UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE GRADO EN INGENIERÍA INFORMÁTICA:

Healthchain: Registro médico electrónico en una red Blockchain

Tutor: José María Álvarez Rodríguez

Autor: María Teresa Nieto Galán

Febrero 2017

Agradecimientos

A mi padre y a mi madre, gracias por haberme educado en los valores que me hacen ser quien soy y los que me han hecho llegar hasta donde he llegado. Gracias por enseñarme que con constancia y esfuerzo puedo conseguir todo lo que me proponga.

A mis compañeros de clase, gracias por haber hecho que mi paso por la universidad sea una experiencia inolvidable. Que a pesar de todas mis quejas y del estrés que hemos estado teniendo todo este tiempo, nunca olvidaré nuestros 9-21 por la universidad y nuestras queridas "Fs".

Olmo, gracias por ayudarme a ver el camino que tantas veces los árboles no me dejaban vislumbrar.

Finalmente, a mi equipo de Blockchain y Fran, todo este trabajo es gracias a la "envidia" que me causasteis mientras yo estaba aprendiendo a mover cajitas y vosotros dando gritos. Gracias por invitarme a ser una más en este mundo tan incierto e interesante.

“Todos somos genios. Pero si juzgas a un pez por su capacidad para trepar árboles, vivirá toda su vida pensando que es un inútil”

Albert Einstein

Resumen

El presente documento corresponde a una parte del Trabajo Final de Grado del Grado de Ingeniería Informática.

Dicho trabajo está compuesto por dos partes fundamentales: el análisis, diseño e implementación de una aplicación web y el presente documento, que recogerá la parte técnica realizada para conseguir dicho propósito.

Asimismo, gran parte de este trabajo se basa en un proyecto interno de investigación sobre la tecnología Blockchain.

Como veremos más adelante Blockchain, de forma resumida, es una base de datos distribuida que posee un conjunto de bloques ordenados cronológicamente. En dichos bloques la información es almacenada de forma íntegra y consistente. Estas características se cumplen debido a que los bloques poseen un enlace entre ellos, de manera que, no se puede realizar ningún tipo de alteración.

El proyecto se basará en un análisis sobre un caso de uso aplicando la tecnología anteriormente mencionada.

En la actualidad hay múltiples casos de uso que pueden servir como ejemplo de aplicación de esta tecnología, con el propósito principal de solucionar problemas a los cuales no se les ha puesto remedio hasta la fecha.

El caso de uso elegido, que corresponderá el eje central del proyecto, es el historial médico electrónico. Hasta la fecha, no existe un estándar del mismo utilizado por toda la comunidad médica tanto pública como privada.

Asimismo, en otros países como Estados Unidos, los historiales médicos también son gestionados por entidades externas a la salud y, en ocasiones, los datos del paciente se ven comprometidos.

Con el propósito de solucionar el problema mencionado que se detallará posteriormente, a lo largo de este proyecto se irá diseñando, utilizando la ingeniería del software, la aplicación web que pondrá remedio a ciertos problemas que en la actualidad no son contemplados.

Abstract

The present document is part of the Final Project Degree of the Degree of Computer Engineering.

This project comprises two fundamental parts: the analysis, design and implementation of a web application and this document, which is the technical part of the project.

Furthermore, part of this work is based on an internal project of research on the Blockchain technology.

As we will see later Blockchain, in summary form, is a distributed database that has a set of blocks sorted chronologically. In those blocks the information is stored in full and consistent form. These features have been met because the blocks have a link between them, so that it is not possible to perform any kind of alteration.

The project will be based on analysis on a use case applying the technology previously introduced.

Currently, there are multiple use cases that can serve us as an example for the implementation of this technology, with the main purpose of solving problems to which they have not been remedied to date.

The selected use case, which will be the main *leitmotif* of the project, is the electronic health record (EHR). Up to the date, there is not yet a complete solution to address the challenges of this area in terms of privacy and security.

Besides, in other countries such as the United States, medical records are also managed by external entities to health, and sometimes the patient data are compromised.

In order to address this challenging environment, an application of Blockchain will be designed using a software engineering approach. A demonstrator of the blockchain technology will be presented through the development of an application taking advantage of the new functionalities

Overview

Blockchain is one of the most promising technologies up to the date. Along this document, an analysis about this technology and a case study on the field of e-Health will be presented and implemented to the demonstrate the capabilities of Blockchain-based technology.

The main motivation relies on the possibility of being able to solve a real problem using Blockchain technology. Furthermore, as a main secondary goal, the project must serve us to learn a new uprising technology that is still under development.

On the other hand, the problem to tackle with Blockchain technology is the secure management of the Electronic Health Record. Electronic health records are poorly managed in terms of security and privacy and, in general, do not belong to the patient. They are usually managed by doctors or insurance companies.

Besides, in countries like the United States of America, there is a problem regarding privacy. Once medical information is transferred to some entity, patients are not anymore owners of their health records implying that their own information can be used for third-party benefit.

What is the meaning of “Blockchain”?

Blockchain is seen as one of the newer and innovative technologies in the current technology landscape. However, as it will be presented later, although it has not yet gained much presence, Blockchain is a technology that emerged years ago and it is currently gaining momentum.

This field of the computer science might be conceived like a type of Internet, an information technology at staggered technical levels and diverse classes of applications for any form of record of assets, inventories and exchange, including the economic area.

