puede crearse externamente. La segunda es que existe una opción explícita para que los mensajes de Ethereum contengan datos. Y la última es que el destinatario de un mensaje de Ethereum, si es una cuenta de contrato, tiene la opción de devolver una respuesta; esto significa que los mensajes de Ethereum también abarcan el concepto de funciones (p. 14).

Estas tres características dan a los mensajes de Ethereum una clara ventaja sobre las transacciones de Bitcoin puesto que, a pesar de que dichas características hacen a los mensajes de Ethereum entidades más complejas de gestionar, también permiten ampliar los ámbitos en los que se los puede aplicar gracias a que permiten realizar transporte de datos y ejecutar funciones que utilicen estos datos.

Por otro lado, con respecto a las transacciones Buterin (2009) menciona que:

> El término de 'transacción' se usa en Ethereum para referirse al paquete de datos firmados que almacena un mensaje para enviarlo desde una cuenta de propiedad externa. Las transacciones contienen el destinatario del mensaje, una firma que identifica al remitente, la cantidad de ether y los datos a enviar, así como dos valores llamados STARTGAS y GASPRICE (p. 14).

Según menciona Buterin, STARTGAS hace referencia al límite de la cantidad de pasos que se dan para ejecutar el código solicitado en una transacción de tal forma que, si no se logra completar la transacción en la cantidad de pasos determinada, se la interrumpe y finaliza revirtiendo todos los cambios evitando que la ejecución se realice de manera infinita. En cuanto al GASPRICE, se refiere a la tarifa que se debe pagar al minero por cada paso computacional que realiza para llevar a cabo la ejecución de la transacción (Buterin, 2009, p. 14).

Aquí entra en juego el concepto de gas que como menciona Ast (2018): "el gas es la unidad que mide el trabajo computacional requerido para ejecutar transacciones o contratos inteligentes en la máquina virtual de Ethereum" (p. 2). En otras palabras, si durante la ejecución se termina el "saldo" para ejecutar transacciones, todos los cambios de estado se revertirán, excepto el pago de las tarifas, y si la ejecución de la transacción se detiene con algo de gas restante, la parte restante de las tarifas se reembolsará al remitente.

3.3.3 Contratos inteligentes en *Ethereum*

Un contrato inteligente desde la perspectiva de Buterin se puede utilizar para representar prácticamente cualquier tipo de activo susceptible de ser digitalizado escribiendo la lógica en unas pocas líneas de código dentro del contrato (Buterin, 2009, p. 1). Con esta idea, se procede a escribir el código del contrato en un lenguaje de programación aceptado por la plataforma. Una vez escrito el código, es suficiente con cargarlo, ingresar las variables iniciales y enviarlo para que sea procesado y pueda ejecutarse. Para su ejecución, los contratos requieren en primer lugar de un software especial denominado Máquina Virtual de *Ethereum* (EVM) que a su vez se ejecuta en cada uno de los nodos de la red de *Ethereum*, y en segundo lugar de la transformación del código a *bytecode* que es el lenguaje que entiende la Máquina Virtual de *Ethereum*.

Puesto que *Ethereum* permite a los usuarios cargar y ejecutar códigos que representan los contratos, pudiendo ser éstos sencillos o arbitrariamente complicados, aunque se debe tener en cuenta como menciona Kiffer *et al* (2017) que "cada operación que el código ejecuta, y cada byte de memoria que usa el código, cuesta 'gas'" (p. 95), es decir que mientras más complejos son estos contratos, más es la cantidad de ether que van a gastar para su ejecución. En Ethereum los contratos también cuentan con su propio equilibrio de ether, e incluso pueden transferir ether y llamar a otros contratos; tienen además su propio almacenamiento y tienen la capacidad de actuar como una cuenta de propiedad externa.

4. Metodología experimental

El término metodología se refiere, según Quecedo y Castaño (2002) "al modo en que enfocamos los problemas y buscamos las respuestas, a la manera de realizar la investigación" (p. 7). Esto nos da la pauta de que la metodología que se usa para resolver un problema puede variar dependiendo de cómo se mire el problema y así mismo de cómo se propongan las respuestas.

La metodología ocupa un lugar importante dentro del proceso de investigación, de tal manera que para Rodríguez y Valldeorialo (2014) "la metodología resulta fundamental en cualquier proceso de investigación, ya que determina el modo como dicha investigación se desarrolla" (p. 31). Considerando estos aspectos sobre la metodología, se ha decidido encaminar esta investigación mediante el método descriptivo, el cual, como indica Pérez (2004) "se orienta hacia el presente y los niveles en los que actúa son la investigación aplicada e investigación activa" (p. 91) que es justamente lo que se busca en esta investigación.

