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

(Received on: 29/04/2020; Accepted on: 01/05/2020; Final version received on: 15/05/2020)

Cita del artículo: Rosero-Correa, L., Morales-Morales, M. y Morales-Cardoso, S. (2020). Recordkeeping of academic degrees through an application based on Blockchain and Smart Contracts. *Revista Cátedra, 3*(2), 72-95.

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 B*lockchain* 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. Introduction

This study summarizes the most important components that were fully developed in the thesis work of Rosero-Correa 2019. The emergence of new technologies tends to generate an impact on society (BBVA Research, 2016, p. 14). This impact can be on a greater or lesser

scale depending on the functionalities and usefulness that the technology offers. Taking these aspects into account, it is possible to understand why some innovations tend to go unnoticed while others are very well received, such as the case of Blockchain, which was born in 2008. and according to Antonopoulos (2017) "it is a technology based on two main pillars, the first, cryptographic algorithms to encrypt data and the second, distributed computing to support the processing of large amounts of information" (p. 22) and thanks A great variety of applications have been generated due to its versatility, as indicated by the examples in Gómez et al (2017) in which "logistics and transportation, property records and the internet of things" are mentioned (p. 6) without forgetting of course cryptocurrencies.

Later to give greater acceptance and enhancement to Blockchain appears the Ethereum project which from the perspective of Buterin can be understood as an alternative protocol that facilitates the construction of decentralized applications (Buterin, 2009, p. 13) thanks to the birth of another concept known as intelligent contracts that in the book Matering Ethereum one of the definitions says that intelligent contracts "are immutable software that are executed in a deterministic way in the context of a Virtual Machine Ethereum as part of the network protocol Ethereum" (Antonopoulos & Wood, 2018, p. 127). These programs are capable of managing assets that are included in such network and one of the most important characteristics is that for all this to happen, the participation of a mediator is not required since, as Mendoza-Tello (2018) "explains, the verification of the validity of the transactions is distributed among all the nodes that make up the Ethereum network, thus guaranteeing the security and integrity of these since they are organized within immutable blocks" (p. 6).

These advantages and characteristics that have allowed Blockchain and the intelligent contracts to begin to be part of various areas have also led them to venture into the academic arena as mentioned by Arenas & Fernandez (2018) when he proposed Blockchain to be used as "a transparent and reliable system to secure, share and verify academic credentials" (p. 2) in order to have a free and immutable repository of documents that have been issued by an academic institution so that they can be consulted by people interested in verifying the validity of such documents.

Taking these ideas as a premise, the present research work consists of analyzing the feasibility of making use of Blockchain's technology on the Ethereum's platform and the intelligent contracts to take advantage of its characteristics such as transparency, security, immutability, decentralization and inherent tools such as cryptography in order to execute the procedure of registration of academic titles in such a way that these are stored in this immutable registry that allows the verification of their originality and validity by people interested in these aspects.

The idea of carrying out this work arises with the intention of providing an alternative way to carry out the registration of academic titles in educational institutions. This is in order to speed up the registration process, automate it, avoid dependence on a central entity and omit bureaucratic processes that usually take a lot of time and cause difficulties for students, teachers and administrative staff. Consequently, the objectives of this work are: i) to develop a proposal for an application based on blockchain technology for the registration of titles, ii) to verify the feasibility of the current tools allowing the development and deployment of intelligent contracts iii) to follow an experimental methodology for the development of intelligent contracts proposed by the authors and to test their validity.

Since the development and execution of intelligent contracts requires a chain of Ethereum blocks and since the process of creating a private chain of blocks is too complex for what is intended to be tested, the tools provided by Truffle will be used as an alternative. These tools allow, among other things, the simulation of a chain of Ethereum blocks in which the intelligent contracts are to be executed and the respective tests to be performed.

With these considerations, the work has been structured in such a way that at the beginning the state of the art is presented, showing the different areas in which the block chain and intelligent contracts have been successfully applied. After that, a section of preliminaries is presented, in which some of the most important topics on which the work is developed are dealt with, in order to generate the bases that allow understanding the general context of what is desired to be done. Next, the development of the proposal is presented where the main points that will conform the structure of the intelligent contracts that are going to be developed are explained, as well as the execution in the chain of blocks, and finally the conclusions obtained are presented.

2. State of play

Due to the great advantages and characteristics offered by both the block chain and intelligent contracts, there have been several scenarios in which they have appeared in recent years and continue to extend increasingly to new fields where innovation gives way to new applications; among the most common are the following:

2.1 Supply chain management

Al hablar de cadena de suministro se debe considerar dos puntos fundamentales, el primero consiste en todo el proceso al que se refiere en si 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 rastrear 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 caso de Skuchain que Bermingham en el sitio web Global Trade Review describe como una compañía estadounidense que se ha aliado con la empresa japonesa NTT Data con la finalidad de construir una plataforma basada en blockchain para la cadena de suministro y gestión logística (Bermingham, 2018).

2.2 The Internet of Things (IoT)

One term that's been causing a stir lately is the Internet of Things. Although the Internet of things is not in itself a new technology, it has generated a great impact on society because of its usefulness in a wide variety of fields. Currently, research continues on how to broaden

its horizons and expand it further because, as Reyna (2018) mentions, "the internet of things seeks a totally connected world, where things can exchange data and interact with each other so that the real world can be represented digitally" (p. 173). At the moment having an interconnected world generating and exchanging huge amounts of data is solved, but this sharing of data between heterogeneous devices needs certain aspects such as a high level of security, and it is at this point where the chain of blocks comes into play because as Hammi (2018) explains "the internet of things is provided with aspects such as integrity, availability, scalability, non-repudiation which refers to the ability to ensure that an entity or in this case a device cannot deny having performed a certain action, as well as identification and mutual authentication" (p. 130), and is corroborated by Makhdoom (2018) mentioning that 'Blockchain with its decentralised architecture and key benefits provides an ideal solution for Internet of Things systems especially in an unreliable environment' (p. 260).