However, the concept of this technology goes far beyond. It is a kind of new paradigm organized and oriented to the discovery, evaluation and transfer of all types of data. As any intention of any existing technology so far, Blockchain was conceived with the aim of changing the way in which we interact with our world, transparent and timely.

This way to change the world, can be applied to many sectors such as financial and health sectors that are somehow quite connected since improvements in health management may imply reduction of costs and deliver of better services for the society.

An important question is, how Blockchain works? Blockchain is a kind of distributed database that works like a ledger that stores the record of any type of transaction. Each transaction is verified by a consensus of the majority of the participants in the system.

That is why, each block contains the information you want to keep, either a transaction or, as we shall see later, a reference to a smart contract.

All blocks, sorted through a chronological order, which give rise to the chain have a hash of the previous block. Thus, a block has a reference to its previous block and once a transaction enters to the system, can no longer be deleted. It is therefore virtually impossible to alter a block that has been stored for a certain time in the chain.

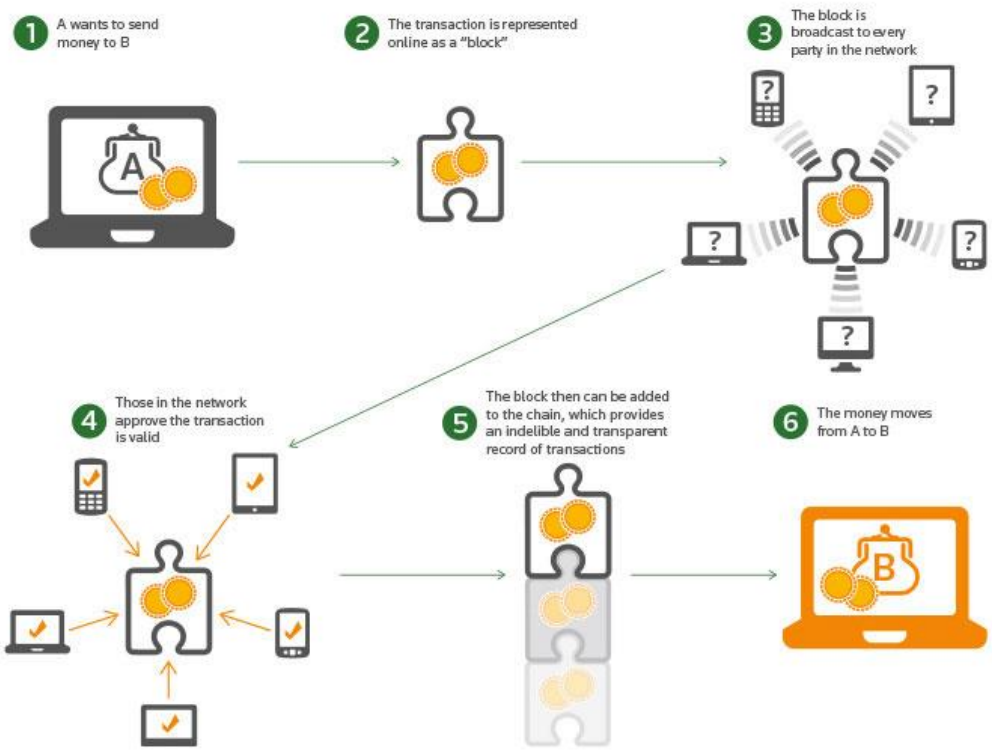
Blocks are continually growing, after a certain period of time, depending on the Blockchain network, a new block is created and it is possible to store the record of the most recent deal. For example, in a network Bitcoin writing of a block can take up to 10 minutes.

Bitcoin, the digital currency peer-to-peer, is the most popular application using this technology. This type of currency is somewhat controversial. However, the underlying technology, Blockchain, has worked at the moment without problems and has many successful use cases in both the financial and non-financial sectors.

The main benefits of Blockchain are much more than economic, is extend to the political, science, social, health. On the other hand, the capacity technology of Blockchain is being exploited by groups specific to addressing problems of the world real.

For example, the coordination, the storage of records and the irrevocability of transactions using this technology are characteristics that could be instrumental in the improvement of society. As an idea, it could provide repositories of public records for all types of societies, including the registration of documents, events, identities and goods. This means that all potential tangible assets (such as cars and houses) and digital assets may be registered and made a transaction in the network Blockchain.

Many blockchain applications can be found but most of them are currently focusing on digital banking. An example of how it works is shown in the following image:



The principal Blockchain use cases are the following:

1. **Loans and peer to peer pays**

Up to the date, all types of bank stocks entail too run time. For example, if an entity wants to make a shipment of money to an entity B, due to the current situation, this process would take days to run.

Using blockchain intermediaries would be eliminated so that the time would diminish greatly and, on the other hand, the associate costs also would descend.

2. **Internet of things (IoT)**

IoT systems are based on models of communication centralized, using the paradigm known as client-server.

Using a decentralized model, using peer to peer protocol, it would provide a new communications architecture and would reduce significantly the costs associated with the installation and the maintenance of big centralized data centres.

3. **Voting Systems**

Electronic voting systems suffer a great defect in design: to use a centralized system. This implies that only there is a provider that controls the code, the database and the outputs of the system.

Using Blockchain as a secure database and transparent would solve this problem and could register the votes of a reliable way.