Por tanto, para el desarrollo de este artículo, los autores proponen la metodología que se detalla a continuación:

- **Determinar la funcionalidad de los contratos inteligentes:** como primer punto, se requiere establecer con exactitud, qué es lo que van a realizar nuestros contratos inteligentes, de tal manera que podamos determinar los recursos y herramientas necesarios para el desarrollo del trabajo.
- **Descripción de la arquitectura requerida:** el paso siguiente consiste en realizar un análisis detallado de las funcionalidades del contrato para establecer los requerimientos de la arquitectura y el entorno de trabajo necesario para llevar a cabo el desarrollo, con el fin de tener una idea clara que permita preparar dicho entorno de trabajo de manera adecuada.
- **Selección de herramientas:** una vez claros los requerimientos para el desarrollo del trabajo, lo siguiente consiste en seleccionar un conjunto de herramientas de preferencia de software libre que den soporte a la arquitectura y todo el entorno especificado por los requerimientos de la primera fase.
- **Desarrollo de los contratos inteligentes de la aplicación:** con la arquitectura y el entorno de trabajo listos, se puede continuar con la creación de los contratos inteligentes, por lo que esta fase se centra en la creación de archivos y escritura de código en base a los siguientes lineamientos:
    - Manejar una estructura de directorios que permita mantener el orden de los archivos de acuerdo con su funcionalidad

    - Utilizar para los nombres de los archivos palabras que permitan identificar el propósito para el que fueron creados y la funcionalidad que cumplen dentro del proyecto
    - Para el código de los contratos inteligentes, nombrar las variables y las funciones de tal modo que al leerlas se pueda entender de manera clara, cuál es su propósito y la función que cumplen dentro del contrato.
- **Despliegue del contrato inteligente:** el objetivo de esta fase es realizar el procedimiento de despliegue de los contratos inteligentes que consiste en insertar dichos contratos en la cadena de bloques. Al mismo tiempo se realiza el monitoreo de la generación de transacciones y el minado de bloques para verificar que se ha desplegado de manera correcta. Esto ayuda a verificar posibles errores y a comprender el funcionamiento del proceso. Dentro de esta fase se consideran tres tareas importantes que son:
    - La depuración, para verificar que no existan errores o inconsistencias dentro del código escrito con el propósito de evitar que el contrato se ejecute de manera incorrecta o produzca resultados inesperados.
    - La compilación, para convertir el código fuente en binario que es lo que se requiere para ejecutar el contrato inteligente.
    - El despliegue, que consiste en enviar el código compilado del contrato inteligente a la red para que esté disponible para los usuarios y se pueda ejecutar en la máquina virtual de *Ethereum*
- **Interacción con el contrato inteligente:** esta fase tiene como objetivo interactuar con los contratos inteligentes mediante llamadas a las funciones que las componen. De esta manera se realizará el registro y consulta de datos mientras a la par se realiza un monitoreo. En primer lugar, se verifica las salidas que produce la ejecución de las funciones y en segundo lugar la generación de transacciones y el minado de bloques para agregarlos a la cadena de bloques.

5. Desarrollo de la propuesta

5.1 Contratos inteligentes en el registro de títulos académicos

La idea de utilizar la cadena de bloques y los contratos inteligentes en el proceso de registrar títulos académicos consiste en proporcionar una manera confiable y segura para verificar la existencia y autenticidad del título que ha conseguido una persona. De tal forma que, cuando se consulte esta información, se tenga la seguridad de que es veraz y no ha sido alterada al saber que se encuentra almacenada en un registro inmutable como lo es la cadena de bloques.

Al ser los contratos inteligentes autoejecutables e implementar automáticamente los términos del acuerdo entre dos partes, permiten la agilización de procesos proporcionando además como menciona Toyoda *et al* (2017) "la capacidad de identificar falsificaciones si es que se encuentra alguna inconsistencia en el proceso" (p. 2). Esta característica permitirá evitar registros de títulos indebidos y garantizará que los títulos asignados a las personas son reales, además de no poder ser modificados debido a la dificultad que representa hacerlo por la lógica con la que se generan los bloques en la cadena, en la que cada uno está criptográficamente enlazado al anterior.

La principal ventaja que representa el realizar el registro de títulos académicos mediante contratos inteligentes basados en la cadena de bloques es que esto abarca un gran número de características adicionales como la inmutabilidad, la transparencia, la seguridad y la descentralización. Este conjunto de características cambia la perspectiva de la forma en la que se lleva a cabo esta actividad en la actualidad y permite formar parte de una tecnología que augura excelentes perspectivas para el futuro. Actualmente lo que mejor acogida tiene son los sistemas descentralizados y la generación de confianza entre entidades desconocidas mediante el uso de herramientas como la criptografía.

5.2 Desarrollo del contrato inteligente

La Figura 4 muestra el esquema general del proceso a seguir para la elaboración del contrato inteligente que abarca desde la codificación hasta el despliegue en una cadena de bloques privada haciendo uso del conjunto de herramientas que ofrece Truffle.