The enormous potential of the Internet of Things combined properly with the blockchain proposes the formation of robust and reliable systems in which one can have a record of everything that happens in the environments one is controlling thanks to the support of the Internet of Things devices. In this sense, Christidis (2016) gives us the guideline of "using these robust systems within the factories, in such a way that processes are automated and user interaction is reduced" (p. 7) while having a shared database with which processes can be tracked thanks to the updates coming from the Internet of Things devices that are automatically propagated throughout the network. A clear example of the combination of the block chain with the Internet of Things can be the case mentioned above of Skuchain who as explained by Bermingham (2018) "seeks to control the supply chain and logistics management by combining the block chain and the Internet of Things based on radio frequency (RDIF)" (p. 2).

2.3 Distributed energy systems

In the present times, by the advance to giant steps of the technology and the easy access to the information, it has been obtained that for the people it is a little simpler the investigation and creation of their own products, such it is the case that the people have begun to create their own alternative sources of energy, partly to have a sustenance before a general electrical failure and to be more autonomous and not to depend so much on the public services, reducing their consumption and the amounts that are paid for the service. But this autonomous generation of energy has been so productive that some of those who do so have even generated surpluses, so that apart from disengaging from the public service of energy have found a new business model giving way to distributed systems of energy, which according to Kumar refer to generation in a decentralized manner, thus improving the overall efficiency of systems in terms of power generation, economy and environment (Kumar, 2018, p. 5).

Within this whole idea of generating electricity in an autonomous and decentralized way, Blockchain appears, which as Andoni (2019) indicates "due to their nature, blockchains could provide a promising solution to control and manage complex energy systems and increasingly decentralized microgrids" (p. 151). On the assumption that this surplus energy can be traded through platforms on a peer-to-peer basis, which is what the block chain provides where no third parties or intermediaries are involved, it would only remain to agree on how this trading will take place, and as proposed by Mylrea (2017) this can be controlled through smart contracts, since 'they facilitate exchanges of energy between peers

by allowing producers and consumers to sell this energy to each other, rather than transacting through a multi-tier system' (p. 17).

An example of the application of the block chain in distributed energy systems is the Power Ledger which is a platform developed to manage the exchange of energy between peers and which runs on Blockchain technology.

3. Preliminaries

3.1 Block chain

As a first idea Galvez (2018), among other things, mentions that "the blockchain is essentially a distributed database that stores records in the form of encrypted blocks that can be verified at any time in the future" (p. 222), and it is for this reason that this technology takes the name of blockchain or Blockchain, since a set of data is gathered in an encrypted manner in a structure that is called a block, which in turn is related to the predecessor blocks in the form of a chain. Another way of looking at the blockchain that is simpler and more intuitive is presented by Crosby (2015), who says that "the blockchain can be viewed as a public ledger of all the digital transactions or events that have been executed and shared among the participating parties within a Blockchain network" (p. 3).

To better understand what the blockchain is, it is important to know that the blocks are the fundamental unit of this chain and are composed of a set of transactions, which as Singh (2018) explains, "were performed in a given period of time" (p. 220), but these blocks by themselves do not represent much but require a link that unites them, which is called the chain and this is achieved according to what is stated by Makhdoom (2018) who says that "the blocks are formed in such a way that each new block is cryptographically connected to the previous block" (p. 255), thus achieving the symbolic link that makes these blocks form a single sequential group inseparable as a chain.

Within the blocks, when observed in a somewhat more technical way as Grewal-Carr (2016) does, two important parts can be noted (p. 5):

> The header, which includes metadata such as, a unique block reference number, the time the block was created, and a link to the previous block
>
> The content, usually a validated list of digital assets and statements of instructions, such as the transactions performed, their amounts, and the addresses of the parties to those transactions (p. 5).

Once these aspects of how the block chain is formed have been defined, two perspectives can be glimpsed. The first one is clear to see since each block has information and this takes up disk space, therefore, as more blocks are added to the chain, disk storage will also increase; the second one, which is a little more complicated to understand although it presents a more encouraging perspective, is the one indicated by Singh (2018) and expresses that "the more data the block chain has, the stronger it becomes" (p. 220).

3.1.1Block Chain Architecture

The chain of blocks breaks the paradigm of a central server that governs the network providing with that the characteristic of decentralization, for this reason, Min (2018) explains that:

> Since the architecture on which a block chain is based is a decentralized mesh network of computers, connected together instead of a single central server, there are a number of layers that govern block chain operations and create the protocols for Blockchain technology applications (p. 3).

These layers can be understood as modules of the architecture, and then after an analysis the layers that make up the block chain architecture are detailed (Min, 2018, p. 3).

The first module corresponds to the data source module that Min describes as the basis for creating a block chain in which the database is distributed, since it is not based on a client-server architecture, and it does not require users to identify themselves in order to validate credentials that can be manipulated or altered. In a second layer is the transaction module which is in charge of validating and creating new transactions, first creating an agreement between the two parties involved and then sending the transaction to the network to be validated by the miners. The third layer corresponds to the block creation module, which is in charge of adding to the chain the new block that has been mined so that each new block is located followed by the previous one and linked to it. In the fourth layer is the consensus module that is in charge of verifying that the transactions are valid by means of a consensus algorithm, thus avoiding the manipulation or corruption of the data. Finally there is the connection and interface module, this module is in charge of providing web interfaces between the users at the same time that it allows to know the real time status of the block chain and the transactions (Min, 2018, pp. 3-5).

3.1.2 Merkle trees

As mentioned above, blocks basically consist of two parts, the header and the body which contains the transactions, but in order for these not to be altered and to be secure within the block they are encrypted in a recursive manner through a multi-level data structure known as the Merkle tree. The transactions are assembled in these blocks in such a way that each consequent block is connected to the previous block via a hash value.

A Merkle tree, also known as a binary hash tree, as defined by Antonopoulos (2017) "is a data structure used to efficiently summarize and verify the integrity of large data sets" (p. 284). The structure of this tree as shown in Figure 1 and as explained by Buterin (2009):

> It consists of a set of nodes with a large number of leaf nodes at the bottom of the tree containing the underlying data, a set of intermediate nodes where each node is the hash of its two children, and finally a single root node, also formed from the hash of its two children, representing the 'top' of the tree" (p. 9)..