4. **Decentralized Markets**

A decentralized market is a structure of market that consists of a network of different devices that allow to create a market without a centralized place. This way, the technology offers to the buyers the access to several offers, making the direct dealing possible with other distributors.

5. **Land registry**

Currently, the records of the property have some scourges such as fraud and corruption.

The justification for the application of Blockchain to this field implies, decrease the costs of performing the property register, achieve transparency and security, eliminating corruption and fraud in the process and the most important, provide fastness title.

6. **Digitization of assets and documents**

With the increase number of data types and formats, the need to integrate and share data through systems has become essential. For the greater part of organizations, this involves the balance delicate of those processes that move data between systems.

7. **Authentication of assets**

The life future of goods can be drastically modified, through the existence of a record that contains its cycle of life and allow his tracking of the chain of supply.

Classification of Electronic Health Record (EHR)

As we have previously introduced, this project also addresses the study of current situation in the field of electronic health records and a possible application of the Blockchain technology to this area.

Regarding one of the key fields in EHRs, there are many classifications for “tagging” electronic health records. However, there is not a universal standard that everyone uses. Among the most popular electronic health records we can find the following:

1. **HL7 (Health Level Seven)[1]**

It is a non-profit organization accredited by the ANSI organization. It is dedicating to proportionate a standard to the electronic integration, exchange and recovery of the health information.

Level Seven is referred to the level seven of the International Organization for Standardization (ISO), the seven layer of the model: the layer application. The level of the application interacts directly and performs common application services for the processes of the application.

2. **UMLS (Unified Medial Language System)[2]**

The Unified Medial Language System is and standard created by de National Library of Medicine (NLM) with the main target of facilitate the develop of computer systems that works with the meaning of the health.

The NLM produces and distributes the databases with the health information and the software tools. The main target is that the developers use all these tools.

3. **SNOMED CT[3]**

SNOMED Clinic Terms is an organized and processed team that includes a recompilation about medial terms and allows proportionate codes, terms, synonyms and definitions used in the clinic documentation.

SNOMED CT insures the exchange of information and it is fundament for an interoperable electronic health record.

4. **ICD 11 (International Classification of Disease) [4]**

The ICD was adopted by the Statistics International Institute in 1893. The classification is maintained by the WHO (World Health Organization).

It is designed like a classification system that proportionate a huge code system of diagnostic to the classification of Diseases.

At the present the last revision is the eleventh.

The Software Engineering project

The best way to solve the problem about the Electronic Health Records using the Blockchain technology is doing a software engineering problem. Following the software engineering principles, it is necessary to do an analysis about the needs that electronic health records system would need to work. Secondly, the next stage is to design a solution that would include the functionality and, finally, is the development of the solution including tests to verify and validate that the system meets the client needs and requirements.

The main goal of the analysis is to study what is expected to design. The study is made up by the following components:

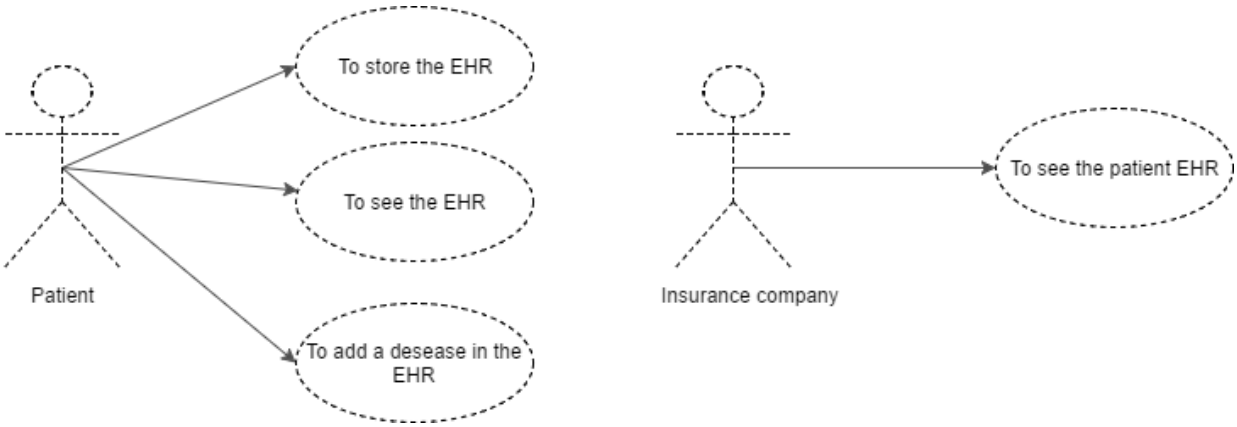
User requirements: Described in a formal language functional requirements and non-functional requirements on the system.

Use cases: are a description of the steps or activities that must be performed to accomplish some process. The characters or entities that will participate in a use case are called actors. In this case there will be two actors, the patient and the insurance company.

To sum up the user requirements will be the following:

1. The user shall be able to store its electronic medical record in the system.
2. The user shall be able to recover its electronic medical record.
3. The user shall not be able to modify the stored data
4. The user shall be able to add new data to its electronic health record.
5. The system shall show its interface in English.
6. The interface shall be responsive.
7. The stored data shall be always available.
8. The user shall be able to have the control of its data and it will decide with whom to share them.

On the other hand, the use case diagrams will be the following:



Finally, taking into account user requirements and use cases, the next step is to define the system requirements, in other words, the funcional and non functional requirements.