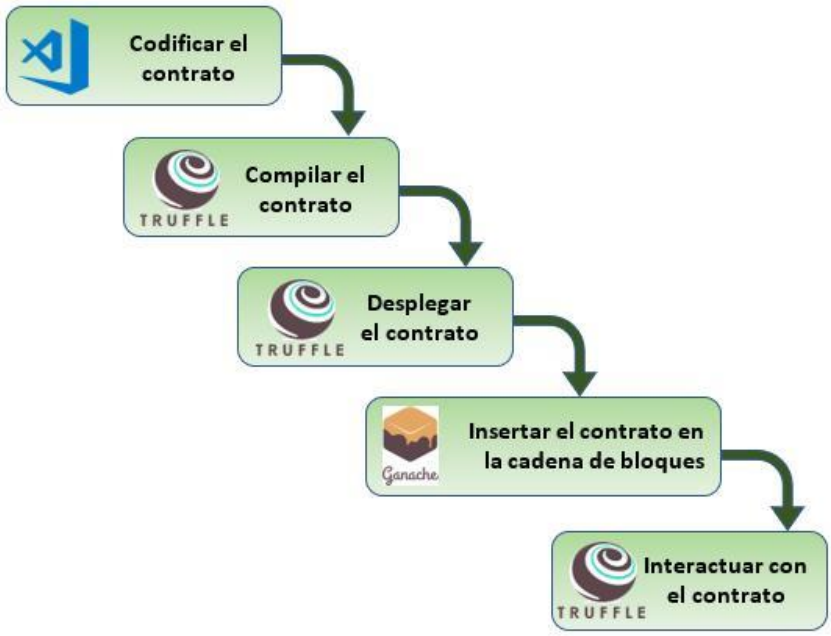

Figura 4. Esquema de las fases para el desarrollo de un contrato inteligente desde la codificación hasta el despliegue

Para el desarrollo se utilizó herramientas de código abierto como es el caso de Visual Studio Code (VSC) el cual se utilizó para escribir el código que, a pesar de ser un producto de Microsoft es libre. Así mismo las aplicaciones del conjunto de herramientas de Truffle son todas de código abierto por lo que pueden ser utilizadas sin la necesidad de licencias.

5.2.1 Definir el contrato inteligente para la gestión de títulos

5.2.1.1 Definir la versión del compilador

Para que se pueda ejecutar el código del contrato inteligente, éste debe ser compilado y como indican Antonopoulos y Wood (2018) "el compilador de línea de comandos para solidity es solc" (p. 134). Este compilador permite convertir el código de solidity en binarios, transformados a su vez en hexadecimales que es lo que entiende y va a ejecutarse en la máquina virtual de Ethereum (Antonopoulos y Wood, 2018, p. 134). Por esta razón, todo contrato inteligente debe iniciarse indicando la versión del compilador que va a utilizar, eso

se hace con el fin de evitar que futuras versiones de compilador puedan introducir cambios incompatibles al momento de realizar la compilación. Al momento de realizar este trabajo la última versión estable del compilador es la v0.5.2 pero se ha optado por utilizar la versión experimental ABIEncoderV2.

5.2.1.2 Declarar el contrato

Para la declaración del contrato se utiliza la palabra reservada *contract* que como indica Antonopoulos y Wood (2018) "es similar a una declaración de clase en otros lenguajes orientados a objetos" (p. 28). A continuación de la palabra *contract* se escribe el nombre del archivo del contrato que por convención se lo hace utilizando la estructura *CamelCase*. Finalmente Antonopoulos y Wood mencionan que se abre y cierra llaves dentro de las cuales se escribirá toda la lógica del contrato, definiendo de esta manera el contrato en sí y el alcance del mismo como ocurre en varios lenguajes de programación (Antonopoulos y Wood, 2018, p. 28).

5.2.1.3 Declarar variables de estado

Un contrato inteligente según Ethereum (2017) "es una colección de código, sus funciones y datos (su estado) que residen en una dirección específica en la Blockchain de Ethereum" (2017, p. 13). Para las variables de estado este caso, se maneja una variable del tipo *address* para asignar la dirección de quién desplegó el contrato. Las tres variables restantes del tipo *mapping* permiten relacionar una dirección con una lista de datos del tipo Titulo, EstudianteTitulado y Estudiante respectivamente, en las que se almacenará registros de dichas estructuras de datos.

5.2.1.4 Declarar el constructor del contrato

El constructor del contrato inteligente es una función que se ejecuta una única vez cuando éste es instanciado y como mencionan Antonopoulos y Wood la manera en la que se escribe el constructor depende de la versión del compilador que se utiliza, así, se puede escribir como una función con el mismo nombre del contrato (para versiones del compilador hasta la 0.4.21) o con la palabra reservada *constructor* (para versiones desde la 0.4.21 y superiores) (Antonopoulos y Wood, 2018, p. 143).

Dentro de la declaración se indica qué acciones se van a realizar para inicializar el contrato, en nuestro caso indicaremos que a la variable de estado propietario se le asignará la dirección de quien creó el contrato.

5.2.1.5 Definir las estructuras de datos

Las estructuras de datos o *structs* son tipos de datos más complejos que se usan para representar objetos de la vida real y se forman mediante la agrupación de varias variables de tipos de datos primitivos (Ethereum, 2017, p. 23). Estas estructuras permiten ampliar las funcionalidades de los contratos inteligentes ya que, se puede manejar tipos de datos personalizados y más complejos, puesto que pueden incluir no solo tipos de datos primitivos sino otros *structs*.