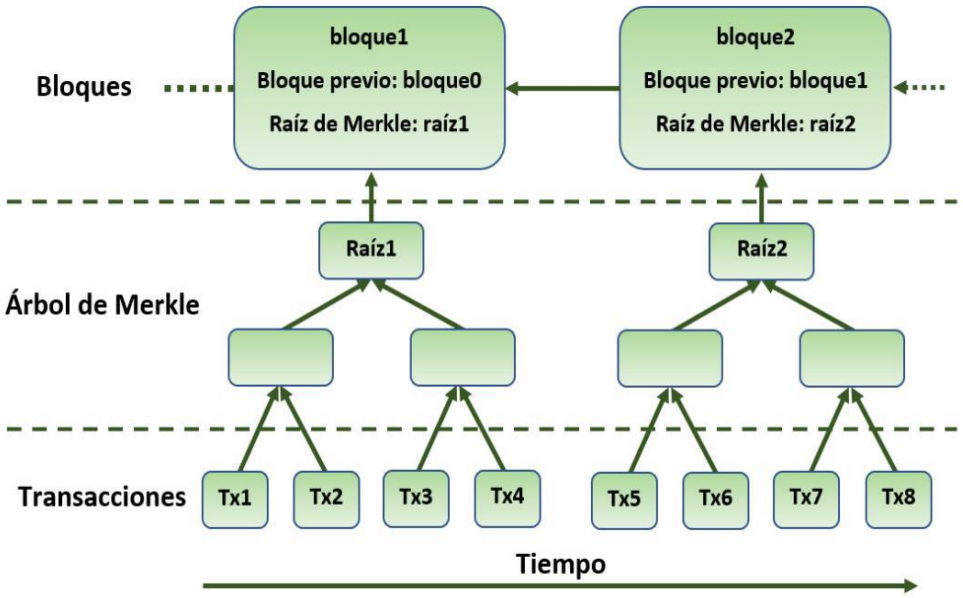

Figure 1. Diagram of a block chain, formed from transactions in a Merkle tree structure (Antonopoulos, 2017, p. 429)

To form the Merkle root, we start from the set of transactions, obtain the hash value of each one and group them in pairs, so that in the next step the hash of the pair is calculated and the process is repeated until only one node is left. If the set of transactions results in an odd number, the strategy is to double the hash value of one of the transactions to follow the recursive process and get the Merkle root.

3,1.3 Hash Functions

The hash cryptographic functions are an important element of the block chain since they are used at the time of building the blocks through the Merkle tree structure. A hash function is basically a function that transforms any message of variable length into a set of characters of fixed length, regardless of the length of the input data.

Hash functions are used in many cryptographic algorithms and protocols of which there are a wide variety of applications in the area of information security and currently hash functions are of paramount importance in applications where efficiency is required to implement integrity verification and authentication as is the case of applications based on Blockchain. Among the most common algorithms in which hash functions are used Medina (2016) mentions that are among others the SHA-256, which actually comes from SHA-1, RIPEMD, BLAKE, Skein (p. 5).

Hash functions are also widely used in cryptography and as an example we can take Alvarez's one that in his work explains how the Advanced Encryption Standard or AES can be used as a pseudo random number generator to serve as a basis for password hash functions which are very useful to encrypt user passwords that are usually of variable length and cannot be used directly as fixed size encryption keys (Alvarez et al., 2018, p. 1).

3.1.4 Consensus protocols

One of the characteristics of Blockchain is that within the whole network there is no central node in charge of orchestrating and managing the other nodes or the information that each

of them stores in its copy of the block chain or otherwise seen in the block chain, there is no central node that ensures that the ledgers in all distributed nodes are all the same, moreover as Zheng et al. (2017) explains "the nodes do not need to rely on other nodes so some protocols are necessary to ensure that the records in different nodes are consistent" (p. 358). The idea is that in a distributed network where the participants are unknown and unreliable, transactions can be verified through consensus, this consensus being the mechanism or set of rules that allows all the nodes to agree on the order of transactions.

Many consensus algorithms exist and Andoni describes many of these in his work on Blockchain in the energy sector (Andoni et al., 2019, pp. 148-150). The three most important ones are presented below:

- Proof of Work (PoW): Proof of work is an algorithm that consists of solving a computationally intensive and complex task in order to add a block to the chain. More specifically, this algorithm must calculate a hash value for the block header, so that as explained by Makhdoom (2018) "this computed cryptographic hash must have a specific number of zeros at the beginning, according to what has been defined in the difficulty level" (p. 258). When a node obtains the target value it transmits the block to the other nodes in the network and these must mutually confirm the accuracy of the hash value, so if the block is validated the other nodes must add it to their local copy of Blockchain and as a reward for the computational work done for the calculation of the hash value the node that solved the task is rewarded.
- Proof of Stake (PoS): This consensus mechanism, instead of the node declaring the result, is the system that chooses a network node to calculate it, through what Singh (2018) calls "a lottery system" (p. 220) in which the nodes that have more "capital" in crypto coins are taken into account, so the more coins a node has, the more likely it is to be chosen to calculate the hash value of the next node to be added to the string. This mechanism has two advantages, the first one is that it improves the latency, the large amounts of calculation and the high energy consumption that are typical of the working test consensus mechanism, the second one refers to the fact that there is less possibility that the block chain suffers an attack because at least 51% of processing capacity would be required to achieve a successful attack. This idea is based, as mentioned by Zheng (2017), on the fact that "people who own more coins are believed to have less interest in attacking the network" (p. 560).
- Practical Byzantine fault-tolerance algorithm (PBFT): This mechanism is designed to resolve conflicts between participating computer nodes in a distributed network, when one set of nodes generates a different output than the others. As mentioned by Andoni (2019), this algorithm "requires at least 2/3 of the network to behave honestly and message overload can increase significantly as the size of the network increases, affecting both speed and scalability" (p. 150), which means that this mechanism can tolerate up to 33% of malicious nodes to remain consistent and secure. Despite being more efficient than other protocols, it is considered expensive because of the number of messages that are necessary to achieve consensus, since as Liang (2012) explains "each and every node is required to send its results using its own internal state and information available to it" (p. 4). According to Zheng (2017), the entire process for achieving consensus through these messages "could be divided into three phases: pre-preparation, preparation and commitment" (p. 560).

3.2 Intelligent contracts

In recent years, Blockchain's ability to run stand-alone scripts has been exploited, whereby developers have created new versions of the blockchain that can perform arbitrary calculations other than the transfer of coins. This is how the Smart Contracts were born, which as Xu (2016) indicates "were introduced as stand-alone programs that run throughout the Blockchain network and can express triggers, conditions and business logic to allow for complicated programmable transactions" (p. 1).