Funcional requirements define funcionalities of the system or its components. On the other hand, non funcional requirements, called quality attributes, define all criteria that can be used to jodge the operation of the system in a specific behaviour.

The non funcional requirements are closely related with the funcional one.

When the analysis is finalished the next step is to do the design.

The design process describes the technical behaviour that must provide the application. Keeping this in mind, first it is necessary to do the components design.

The component diagram serves us to model the system and all its parts that will be deployed in the implementation phase. According to this architectural diagram, one can have an overview of the systems architecture. In other words, the component diagrams are used to make a representation of the static view of a system.

When the components design is done, it is necessary to chose the operational enviroment of the system. In this case, the better way to develop an architecture with a variety of technologies is using the microservices architecture.

Microservices is a new method of developing applications that have grown up in the recent past years. Using this architecture the software is split up in into multiple services that are independient between each other.

The benefits of the microservices are the following:

Simplicity of the services that make up the application.

- Interdependence.
- Reduction in the time of the development of the application.
- Multi-Technology.
- Deployment Units small.
- Fault Tolerance.
- The configuration management.
- Location of the services.
- Easy horizontal scale.
- Decentralization.
- Components recycling.

Every microservice must be independent to the rest and its deployment must not affect from any form to the rest of microservices that compose the application. In other words, this means that an application can be built in various services developed in different languages.

It should be noted that communications between the micro services are performed using HTTP REST . This is due to the REST API that have the same and exposed to the outside.

In this case there will be the following microservices:

- Front-1 and front-2: web interfaces of the application.
- Back microservice: backen of the application. There will be a server that will have all the business logic. It depends on the database component and the Blockchain components.

Finally, after having made the design of the apllication, the next step is proceed to the implementation of the same one.

In the implementation section, there will be detailed how was this process has been done. Then, once completed, there will be detailed the test cases to check if the behaviour of the application is the expected.

The tecnologies used in the implementation of the techology are the following.

1. **Docker[5]**

Docker is a platform used to create and run an application on any platform. Its code is implemented in the language created by Google, Go. Docker is open source.

It is used by the companies to simplify and to accelerate the process of the development of applications. On the other hand, it is used too for the deployment of the same ones. This is achieved thanks to the containers that are provided thanks to Docker.

The benefits of Docker are the folowing ones:

- Isolation between applications.
- Isolation between the applcation and the server.
- Promoting the adoption of the minimal privilege.
- Portability

2. **AngularJS[6]**

AngularJS is an open source framework developed by Google. It permit the creation of dynamic web interfaces using JavaScript.

AngularJS uses a dependency inyection created by the company that eliminates a huge part of code that should be developed by the application developer. In other words, it should not be necessary to implement a backend component in a web application. This type of web applications are known as Single Page Applications (SPA).

The benefits of the use of AngularJS are the following:

- Modular development.
- Great for Single Page Application.
- Templating.
- Two-way data-binding-

3. NodeJS[7]

NodeJS is an execution enviroment and a library of libraries writen in Javascript. It enables the management of I/O events and it is asynchronous.

NodeJS was developed by Google and is open source. It can be executed in any platform: Windows, Linux and MacOS.

4. Git[8]

Git is a software used by the version controls. It was developed by Linus Torvalds in 2005 with the main target of the improvements of the efficiency in the version control.

The version control is a system that registry the changes of a file o a number of files. In other words, if something wrongs happens with the code, the could have the availabilty to restore to a previous version.

The benefits of Git are the following:

- File recovery
- Write speed
- Branching
- Huge perfomance
- Distributed management
- Reduction of the costs
- Agil development
- Selective sharing

5. **MongoDB[9]**

MongoDB is an open source non-SQL database. It is the most used database in the present. The information is stored in JSON standard.

Its main benefits are: scalability, performance, lower costs and improve the user experience.

6. **Rest[10]**

REST, Representational State Transfer is a software architecture created by hypermedia distributed systems. These systems integrate text, audio, video, maps and other integration supports.

Rest is the most used standard in the present day in the area of APIs development.

Conclusions

The designed application can mark one earlier and one later in the management of the electronic health records. The management of the health sector has seen a great evolution in recent years, from physical management to digital management.

The problem, so far, is that the patient has no access to her own record, only it is managed by the doctors in Spain and by the same ones and the insurers in other countries such as the United States.

With the design and use of this application, the electronic health record would belong to the patient and all the changes that he will make will be conducted in a transparent and full manner.

The development of this project was born as an idea to make an application with a real use case using a new technology that has planned a great future, Blockchain.

Blockchain is still under study and development. There are some emerging Blockchain platforms that are still under development. On the other hand, there also are applications with a good state of maturity.

Blockchain seems to be the next revolution as it was Internet at the moment of its birth. All the characteristics that we have been analyzing along this project are very necessary features to boost digital economy and health (and many other sectors) reducing costs and deliverying high-quality and cost-effective services.

Tabla de contenido

Índice de ilustraciones

Índice de tablas

Introducción

Blockchain está visto como una de las tecnologías más nuevas e innovadoras en este nuestro mundo, la informática. Sin embargo, como veremos más adelante, aunque no haya cobrado mucha presencia, Blockchain es una tecnología que surgió hace años. Aun así, hasta la fecha, se encuentra en un periodo de investigación y crecimiento.

Este campo de la informática podría ser concebido como un tipo de Internet, una tecnología de la información con niveles técnicos escalonados y diversas clases de aplicaciones para cualquier forma de registro de activos, inventarios e intercambio, incluyendo el área económica.