Para la creación del contrato, según el criterio de los autores, se definieron tres estructuras de datos para gestionar cada uno de los objetos que forman parte del proceso de registro de títulos académicos. Dichas estructuras se describen a continuación:

1. La estructura de datos Titulo que contiene tres campos: un identificador de tipo *string*, un nombre de tipo *string* y un entero sin signo que almacenará la marca de tiempo en la que se crea un título.
2. La estructura de datos Estudiante que contiene cuatro campos: la cédula de tipo *string*, el nombre del estudiante, la marca de tiempo de tipo entero sin signo y el *booleano* titulo para indicar si se le ha asignado un título al estudiante.
3. La estructura de datos EstudianteTitulado que contiene cinco campos: el nombre del revisor de tipo *string*, la calificación con la que se registra el título de tipo *uint* o entero sin signo, la marca de tiempo en la que se crea de tipo entero sin signo, el estudiante del tipo Estudiante definido mediante un *struct* y el título de tipo Titulo definido mediante un *struct*.

5.2.1.6 Definir modificadores de funciones

Los modificadores de función de acuerdo con lo que explica Ethereum (2017) "son una forma cómoda de validar las entradas de las funciones" (p. 29). Estos modificadores vienen a constituir una propiedad que permite cambiar el comportamiento que tienen las funciones dentro del contrato. La forma más común de usarlos es para comprobar el cumplimiento de una condición antes de ejecutar la función. Para nuestro contrato se ha creado un modificador que comprueba y obliga a que sea el propietario del contrato, es decir el que lo creó, el único que puede ejecutar la función a la que se aplica este modificador.

5.2.1.7 Funciones definidas en el contrato

Para nuestro contrato se definieron varias funciones clasificadas como primarias y secundarias de acuerdo con la tarea que realizan dentro el contrato, como se resume en la Cuadro 1.

Identificador	Tipo	Descripción
registrarNuevoEstudiante	Principal	Permite ingresar el registro de un nuevo estudiante proporcionando el nombre y la cédula
registrarNuevoTitulo	Principal	Permite ingresar el registro de un nuevo título proporcionando el identificador y el nombre del título
registrarEstudianteTitulado	Principal	Permite registrar un título a un estudiante proporcionando el nombre del revisor, la calificación del título, la cédula del estudiante y el identificador del título.
verificarEstudiantes	Secundaria	Realiza una búsqueda en el registro de estudiantes mediante la cédula del estudiante.
verificarTitulos	Secundaria	Realiza la búsqueda en el registro de títulos mediante el identificador del título.
obtenerEstudiante	Secundaria	Permite extraer un estudiante desde registro mediante su cédula.
obtenerTitulo	Secundaria	Permite extraer un título desde el registro mediante su identificador.
obtenerListaDeEstudiantes	Secundaria	Permite recuperar todos los registros existentes de los estudiantes.

obtenerListaDeTitulos	Secundaria	Permite recuperar todos los registros existentes de los títulos.
obtenerListaDeEstudiantesTitulados	Secundaria	Permite recuperar todos los registros existentes de los estudiantes titulados.

Cuadro 1. Resumen de las funciones del contrato Titulos

5.2.2 Crear el contrato inteligente para el manejo de cadenas

Al igual que en el caso anterior, lo primero a definir en el contrato es la versión del compilador que se va a utilizar, seguido de la definición del contrato y dentro de éste sus respectivas funciones que se resumen en el Cuadro 2.

Identificador	**Tipo**	**Descripción**
compare()	Principal	Permite realizar la comparación de las dos cadenas de texto devolviendo como respuesta un entero que indica si las cadenas transformadas a *bytes* son iguales o no.
equal()	Secundaria	Se encarga de recibir los dos parámetros tipo *string* que se desea comparar para posteriormente enviarlos a su procesamiento en la función *compare()*.

Cuadro 2. Resumen de las funciones del contrato StringUtils

5.2.3 Definir los archivos de migración de los contratos

Para desplegar los contratos inteligentes se requiere construir un *script* de migración el cual va a permitir la implementación del contrato y como indican Antonopoulos & Wood se puede crear un *script* por cada uno, o a su vez un solo *script* que reúna todos los contratos de modo que se los pueda desplegar de manera secuencial (Antonopoulos & Wood, 2018, p. 241).

Para el despliegue de nuestros contratos se usa un solo archivo de migración para los dos contratos generados, Titulos y StringUtils. La estructura del *script* es bastante simple, consta de dos variables de las que cada una hace referencia al archivo del contrato inteligente que se desea desplegar, luego se hace uso del objeto especial de node.js *module.exports* que permitirá exponer los contratos como módulos asignándole el resultado del despliegue de los contratos inteligentes, éste a su vez hace una llamada a la función asíncrona *doDeploy()* que recibe como parámetro el objeto *deployer* que permitirá desplegar los contratos y enlazarlos ya que Titulos importa StringUtils para hacer uso de sus funciones.

5.3 Despliegue del contrato inteligente

Para poder desplegar los contratos inteligentes, se requiere contar con una cadena de bloques de Ethereum, la cual como sugieren Antonopoulos y Wood se la simulará de manera local, generando una instancia privada mediante la herramienta Ganache (Antonopoulos & Wood, 2018, p. 234). La Figura 5 muestra el resultado de la ejecución de la herramienta Ganache la cual generará dicha cadena privada, además de 10 cuentas con sus respectivas direcciones y 100 ethers iniciales que se usarán para procesar las transacciones.