Taking into account that intelligent contracts can be a complete program stored in a blockchain platform and distributed over all the nodes of the network. In order for this contract to exist within the network, Destefanis must proceed, as indicated, to store the intelligent contract in the block chain by means of a contract creation transaction, to which an address is assigned to identify it, which is generated as long as the creation transaction has been successfully executed (Destefanis et al., 2018, p. 21). Once these contracts are executed over the block chain they are responsible for managing the assets that such a platform includes through transactions that go beyond simple currency purchase/sale transactions, and may have more extensive instructions built into them all without depending, as Gürkaynak (2018) explains, "on an intermediary party, such as a bank or government agency, for value transfers, while providing the parties involved in the transaction with absolute confidence in the validity and security of the transaction" (p. 848).

3.2.1 Structuring of intelligent contracts

As shown in Figure 2, an intelligent contract basically consists of an account balance in virtual currencies in this case Ether which is the Ethereum's cryptomoney, a private storage and an executable code. This code when stored by a transaction as mentioned by Luu (2016) "is a 'standalone agent' stored in the block chain, coded as part of the 'creation' transaction that introduces the contract in the block chain" (p. 256), is for this reason that it is identified by assigning a unique address of 20 bytes and once introduced in the block chain can not be modified.

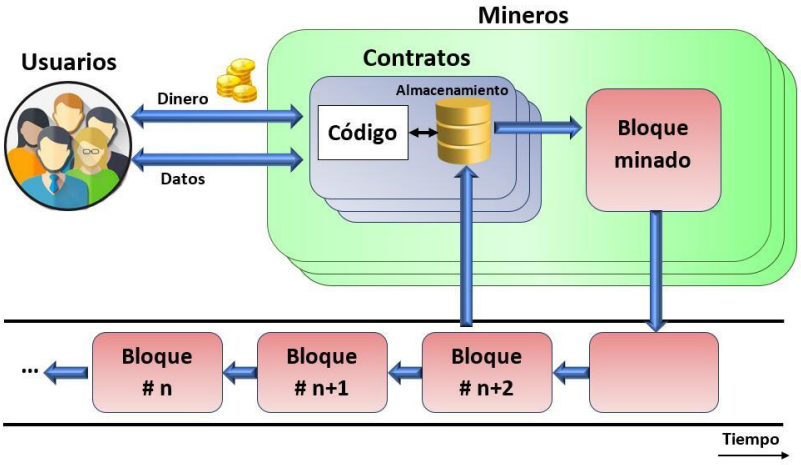

Figure 2. Structure of the elements that make up an intelligent contract (Alharby y Moorsel, 2017).

The additional step required for the contract to be inserted in the block chain is for the creation transaction to be within the set of transactions that will make up the Merkle tree and to be added to the block that is to be mined and included in the block chain.

3.2.2 Smart contracts work

Generally speaking, transactions sent to an intelligent contract go through three phases, the first being the inputs, the second corresponding to the contract interpreter and the last being the outputs as shown in Figure 3 and detailed below.

- **Entries:** in this phase you specify the contract identifier, the transaction request, any dependencies that may exist and the current status of the general ledger**.**
- **Contract Interpreter:** This phase is loaded with the current general ledger status and the intelligent contract code. These transactions are processed according to the procedure outlined in Hyperlerger magazine (2018) which states that "when the contract interpreter receives a request, it immediately checks it and then rejects any invalid request" (p. 4). The contract can, depending on the transaction it receives, read/write to your private storage, store money in your account balance, send/receive messages or money from users/other contracts or even create new contracts.
- **Outputs:** if the request is valid, outputs are generated including, a new status and any side effects. When all processing is complete, the interpreter packs the new status, a correction statement and any order suggestions required for the consensus services. That package is sent to the consensus service for final engagement with the block chain..

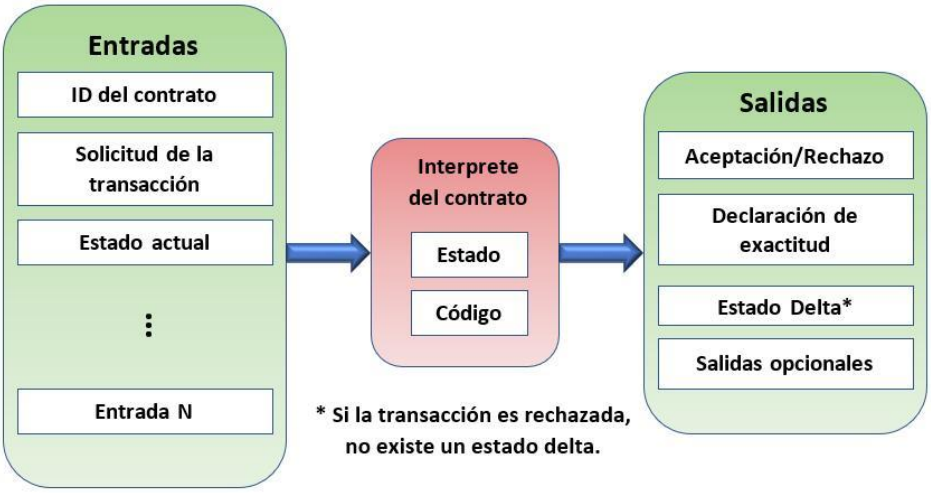

Figure 3. Phases to be followed for the execution of an intelligent contract (Buterin, 2009)

The state that the contract has at the moment of receiving a transaction will change to a delta state in case the transaction is executed correctly, if the transaction request is not validated, the contract code will not be executed and therefore a delta state cannot exist since the current state has not been altered.

3.3 Ethereum

Ethereum was conceived at a time when people recognized the power of the Bitcoin model and were trying to go beyond the applications of cryptomontages. That's how the young programmer Vitalik Buterin, who had a certain passion for Boitcoin, brought Ethereum to life after spending a series of stages beginning in 2013 when he began to think about how to expand the capabilities of Bitcoin and Mastercoin by proposing in October of that year a more general approach in which he conceived more flexible contracts with which to replace

the language of the specialized Mastercoin contract. Thus by December he began to share a technical document describing the central idea behind Ethereum which consisted, as Antonopoulos (2018) mentions, "in a chain of blocks that is complete and general-purpose Turing" (p. 41).