Sin embargo, el concepto de esta tecnología va mucho más allá. Es un nuevo paradigma organizado y orientado al descubrimiento, evaluación y transferencia de todo tipo de datos.

Como todo propósito de toda tecnología existente hasta el momento, Blockchain nació con el objetivo de poder modificar la forma en la que interaccionamos con nuestro mundo, de forma transparente e inmediata.

Esta forma de cambiar el mundo, puede basarse desde el sector financiero hasta en el sector de la sanidad, sector sobre el cual se basará el caso de uso en el que se centrará este estudio. Si observamos la siguiente imagen, Blockchain aparece como una de las tecnologías emergentes en el momento con mayores expectativas a corto plazo. [11]

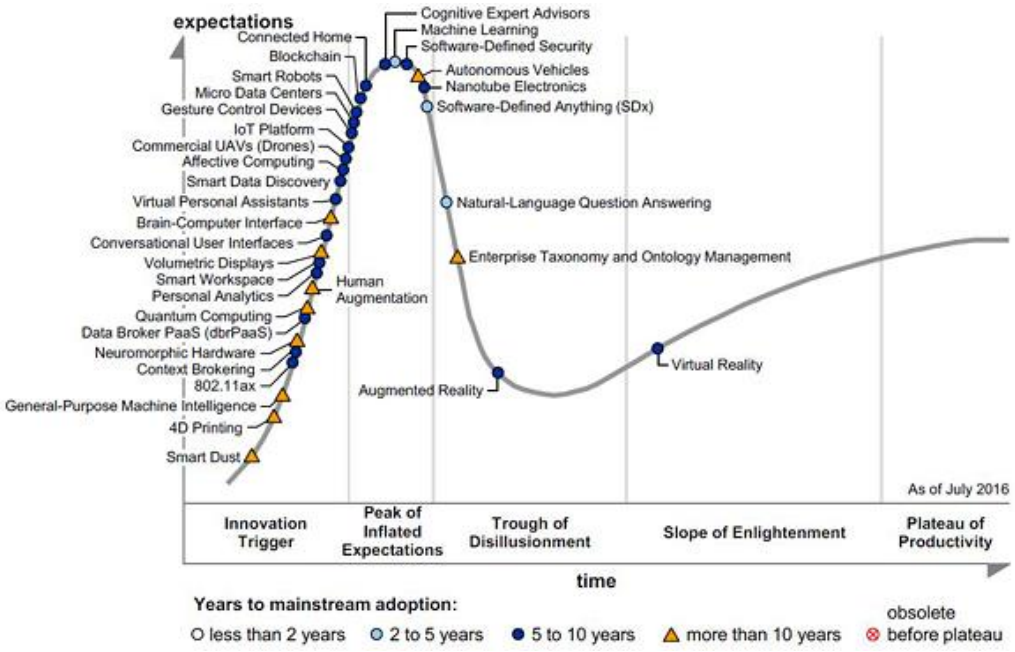


Ilustración: Gráfica de tecnologías emergentes en la actualidad. Source (Gartner. Julio 2016) [11]

[12] Actualmente, en la mayoría de los países existe el problema de la gestión de los historiales médicos, sobre todo en EEUU donde los historiales también son gestionados por aseguradoras. Desde que se estableció la ley federal de Portabilidad y Responsabilidad de Seguro de Salud en 2003, son los pacientes quienes tienen acceso a su historial médico y quienes deciden con quién compartir ese tipo de información, sin embargo, esa medida no es suficiente.

La privacidad que posee la información que contienen los historiales médicos varía en función de la localización de los expedientes y el propósito por el que se obtuvo dicha información. Y hasta el momento, las leyes que regulan la privacidad de este tipo de bien personal es diferente en función de la situación.

Por otro lado, la aplicación de esta ley ha provocado la desaparición de la confidencialidad de los pacientes debido a la oportunidad de recibir desde la cobertura de un servicio médico hasta la seguridad en el lugar de trabajo.

Tras esta introducción sobre la tecnología sobre la cual estará basada este proyecto y sobre el caso de uso, donde se buscará una solución al problema de la gestión de los historiales médicos, se procederá a hacer un breve análisis como primera toma de contacto con esta tecnología.

Hasta el momento, los historiales médicos están compartidos entre distintas entidades, tanto las pertenecientes al mundo de la salud como las que no.

Los pacientes pueden gestionar los permisos de visionado de sus historiales médicos a quienes se crea conveniente. Sin embargo, esto no quiere decir que los pacientes deciden realmente con quién compartir su información. Actualmente, las entidades o sucesos que hacen que se modifique acceso a dichos historiales son las siguientes:

- Agencias gubernamentales: pueden solicitar el acceso al historial médico con el fin de verificar reclamaciones.
- Divulgación de información: cuando se realiza cualquier tipo de estudio sobre la evaluación de las prácticas médicas la información también se puede ver comprometida.
- Empresas aseguradoras: pueden tener acceso completo al historial médico a la hora de que se desee emitir una póliza o hacer un pago sobre una póliza ya creada.
- La agencia de información médica MIB (Medical Information Bureau). Es un centro de base de datos donde se almacenan los historiales médicos y está compartido entre empresas aseguradoras.