Uno de los aspectos más importantes de esta herramienta es que como explican Antonopoulos y Wood (2018) "ofrece una Interfaz de Programación de Aplicaciones y un conjunto de comandos de Llamada a Procedimiento Remoto codificados como Notación de Objetos Javascript que usualmente se denomina API JSON-RPC" (p. 52). La herramienta Ganache, por defecto proporciona esta interfaz en la dirección *localhost* y el puerto 7 545, que es a donde se debe conectar para interactuar con la cadena de bloques.

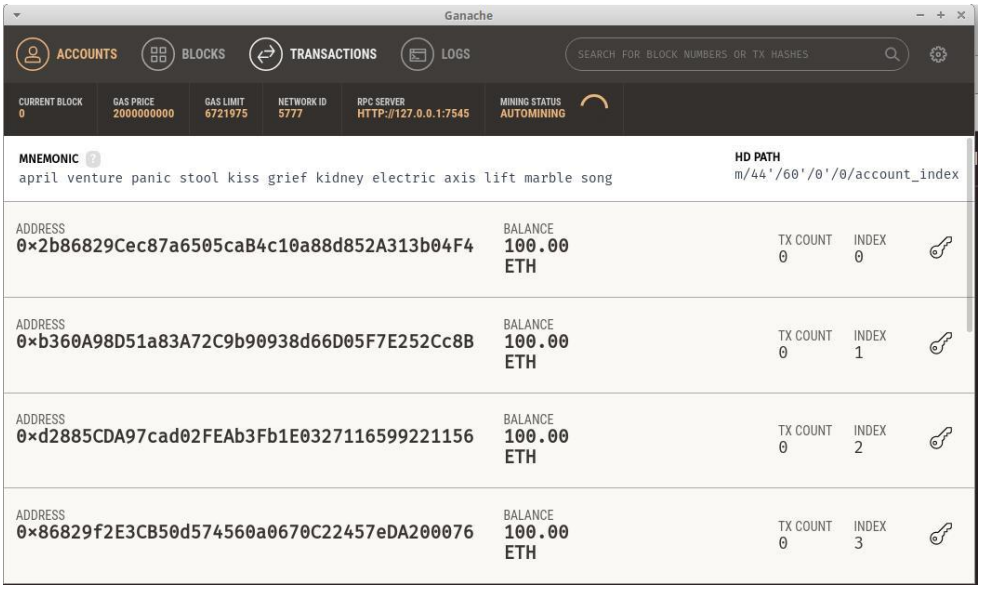

Figura 5. Captura de pantalla de la generación de cuentas en la herramienta Ganache

Con la herramienta *Ganache* ejecutándose y el servidor RPC levantado, se debe implementar el archivo de configuración de *Truffle* llamado truffle-config.js en el que se debe indicar la red de la cadena de bloques a la que se va a conectar, es decir la dirección del servidor de la cadena de bloques proporcionado por *Ganache.*

Con todos los archivos generados tanto de código como de configuración y con las herramientas preparadas, lo que resta es desplegar el contrato inteligente para lo cual se hará uso de la herramienta *Truffle* o más específicamente del comando *truffle migrate* que ejecutará todas las migraciones especificadas en los archivos de migración que por lo general se ubican en el directorio *migrations* y a su vez generarán un nuevo directorio en el proyecto llamado *builds* en el que está el directorio *contracts* que contiene un archivo *json* por cada uno de los contratos en el que existe información correspondiente al despliegue de los mismos.

5.4 Interacción con el contrato inteligente

La Figura 6 muestra el esquema del proceso de interacción con el contrato inteligente una vez que éste ha sido desplegado. La manera más básica a la que se puede acceder para interactuar con el contrato es mediante la consola que proporciona la herramienta *Truffle*, la misma que como explican Antonopoulos y Wood (2018) "es un entorno JavaScript interactivo que proporciona acceso al entorno Truffle y, a través de web3, a la cadena de bloques" (p. 235). Dentro de esta consola, se puede instanciar el contrato inteligente que reside en la cadena de bloques, y a través de esta instancia hacer llamadas a las funciones definidas en dicho contrato.

Para poder ejecutar las funciones de un contrato inteligente, se debe tener una instancia de éste. Para obtenerla, mediante la consola de *Truffle* se asigna a una variable el despliegue del contrato ejecutando el comando Titulos = Titulos.dployed() que generará una salida correspondiente a toda la información del contrato, como el nombre, el código que lo compone, el código compilado y el ABI o Interfaz Binaria de Aplicación que en *Ethereum*, es básicamente la forma en cómo se pueden realizar las llamadas de contrato.