This platform is different from the previous Blockchain systems and besides being the leading Blockchain protocol in terms of innovation it is known, as explained by Dika (2017), "as the world's computer and as the future of the Internet with block chain technology" (p. 8). This is due to its novel idea of distributed computational processing of applications without the intervention of third parties and the predominance of transparency, it can also be mentioned that it is an open source platform which represents a cultural change from some of its predecessors.

The fact that Ethereum represents a chain of blocks with a complete Turing programming language built in, means that according to Vujičić (2018) "Ethereum supports all types of calculations, including loops and state transition, as well as other improvements to the structure of the chain of blocks" (p. 4). Within the Ethereum block chain, Ether (ETH) crypto-currency is handled to enable payment of financial transactions and processing of applications. Such applications as mentioned in the publication of coinpy.net (2018) "can be programmed in seven different languages including JavaScript, Go, Python and Lisp" (p. 3).

3.3.1 Ethereum Accounts

Within Ethereum as Buterin explains there are two types of accounts that can be seen as externally owned accounts to those that do not have contract code in their storage, and the other type of accounts are contract accounts, created specifically to execute the code of the intelligent contracts they host in their internal storage (Buterin, 2009, p. 13).

These accounts, as Buterin (2009) explains, are composed of four fields:

- The nonce, a counter that is used to ensure that each transaction can only be processed once.
- The current ether balance of the account.
- The contract code of the account, if any.
- The default empty account storage (p. 13).

Non-contract accounts also have their internal storage which will remain empty since it is intended to store the smart contract code, and since externally owned accounts do not handle contracts, they have nothing to store in their internal storage.

3.3.2 Messages and transactions

In Ethereum the concepts of messages and transactions are handled and as Buterin (2009) explains:

> The 'messages' in Ethereum have some similarity with the transactions in Bitcoin, although with three characteristics that differentiate them. The first is that an Ethereum message can be created by an external entity or by a contract, while a Bitcoin transaction can only be created externally. The second is that there is an explicit option for Ethereum messages to contain data. And the last is that the recipient of an Ethereum message, if it is a contract account, has the option to return a response;

> this means that Ethereum messages also cover the concept of roles (p. 14).

These three features give Ethereum messages a clear advantage over Bitcoin transactions because, although these features make Ethereum messages more complex to manage, they also allow you to expand the areas in which you can apply them by allowing you to perform data transport and execute functions that use this data.

On the other hand, with respect to Buterin transactions (2009) it mentions that:

> The term 'transaction' is used in the Ethereum to refer to the signed data packet that stores a message to be sent from an externally owned account. Transactions contain the recipient of the message, a signature that identifies the sender, the amount of ether and the data to be sent, as well as two values called STARTGAS and GASPRICE (p. 14).

As mentioned by Buterin STARTGAS, it refers to the limit of the number of steps that are given to execute the requested code for a transaction in such a way that, if the transaction is not completed in the determined number of steps, it is interrupted and ends up reversing all the changes avoiding that the execution is carried out in an infinite way. As for the GASPRICE, this refers to the fee that must be paid to the miner for each computational step he takes to carry out the execution of the transaction (Buterin, 2009, p. 14).

Here the concept of gas comes into play, which as Ast (2018) mentions in his publication in Medium, "gas is the unit that measures the computational work required to execute transactions or intelligent contracts in the Ethereum's virtual machine" (p. 2). In other words, if during the execution the "balance" for executing transactions is terminated, all status changes will be reversed, except the payment of fees, and if the execution of the transaction is stopped with some gas remaining, the remaining part of the fees will be refunded to the sender.

3.3.3 Smart contracts in Ethereum

An intelligent contract from Buterin's perspective can be used to represent virtually any type of asset that can be digitized by writing the logic in a few lines of code within an intelligent contract (Buterin, 2009, p. 1). With this in mind, the contract code is written in a programming language accepted by the platform. Once the code is written, it is enough to load it in, enter the initial variables and send it to be processed and executed. For its execution, the contracts require firstly a special software called Ethereum Virtual Machine (EVM) that in turn runs in each of the nodes of the Ethereum network and secondly the transformation of the code to bytecode which is the language that understands the Ethereum Virtual Machine.

Since Ethereum allows users to load and execute code that represents the contracts, these can be simple or arbitrarily complicated, although it should be taken into account as mentioned by Kiffer (2017) that "each operation that the code executes, and each byte of memory that the code uses, costs 'gas'" (p. 95), that is, the more complex these contracts are, the more ether they will spend for their execution. At Ethereum the contracts also have their own balance of ether, and can even transfer ether and call other contracts, have their own storage, and have the ability to act as an external proprietary account at Ethereum.

4. Experimental methodology

The term methodology refers, according to Quecedo and Castaño (2002) "to the way we approach problems and seek answers, to the way we conduct research" (p. 7). This gives us the guideline that the methodology used to solve a problem can vary depending on how the problem is looked at and also on how the answers are proposed.

The methodology occupies an important place within the research process, so that for Rodríguez and Valldeorialo (2014) "the methodology is fundamental in any research process, since it determines the way in which the research is developed" (p. 31). Considering these aspects on the methodology, it has been decided to direct this investigation by means of the descriptive method, which, as indicates Pérez (2004) "is oriented towards the present and the levels in which it acts are the applied investigation and active investigation" (p. 91) which is precisely what is sought in this investigation.