- Empresas: las empresas también pueden poseer dicha información al solicitar al empleado que autoricen la divulgación de su información médica. Esta información puede ser requerida con el fin de realizar una inspección de antecedentes laborales.

Sin embargo, a pesar de las posibles controversias que puede generar este problema, existe la necesidad de poder poseer los expedientes médicos electrónicos. De este modo, se puede almacenar y recuperar todo el historial médico de un paciente. Así se ahorrarían los posibles problemas que se producirían al utilizar archivos médicos de carácter físico, como posibles extravíos.

En la actualidad hay leyes que regulan este problema, sin embargo, las medidas acordadas hasta el momento no son las suficientes para poder proteger algo tan importante como este tipo de información.

1.1. Blockchain

Llegados a este punto, podríamos preguntarnos: ¿Qué es Blockchain? La respuesta a esta pregunta se irá desarrollando durante el estudio, porque como toda tecnología, no se puede dar una respuesta rápida y concreta ya que son múltiples los factores que influyen sobre la misma y, para poder llegar a comprender todo el conjunto, primero, hay que comprender todas las partes involucradas que lo forman.

Sin embargo, como una primera toma de contacto, Blockchain o "cadena de bloques" es una base de datos distribuida que funciona como un "libro" o "ledger" que almacena el registro de cualquier tipo de transacción. Cada transacción está verificada por un consenso de la mayoría de los participantes en el sistema.

Es decir, cada bloque contiene la información que queramos conservar, ya sea una transacción o, como veremos más adelante, una referencia a un smart contract o contrato inteligente.

Todos los bloques, ordenados mediante un orden cronológico, que dan lugar a la cadena tienen un hash del bloque anterior. De este modo, un bloque referencia a su bloque anterior y una vez que una transacción entra en el sistema, ya no puede ser eliminada. Por ello, es prácticamente imposible poder alterar un bloque que ha sido almacenado durante un determinado tiempo en la cadena. Los bloques están continuamente creciendo, cada un determinado periodo de tiempo en función de la red Blockchain escogida se crea un nuevo bloque y así se puede almacenar el registro de la transacción más reciente. Por ejemplo, en la red Bitcoin la escritura de un bloque tarda 10 minutos.

Blockchain posee información desde el bloque génesis (bloque que incluye las primeras transacciones) hasta el último bloque que contiene la última transacción.

En la siguiente imagen se puede observar un fragmento de la estructura de bloques de Blockchain donde, tal y como se ha comentado, se almacenan las transacciones.

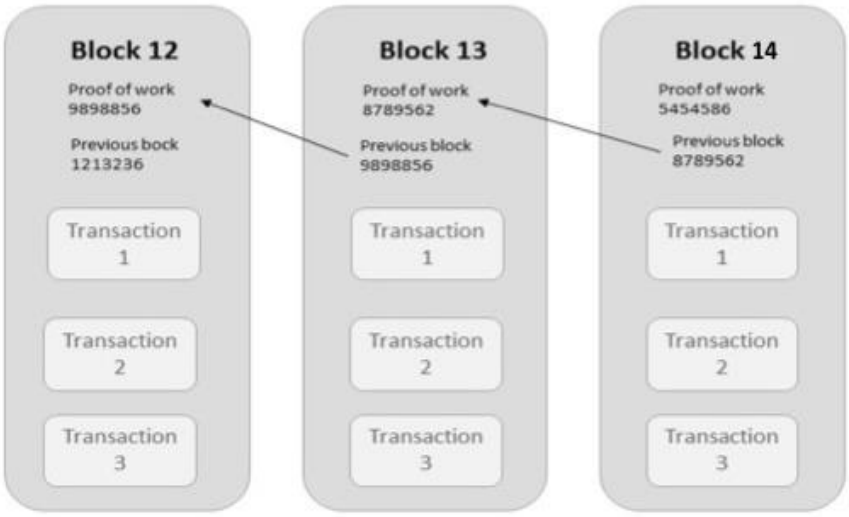


Ilustración 1: Estructura Blockchain

Bitcoin, la moneda digital peer-to-peer, es el ejemplo más popular que usa esta tecnología. Este tipo moneda es algo polémica, pero la tecnología subyacente, Blockchain, ha trabajado hasta el momento sin problemas y posee casos de uso tanto en el sector financiero como en el no financiero.

Blockchain establece un sistema de creación de un consenso distribuido en el mundo digital. Esto permite a las entidades que participan en él saber con seguridad qué evento digital sucedió en un momento determinado gracias a que éste fue registrado. Asimismo, abre la puerta al desarrollo de una economía digital abierta y no centralizada.

Los principales beneficios de Blockchain son mucho más que económicos, se extienden a la política, ciencia, social, salud… Por otro lado, la capacidad tecnológica de Blockchain está siendo aprovechada por grupos específicos para abordar problemas del mundo real.

Por ejemplo, la coordinación, el almacenamiento de registros y la irrevocabilidad de transacciones utilizando esta tecnología son características que podrían ser fundamentales en el progreso de la sociedad. Como una primera aproximación, se podrían ofrecer repositorios de registros públicos para todo tipo de sociedades, incluyendo el registro de documentos, eventos, identidades y bienes. Esto significa que todos los posibles bienes tangibles (como coches y casas) y bienes digitales podrán ser registrados y hechos una transacción en la red Blockchain.

1.1.1. ¿Cómo funciona Blockchain?