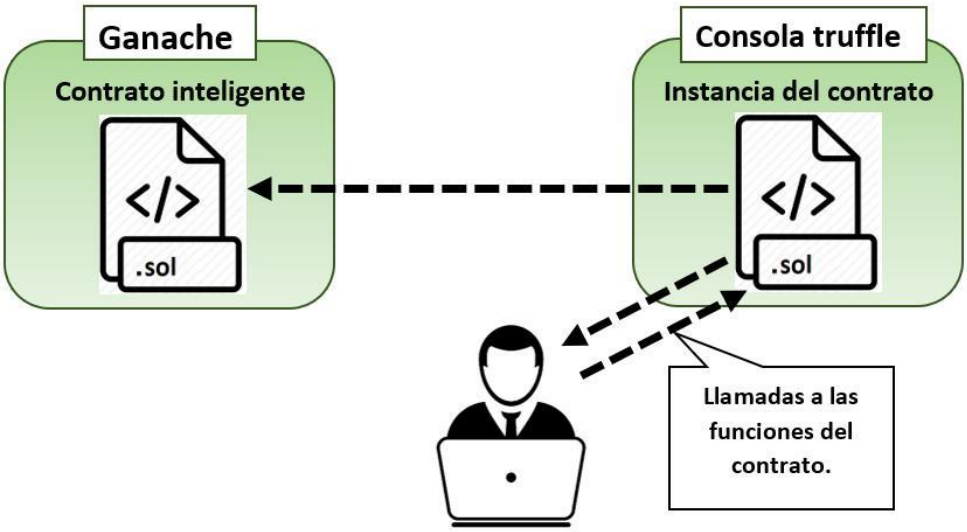

Figura 6. Esquema del proceso de interacción del usuario con el contrato inteligente una vez que este ha sido desplegado.

Una vez obtenida la instancia del contrato en la consola te *Truffle,* los comandos que se ejecuten a través de esta instancia se verán reflejados en la cadena de bloques como transacciones y bloques minados siempre que dichas transacciones se hayan ejecutado de manera exitosa.

6. Conclusiones

A través de esta investigación se ha podido establecer un modelo de aplicación junto con todos los componentes tecnológicos que conforman la arquitectura basada en tecnología *blockchain* para el registro de títulos académicos. Además, se ha podido verificar la factibilidad que *blockchain* presenta para ser usado como base arquitectónica en el desarrollo y despliegue de contratos inteligentes. Para ello se diseñaron dos contratos inteligentes, uno principal destinado a gestionar la creación de registros tanto para títulos y estudiantes, así como para permitir la asignación de los títulos; y un secundario que sirve de apoyo aportando funcionalidades adicionales requeridas por el contrato principal para poder efectuar sus tareas.

Se pudo comprobar que la metodología propuesta en este artículo es válida para el desarrollo de contratos inteligentes. Esto debido a que las fases propuestas en dicha metodología permiten tener un panorama claro del desarrollo de principio a fin. Además, en cada fase se realiza un análisis o se sigue un conjunto de pasos específicos que permiten agilizar los procesos y reducir los errores.

Hablar de *Blockchain* es, sin lugar a duda, hablar de una revolución tecnológica significativa dado que ofrece un potencial extraordinario especialmente en aquellas áreas en donde se requiere un registro confiable e inmutable de cada transacción como por ejemplo el registro de títulos académicos, es por eso que su utilidad trasciende más allá de las criptomonedas y con los instrumentos adecuados como por ejemplo *Ethereum*, se puede explotar todo ese potencial.

Realizar el registro de títulos académicos a través de contratos inteligentes sobre la tecnología *Blockchain* generaría un enorme impacto para la educación superior ya que permitiría mantener un registro altamente confiable de los títulos que posee una persona sin que éste pueda ser modificado a la vez que está disponible para el público en general de modo que puedan acceder a este registro ya sea por cuestiones de información o para validación de datos garantizando que si un título ha sido asignado a un estudiante dentro de este registro en la cadena de bloques, éste es legítimo.

Bibliografía

Alharby, M., y Moorsel, A. Van. (2017). Blockchain-Based Smart Contracts : a Systematic Mapping Study, 125–140. https://doi.org/10.5121/csit.2017.71011

Álvarez, R., Andrade, A., y Zamora, A. (2018). Optimizing a Password Hashing Function with Hardware-Accelerated Symmetric Encryption. *Symmetry*, *10*(12), 705. https://doi.org/10.3390/sym10120705

Andoni, M., Robu, V., Flynn, D., Abram, S., Geach, D., Jenkins, D., … Peacock, A. (2019). Blockchain technology in the energy sector: A systematic review of challenges and opportunities. *Renewable and Sustainable Energy Reviews*, *100*(February 2018), 151. https://doi.org/10.1016/j.rser.2018.10.014

Antonopoulos, A. M. (2017). *Mastering Bitcoin [Book]*. (T. McGovern, Ed.) (Second Edi). O'Reilly Media. Retrieved from https://www.oreilly.com/library/view/mastering-bitcoin/9781491902639/

Antonopoulos, A. M., y Wood, G. (2018). *Mastering Ethereum* (First). USA: O'Reilly Media, Inc.