Therefore, for the development of this article, the authors propose the methodology detailed below:

- **Determine the functionality of the intelligent contracts:** as a first point, it is required to establish exactly what our intelligent contracts are going to do, so that we can determine the resources and tools necessary for the development of the work.
- **Description of the required architecture:** the next step is to carry out a detailed analysis of the contract's functionalities to establish the requirements of the architecture and the working environment necessary to carry out the development, in order to have a clear idea that allows to prepare this working environment in an adequate way.
- **Selection of tools:** once the requirements for the development of the work are clear, the next step is to select a set of preferably free software tools that support the architecture and the whole environment specified by the requirements of the first phase.
- **Development of the intelligent contracts of the application:** with the architecture and the work environment ready, the creation of the intelligent contracts can continue, so this phase is focused on the creation of files and writing of code based on the following guidelines:
    - Manage a directory structure that allows you to maintain the order of the files according to their functionality
    - oUse words for file names that identify the purpose for which they were created and the functionality they serve within the project
    - o For the code of intelligent contracts, name the variables and functions in such a way that by reading them you can clearly understand what their purpose and function is within the contract.
- **Deployment of the intelligent contract:** the objective of this phase is to carry out the procedure for the deployment of intelligent contracts which consists of inserting these contracts in the block chain. At the same time, monitoring of transaction generation and block mining is carried out to verify that it has been deployed correctly. This helps to verify possible errors and to understand how the process works. Within this phase three important tasks are considered which are:

  - oDebugging, to verify that there are no errors or inconsistencies within the written code in order to prevent the contract from being executed incorrectly or producing unexpected results.
  - oCompilation, to convert the source code into the binary that is required to execute the intelligent contract.
  - oDeployment, which involves sending the compiled Smart Contract code to the network so that it is available to users and can be executed in the Ethereum virtual machine.
- **Interaction with the intelligent contract:** this phase aims to interact with the intelligent contracts by calling the functions that make them up. In this way, data will be recorded and consulted while monitoring is carried out at the same time. Firstly, the outputs produced by the execution of the functions are verified and secondly the generation of transactions and the mining of blocks to add them to the block chain.

5. Proposal development.

5.1 Smart contracts in the registration of academic titles

5.1The idea of using the chain of blocks and smart contracts in the process of registering academic degrees is to provide a reliable and secure way to verify the existence and authenticity of the degree a person has earned. So that, when you look at this information, you can be sure that it is true and has not been altered by knowing that it is stored in an immutable record such as the chain of blocks.

As intelligent contracts are self-executing and automatically implement the terms of the agreement between two parties, they allow for the streamlining of processes by providing, as Toyoda (2017) mentions, "the ability to identify forgeries if any inconsistency is found in the process" (p. 2). This feature will prevent the registration of undue titles and will guarantee that the titles assigned to persons are real and cannot be modified due to the difficulty of doing so because of the logic with which the blocks in the chain are generated, in which each one is cryptographically linked to the previous one.

The main advantage of registering academic titles through intelligent contracts based on the block chain is that this covers a large number of additional characteristics such as immutability, transparency, security and decentralization. This set of features changes the perspective of the way this activity is carried out today and makes it possible to be part of a technology that augurs well for the future. What is currently most welcome are the decentralized systems and the generation of trust between unknown entities through the use of tools such as cryptography.

5.2 Development of the intelligent contract

Figure 4 shows the general outline of the process to be followed for the elaboration of the intelligent contract, which ranges from codification to deployment in a private block chain by making use of the set of tools offered by Truffle.

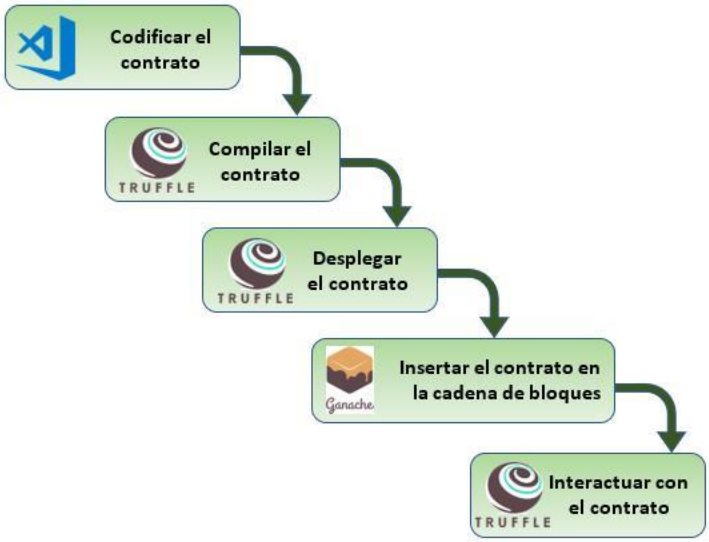

Figure 4. Diagram of the phases for the development of an intelligent contract from codification to deployment

For the development we used open source tools such as Visual Studio Code (VSC) which was used to write the code that, despite being a Microsoft product is free. Likewise, the applications of Truffle's toolkit are all open source so they can be used without the need for licenses.

5.2.1 Defining the Intelligent Contract for Securities Management

5.2.1.1 Defining the compiler versionr

In order for the intelligent contract code to be executed it must be compiled and as Antonopoulos and Wood (2018) state "the command line compiler for solidity is solc" (p. 134). This compiler allows to convert, as Antonopoulos mentions, the solidity code into binaries transformed in turn into hexadecimals which is what he understands and will be executed in the Ethereum virtual machine (Antonopoulos & Wood, 2018, p. 134). For this reason, any intelligent contract must be started by indicating the compiler version that you are going to use, this is done in order to avoid that future compiler versions may introduce incompatible changes at the time of compilation. At the time of this work the last stable version of the compiler is v0.5.2 but we have chosen to use the experimental version ABIEncoderV2.

5.2.1.2 Declare the contract

For the declaration of the contract, the word reserved contract is used which, as Antonopoulos (2018) states, "is similar to a class declaration in other object-oriented languages" (p. 28). The word contract is followed by the name of the contract file, which is conventionally named using the CamelCase structure. Finally Antonopoulos and Wood mention that it opens and closes keys within which all the contract logic will be written, thus defining the contract itself and its scope as it happens in several programming languages (Antonopoulos & Wood, 2018, p. 28).

5.2.1.3 Declare state variables

A smart contract according to Ethereum (2017) "is a collection of code (its functions and data (its status) that reside at a specific address in the Ethereum Blockchain" (2017, p. 13). For state variables in this case, an address variable is handled to assign the address of who

deployed the contract. The remaining three variables of the mapping type allow you to relate an address to a list of data of type Title, StudentDegree and Student respectively, in which records of these data structures will be stored.

5.2.1.4 Declaring the contract constructor

The intelligent contract constructor is a function that is executed only once when it is instantiated and as mentioned by Antonopoulos and Wood the way the constructor is written depends on the compiler version being used, so it can be written as a function with the same contract name (for compiler versions up to 0.4.21) or with the reserved word constructor (for versions from 0.4.21 and above) (Antonopoulos & Wood, 2018, p. 143).