[13] Un ejemplo de utilización de Blockchain es el que se podrá ver en la siguiente imagen. Se parte del caso de uso en el que un usuario A quiere hacer una transferencia financiera a un usuario B y de que se está usando una plataforma Blockchain con criptomoneda ya que, como veremos más adelante, hay varios tipos de redes Blockchain. Los pasos que suceden son los siguientes:

1. A decide mandar dinero a B.
2. La transacción es representada en la red como un bloque.
3. Cada nodo que conforma la red recibe el bloque de la transacción
4. Cada nodo valida el bloque y se aprueba mediante un consenso con la transacción es válida.
5. El bloque es añadido a la cadena de bloques.
6. Se realiza la transacción financiera de A a B.

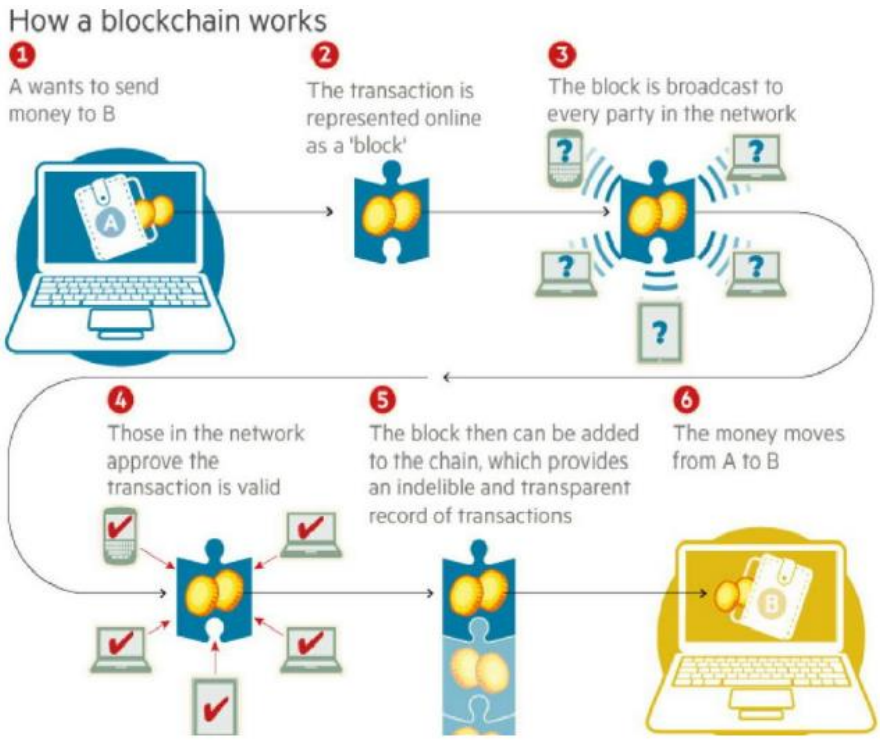


Ilustración 2: Funcionamiento Blockchain [11]

Como podemos observar no existe la entidad de "banco" como intermediario, por lo que la característica principal de Blockchain es que es un sistema "sin confianza", es decir, no hay una entidad centralizada en la que los nodos de la red han de confiar.

Sin embargo, los usuarios pueden confiar en el sistema ya que está almacenado en todo el mundo, es decir, muchos nodos diferentes descentralizados y mantenidos por contadores de mineros.

El Blockchain permite la desintermediación y descentralización de todas las transacciones de cualquier tipo entre todas las partes sobre una base global.

Para entender mejor el concepto de descentralización y las ventajas que esto ofrece, se puede ver la siguiente imagen comparativa:

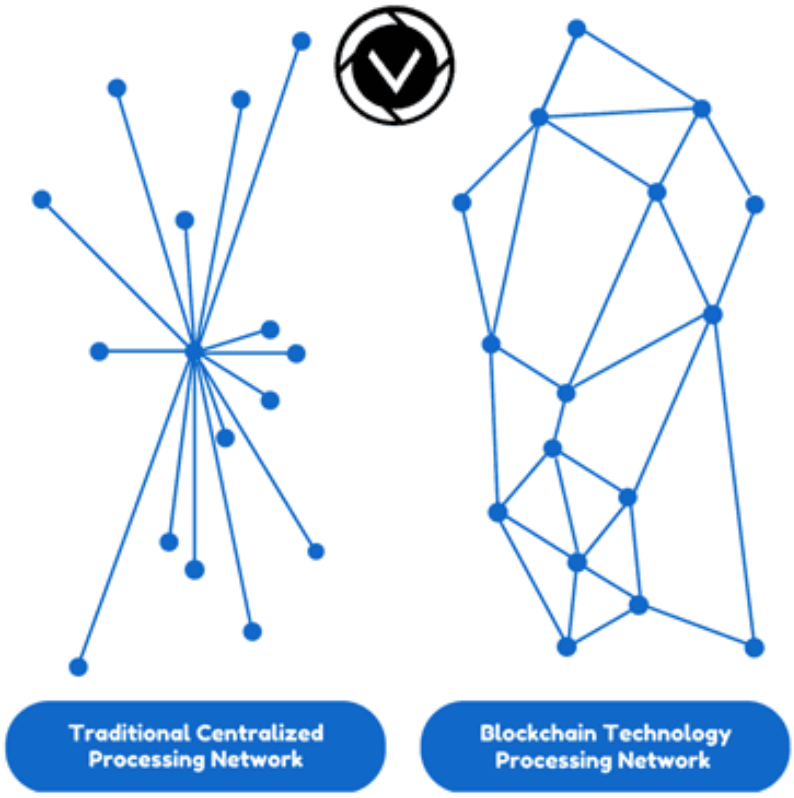


Ilustración 3: Red centralizada VS Red descentralizada

En una red centralizada la propiedad de bienes y la transferencia entre empresas es actualmente ineficiente, lenta, costosa y vulnerable a la manipulación. Cada persona tiene su propio libro donde las discrepancias entre las partes comerciales pueden aumentar el asentamiento tiempos. Y, asimismo, se depende de una tercera parte en la que hay que confiar como, por ejemplo, las Autoridades Certificadoras en el caso del certificado digital.