Arenas, R., y Fernandez, P. (2018). CredenceLedger: A Permissioned Blockchain for Verifiable Academic Credentials. *2018 IEEE International Conference on Engineering, Technology and Innovation, ICE/ITMC 2018 - Proceedings*, 2. https://doi.org/10.1109/ICE.2018.8436324

Ast, F. (2018). Entendiendo el Gas en Ethereum. Retrieved April 19, 2020, from https://medium.com/la-disrupción-del-blockchain/entendiendo-el-gas-en-ethereum-e77a6f30090f

BBVA Research. (2016). TECNOLOGÍA BLOCKCHAIN. *BBVA Innovation Center*, 14. Retrieved from https://www.bbva.com/wp-content/uploads/2017/10/ebook-cibbv-tecnologia_blockchain-es.pdf

Bermingham, F. (2018). Skuchain uses blockchain and IoT for new supply chain platform. Retrieved from https://www.gtreview.com/news/fintech/skuchain-uses-blockchain-and-iot-to-launch-supply-chain-platform/

Buterin, V. (2009). A Next Generation Smart Contract y Decentralized Application Platform, (January). https://doi.org/10.5663/aps.v1i1.10138

Christidis, K., y Devetsikiotis, M. (2016). Blockchains and Smart Contracts for the Internet of Things. *IEEE Access*, *4*, 7. https://doi.org/10.1109/ACCESS.2016.2566339

coinPY.net. (2018). Guía básica de ETHEREUM. *CoinPY.Net Crypto Hosting*, 13. Retrieved from https://www.coinpy.net/assets/docs/eth-guide-es.pdf

Crosby, M., Pattanayak, P., Verma, S., y Kalyanaraman, V. (2015). Blockchain Technology Beyond Bitcoin. *Sutardja Center for Entrepreneurship & Technology Technical Report*. https://doi.org/10.1515/9783110488951

Destefanis, G., Marchesi, M., Ortu, M., Tonelli, R., Bracciali, A., y Hierons, R. (2018). Smart contracts vulnerabilities: A call for blockchain software engineering? *2018 IEEE 1st International Workshop on Blockchain Oriented Software Engineering, IWBOSE 2018 - Proceedings*, *2018-Janua*(March), 19–25. https://doi.org/10.1109/IWBOSE.2018.8327567

Dika, A., y Nowostawski, M. (2017). Ethereum Smart Contracts: Security Vulnerabilities and Security Tools, (December). Retrieved from https://brage.bibsys.no/xmlui/bitstream/handle/11250/2479191/18400_FULLTEXT.pdf

Ethereum. (2017). Solidity Documentation. *Ethereum Foundation*, *1*(1). Retrieved from https://ethereum.github.io/solidity/docs/home/

Galvez, J. F., Mejuto, J. C., y Simal-Gandara, J. (2018). Future challenges on the use of blockchain for food traceability analysis. *TrAC - Trends in Analytical Chemistry*, *107*, 222–232. https://doi.org/10.1016/j.trac.2018.08.011

Gómez, S. C., Castro, S., Presidente, G., Malagón, J., Técnico, V., Montoya, G., … Sánchez, A. (2017). Blockchain: mirando más allá del Bitcoin. *Semana Económica*, *1084*, 6.

Grewal-Carr, V., y Marshall, S. (2016). Blockchain Enigma. Paradox. Opportunity. *Deloitte*, 5. Retrieved from https://www2.deloitte.com/content/dam/Deloitte/uk/Documents/Innovation/deloitte-uk-blockchain-full-report.pdf

Gürkaynak, G., Yılmaz, İ., Yeşilaltay, B., y Bengi, B. (2018). Intellectual property law and practice in the blockchain realm. *Computer Law and Security Review*, *34*(4), 847–862. https://doi.org/10.1016/j.clsr.2018.05.027

Hammi, M. T., Hammi, B., Bellot, P., y Serhrouchni, A. (2018). Bubbles of Trust: A decentralized blockchain-based authentication system for IoT. *Computers and Security*, *78*, 126–142. https://doi.org/10.1016/j.cose.2018.06.004

Hyperledger. (2018). Hyperledger Architecture, Volume II (Smart Contracts). *Hyperledger*, *II*. Retrieved from https://www.hyperledger.org/wp-content/uploads/2018/04/Hyperledger_Arch_WG_Paper_2_SmartContracts.pdf

Kiffer, L., Levin, D., y Mislove, A. (2017). Stick a fork in it: Analyzing the Ethereum network partition. *Proceedings of the 16th ACM Workshop on Hot Topics in Networks - HotNets-XVI*, (March), 94–100. https://doi.org/10.1145/3152434.3152449

Kumar, N. M. (2018). Blockchain: Enabling wide range of services in distributed energy system. *Beni-Suef University Journal of Basic and Applied Sciences*, (August), 5. https://doi.org/10.1016/j.bjbas.2018.08.003

Liang, G., Sommer, B., y Vaidya, N. (2012). Experimental performance comparison of byzantine fault-tolerant protocols for data centers. *Proceedings - IEEE INFOCOM*, 4. https://doi.org/10.1109/INFCOM.2012.6195507

Luu, L., Chu, D.-H., Olickel, H., Saxena, P., y Hobor, A. (2016). Making Smart Contracts Smarter. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS'16*, 256. https://doi.org/10.1145/2976749.2978309

Makhdoom, I., Abolhasan, M., Abbas, H., & Ni, W. (2018). Blockchain's adoption in IoT: The challenges, and a way forward. *Journal of Network and Computer Applications*, *125*(March 2018), 251–279. https://doi.org/10.1016/J.JNCA.2018.10.019