Within the declaration it is indicated which actions will be performed to initialize the contract, in our case we will indicate that the owner state variable will be assigned the address of the person who created the contract.

5.2.1.5 Defining Data Structures

Data structures or structs according to Ethereum are more complex data types that are used to represent real-life objects and are formed by grouping together several variables of primitive data types (Ethereum, 2017, p. 23). These structures allow you to extend the functionality of intelligent contracts by handling custom and more complex data types, as they can include not only primitive data types but other structures.

For the creation of the contract, according to the authors' criteria, three data structures were defined to manage each of the objects that are part of the registration process of academic titles. These structures are described below:

1. The data structure Title containing three fields: a string type identifier, a string type name, and an unsigned integer that will store the timestamp on which a title is created.
2. The Student data structure containing four fields: the string ID, the student name, the unsigned integer timestamp and the title boolean to indicate if the student has been assigned a title.
3. The StudentTitle data structure containing five fields: the name of the reviewer of type string, the grade with which the title of type uint or unsigned integer is recorded, the timestamp in which it is created of type unsigned integer, the student of type Student defined by a struct and the title of type Title defined by a struct.

5.2.1.6 Defining function modifiers

Function modifiers as explained by Ethereum (2017) "are a convenient way to validate function inputs" (p. 29). These modifiers constitute a property that allows changing the behavior of the functions within the contract. The most common way to use them is to check the fulfillment of a condition before executing the function. For our contract, we have created a modifier that checks and forces the owner of the contract, i.e. the one who created it, to be the only one who can execute the function to which this modifier applies..

5.2.1.7 Functions defined in the contract

For our contract we defined several functions classified as primary and secondary according to the task they perform within the contract as summarized in Table 1.1.

Identifier	Type	Description

RegisterNew Student	Main	Allows you to enter the registration of a new student by providing the name and ID
RegisterNewTitle	Main	Allows the entry of a new title record by providing the title identifier and name
RegisterStudentDegree	Main	Allows a student to register a degree by providing the name of the reviewer, the degree grade, the student's ID and the degree identifier.
check outStudents	Secondary	Searches the student register by providing the student's ID.
CheckTitles	Secondary	Performs a title registry search using the title identifier.
GetStudent	Secondary	Allows you to extract a student from the register by means of his/her ID.
GetTitle	Secondary	Allows to extract a title from the register by means of its identifier.
GetStudentList	Secondary	Allows to retrieve all the existing student records.
GetTitleList	Secondary	Allows to retrieve all the existing records of the titles.
GetStudentListDegrees	Secondary	Allows to retrieve all the existing records of the graduated students.

Table 1. Summary of contract functions Titles

5.2.2 Creating the intelligent contract for chain managemen

As in the previous case, the first thing to be defined in the contract is the version of the compiler to be used, followed by the definition of the contract and within this its respective functions, which are summarized in Table 2.

Identifier	Type	Description
compare()	Main	It allows the comparison of the two text strings by returning an integer response that indicates whether the strings transformed into bytes are equal or not.
equal()	Secondary	It is in charge of receiving the two string type parameters that you want to buy and then send them to be processed in the compare() function.

Table 2. Summary of the functions of the Stringutils contract

5.2.3 Define the migration files of the contracts

To deploy the intelligent contracts it is required to build a migration script which will allow the implementation of the contract and as indicated by Antonopoulos and Wood, one script can be created for each one, or a single script can be created to gather all the contracts so that they can be deployed sequentially (Antonopoulos & Wood, 2018, p. 241).

For the deployment of our contracts, a single migration file is used for the two generated contracts, Titulos and StringUtils. The structure of the script is quite simple, it consists of two variables of which each one refers to the file of the intelligent contract that is desired to deploy, then it makes use of the special object of node.js module.exports that will allow to expose the contracts as modules assigning to it the result of the deployment of the intelligent contracts, this in turn makes a call to the asynchronous function doDeploy() that receives as a parameter the object deployer that will allow to deploy the contracts and to link them since Titulos imports StringUtils to make use of its functions..

5.3 Smart contract deployment

In order to deploy the intelligent contracts, it is required to have a chain of Ethereum blocks, which as Antonopoulos and Wood suggest will be simulated locally, generating a private instance using the Ganache tool (Antonopoulos & Wood, 2018, p. 234). Figure 5 shows the result of the execution of the Ganache tool which will generate this private chain, in addition to 10 accounts with their respective addresses and 100 initial ethers that will be used to process the transactions.

One of the most important aspects of this tool is that as explained by Antonopoulos and Wood (2018) "it offers an Application Programming Interface and a set of Remote Procedure Call commands coded as Javascript Object Notation usually called JSON-RPC API" (p. 52). The Ganache tool, by default provides this interface at the localhost address and port by 7545, which is where you must connect to interact with the block chain. .

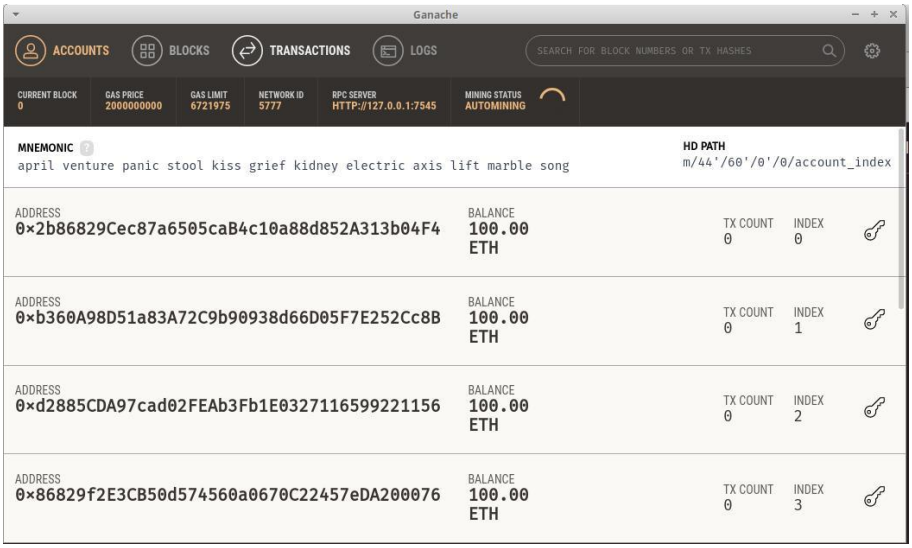

Figure 5. Screenshot of account generation in the Ganache tool

With the Ganache tool running and the RPC server up, the Truffle configuration file called truffle-config.js must be implemented in which the network of the block chain to be connected must be specified, i.e. the address of the server of the block chain provided by Ganache.