Medina, M. F. (2016). Análisis y comparación de monedas criptográficas basadas en la tecnología blockchain, 5. Retrieved from http://openaccess.uoc.edu/webapps/o2/bitstream/10609/56344/9/mmedinareyTFM0616presentación.pdf

Mendoza-Tello, J. C., Mora, H., Pujol-López, F. A., y Lytras, M. D. (2018). Social Commerce as a Driver to Enhance Trust and Intention to Use Cryptocurrencies for Electronic Payments. *IEEE Access*, *6*(September), 6. https://doi.org/10.1109/ACCESS.2018.2869359

Microsoft. (2018). How blockchain will transform the modern supply chain. *Microsoft*, 5. Retrieved from https://azure.microsoft.com/mediahandler/files/resourcefiles/how-blockchain-will-transform-modern-supply-chain/how-blockchain-will-transform-modern-supply-chain.pdf

Min, H. (2018). Blockchain technology for enhancing supply chain resilience. *Business Horizons*, 3. https://doi.org/10.1016/j.bushor.2018.08.012

Mylrea, M., y Gourisetti, S. N. G. (2017). Blockchain for smart grid resilience: Exchanging distributed energy at speed, scale and security. *Proceedings - 2017 Resilience Week, RWS 2017*, 18–23. https://doi.org/10.1109/RWEEK.2017.8088642

Pérez, G. (2004). *Modelos de investigación cualitativa en educación social y animación sociocultural. Aplicaciones prácticas.* (F. Rubio, Ed.) (4ta ed.). Madrid. Retrieved from https://books.google.com.ec/books?id=iiaMN5VQBnwC&printsec=frontcover&hl=es#v=onepage&q&f=false

Quecedo, R., y Castaño, C. (2002). Introducción a la metodología de investigación cualitativa. *Revista de Psicodidáctica*. Retrieved from https://www.redalyc.org/pdf/175/17501402.pdf

Reyna, A., Martín, C., Chen, J., Soler, E., y Díaz, M. (2018). On blockchain and its integration with IoT. Challenges and opportunities. *Future Generation Computer Systems*, *88*, 173–190. https://doi.org/10.1016/j.future.2018.05.046

Rodríguez Gómez, D., y Valldeorialo Roquet, J. (2014). Metodología de la investigación. *Universitat Oberta de Catalunya*, 82. Retrieved from https://www.redalyc.org/pdf/175/17501402.pdf

Rosero Correa, L. E. (2019). Propuesta de una aplicación basada en la tecnología blockchain para el registro de títulos académicos (Bachelor's thesis, Quito: UCE)

Singh, M., y Kim, S. (2018). Branch based blockchain technology in intelligent vehicle. *Computer Networks*, *145*, 219–231. https://doi.org/10.1016/j.comnet.2018.08.016

Toyoda, K., Takis Mathiopoulos, P., Sasase, I., y Ohtsuki, T. (2017). A Novel Blockchain-Based Product Ownership Management System (POMS) for Anti-Counterfeits in the Post Supply Chain. *IEEE Access*, *5*(XXX), 17465–17477. https://doi.org/10.1109/ACCESS.2017.2720760

Vujičić, D., Jagodić, D., y Randić, S. (2018). Blockchain technology, bitcoin, and Ethereum: A brief overview. *2018 17th International Symposium on INFOTEH-JAHORINA, INFOTEH*

2018 - Proceedings, *2018-Janua*(August), 1–6. https://doi.org/10.1109/INFOTEH.2018.8345547

Xu, X., Pautasso, C., Gramoli, V., Ponomarev, A., y Chen, S. (2016). The blockchain as a software connector. Retrieved from http://web.b.ebscohost.com/ehost/detail/detail?vid=0&sid=11a67777-b990-48ef-a0f8-e1668042182a%40sessionmgr103&bdata=Jmxhbmc9ZnImc2l0ZT1laG9zdC1saXZl#AN=20113397930&db=lah

Zheng, Z., Xie, S., Dai, H., Chen, X., y Wang, H. (2017). An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. *Proceedings - 2017 IEEE 6th International Congress on Big Data, BigData Congress 2017*, (October), 557–564. https://doi.org/10.1109/BigDataCongress.2017.85

Autores

LUIS ROSERO-CORREA. Es ingeniero en informática graduado en la Universidad Central del Ecuador.

MARIO MORALES-MORALES. Es ingeniero de sistemas graduado en la Escuela Politécnica Nacional, Ecuador. Sus estudios de maestría en administración de negocios los realizó en la Universidad San Martín de Porres, Perú. Ha obtenido certificaciones en dirección de proyectos (PMI) y analítica de datos, con una extensa experiencia en proyectos empresariales en la región andina.

Actualmente es docente a la Facultad de Ingeniería, Ciencias Física y Matemática de la Universidad Central del Ecuador, y cursa el doctorado en informática en la Universidad de Alicante.

SANTIAGO MORALES-CARDOSO. Es doctor en informática por la Universidad de Alicante, España. Obtuvo el título de ingeniero informático, maestría en ciencias de la ingeniería y maestría en gestión informática empresarial en la Universidad Central del Ecuador.

Actualmente es docente en la Facultad de Ingeniería, Ciencias Físicas y Matemática.