With all the code and configuration files generated and with the tools prepared, all that remains is to deploy the intelligent contract for which the Truffle tool will be used or more specifically the truffle migrate command which will execute all the migrations specified in the migration files which are generally located in the migrations directory and will in turn

generate a new directory in the project called builds in which there is a contracts directory containing a json file for each of the contracts in which there is information corresponding to the deployment of the contracts.

5.4 Interaction with the intelligent contract

Figure 6 shows the outline of the interaction process with the intelligent contract once it has been deployed. The most basic way to interact with the contract is through the console that provides the Truffle tool, which as explained by Antonopoulos and Wood (2018) "is an interactive JavaScript environment that provides access to the Truffle environment and, via web3, to the block chain" (p. 235). Within this console, one can instantiate the intelligent contract that resides in the block chain, and through this instance make calls to the functions defined in that contract.

To be able to execute the functions of an intelligent contract, you must have an instance of it. To obtain it, through the Truffle console a variable is assigned to the contract display by executing the command Titulos = Titulos.dployed() which will generate an output corresponding to all the information of the contract, such as the name, the code that composes it, the compiled code and the ABI or Binary Application Interface which, in Ethereum, is basically the way in which the contract calls can be made.

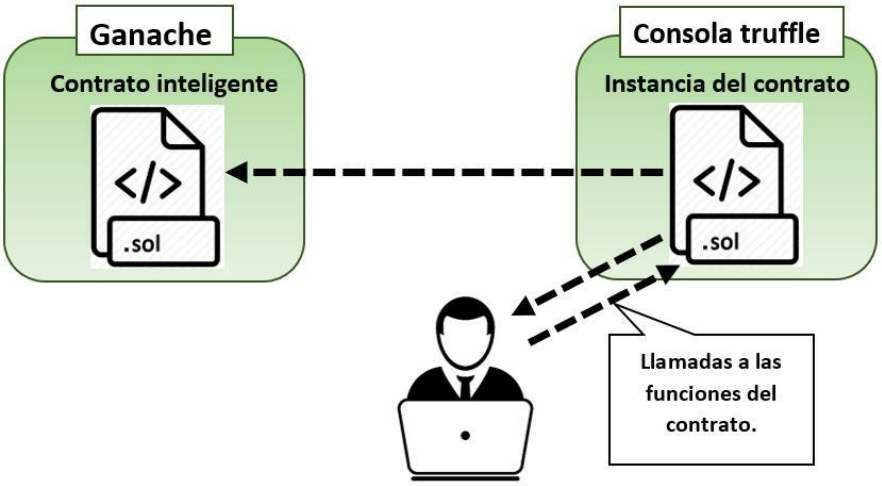

Figure 6. Diagram of the process of user interaction with the intelligent contract once it has been deployed

Once the contract instance is obtained in the te Truffle console, the commands executed through this instance will be reflected in the block chain as transactions and mined blocks as long as these transactions have been successfully executed.

6. Conclusions

Through this research it has been possible to establish an application model together with all the technological components that make up the architecture based on blockchain technology for the registration of academic titles. In addition, it has been possible to verify the feasibility that blockchain presents to be used as an architectural basis for the development and deployment of intelligent contracts. For this purpose, two intelligent contracts were designed, a main one to manage the creation of registries for both degrees and students, as well as to allow for the assignment of degrees; and a secondary one that

serves as a support by providing additional functionalities required by the main contract in order to carry out its tasks.

It was possible to verify that the methodology proposed in this article is valid for the development of intelligent contracts. This is due to the fact that the phases proposed in this methodology allow for a clear picture of the development from start to finish. In addition, in each phase an analysis is made or a set of specific steps is followed that allow to speed up the processes and reduce errors.

To speak of Blockchain is, without a doubt, to speak of a significant technological revolution since it offers an extraordinary potential especially in those areas where a reliable and immutable registry of each transaction is required as for example the registry of academic titles, that's why its utility transcends the cryptomonalities and with the appropriate instruments as for example Ethereum, all that potential can be exploited.

Making the registration of academic titles through intelligent contracts on Blockchain technology would generate an enormous impact for higher education since it would allow to maintain a highly reliable record of the titles that a person has without being able to modify it and at the same time it is available to the general public so that they can access this record either for information purposes or for data validation ensuring that if a title has been assigned to a student within this record in the blockchain it is legitimate.

Reference

Álvarez, R., Andrade, A., & 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., & Wood, G. (2018). *Mastering Ethereum* (First). USA: O'Reilly Media, Inc.

Arenas, R., & 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 & Decentralized Application Platform,

(January). https://doi.org/10.5663/aps.v1i1.10138

Chen, X., Zheng, Z., Xie, S., Dai, H.-N., & Wang, H. (2018). Blockchain challenges and opportunities : a survey. *Inderscience Enterprises Ltd.*, *14*(4), 352–375.

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

Singh, M., & 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., & 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., & 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., & 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., & 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

Authors

LUIS ROSERO-CORREA. He is a computer engineer graduated from the Central University of Ecuador.

MARIO MORALES-MORALES. He is a systems engineer graduated from the Escuela Politécnica Nacional, Ecuador. He studied a master's degree in business administration at the Universidad San Martín de Porres, Peru. He has obtained certifications in project management (PMI) and data analysis, with extensive experience in business projects in the Andean region.

He currently teaches at the Faculty of Engineering, Physical and Mathematical Sciences of the Central University of Ecuador, and is studying for a PhD in Computer Science at the University of Alicante.

SANTIAGO MORALES-CARDOSO. Doctor in Computer Science from the University of Alicante, Spain. He obtained a degree in computer engineering, a master's degree in engineering sciences and a master's degree in computer business management at the Central University of Ecuador.

He currently teaches at the Faculty of Engineering, Physical Sciences and Mathematics.