Por el contrario, en un sistema descentralizado, la tecnología Blockchain puede utilizarse para compartir el libro de registros o "ledger" a través de la red empresarial. La red será privada para las partes interesadas, tan sólo las partes autorizadas están autorizadas a unirse, y se ofrecerá un entorno seguro usando la criptografía para asegurar que los participantes solamente ver lo que están autorizados a ver.

La compartición del ledger será sólida, ya que está replicado y distribuido. Todas las transacciones de contabilidad requerirán de un consenso a través de la red, donde la procedencia de la información es clara y transparente. Las transacciones serán inmutables (invariables) y finales.

De este modo, los bienes y servicios se proporcionan de manera más eficaz, con el potencial de reducir costes en todos los niveles.

1.2. Motivación y Objetivos

La principal motivación por la que se ha realizado el proyecto pertinente es la posibilidad de poder solucionar un problema real mediante el uso de la tecnología Blockchain.

Tal y como se ha comentado anteriormente, el problema de la confidencialidad sobre la información médica en Estados Unidos es algo que no debe ser dejado en el olvido. Mediante el uso de la tecnología Blockchain, y gracias a su infraestructura y a la posibilidad de la realización de contratos inteligentes, de los cuales se hablará más adelante, se podría plantear una posible solución.

Por otro lado, el propósito de este estudio es dar a conocer más en profundidad este tipo de tecnología. Hasta ahora, y tal y como se ha comentado, Blockchain es una tecnología que está bastante inmadura, que se le vaticina un gran futuro y que no es muy conocida hasta la fecha.

Asimismo, como motivación personal a la hora de realizar el proyecto es la autodidáctica de esta tecnología y de todas las tecnologías subyacentes necesarias para poder llegar a comprenderla. Hasta ahora todas las tecnologías aprendidas en la universidad conforman una gran base sobre la cual podré crecer como profesional y así poder ampliar mis conocimientos.

Estas tecnologías utilizadas para desarrollar el proyecto son bastante comunes y “modernas” en el mundo profesional y entre ellas se encuentran: NodeJS, AngularJS, Docker, Git, etc.

En cuanto los objetivos que se plantean, y se esperan cumplir durante el desarrollo del presente proyecto, son los siguientes:

- Investigación sobre la tecnología Blockchain.
- Estudio y comparativa de las principales plataformas en la actualidad.
- Despliegue de una red Blockchain
- Implementación de la solución que pueda plantear el remedio del problema de la gestión de los historiales médicos.

1.3. Estructura del documento

Como iremos viendo a lo largo del proyecto, el documento está claramente estructurado en los siguientes apartados:

Introducción

Apartado en el que nos encontramos en este instante. Se trata de una sección inicial en la que se presenta el proyecto, se hace una introducción al problema a representar y se presentan las secciones en las que se dividirá el presente documento.

Estado del arte

En esta sección se realizará un análisis en profundidad sobre la tecnología que se utilizará a lo largo del proyecto. Este estudio a desarrollar se hará haciendo hincapié en el funcionamiento de Blockchain y su estado actual, así como las distintas plataformas que hay desarrolladas o que se encuentran en proceso de desarrollo.

Algoritmos de consenso de las redes Blockchain:

Visto el funcionamiento de Blockchain, se analizarán los distintos algoritmos de consenso de las redes estudiadas y se verán cuáles son las diferencias más significativas entre los mismos.

Análisis

Sección en la que se realizará un análisis de todo aquello que se espera que realice el sistema a desarrollar. El análisis se realizará haciendo una definición de los casos de uso del sistema y una especificación de requisitos.

Diseño de la aplicación:

En este apartado se determinará cuál será el funcionamiento del sistema, haciendo el diseño de las partes que formarán parte del mismo sin tener en cuenta cual será la implementación del mismo.
El diseño estará compuesto por el estudio del entorno operacional y por los componentes que aparecen en el mismo.

Implementación y pruebas

Apartado en el que se realizará una explicación sobre la implementación llevada a cabo para poder desarrollar el comportamiento del sistema y sus funcionalidades requeridas.
Posteriormente se hará una definición de los casos de prueba que se deberán llevar acabo para hacer las comprobaciones necesarias.

Marco regulador y entorno socioeconómico

Apartado en el que se analiza la existencia de regulaciones y normativas legales que puedan ser aplicables al proyecto.

Planificación y presupuesto

En esta sección se presentará la planificación realizada a la hora de seguir el proyecto, mediante diagramas de Gantt. Y, por otro lado, el presupuesto que costará la realización del proyecto.

Conclusiones y trabajo futuro

Es el fin del proyecto, se presentarán las conclusiones sobre el mismo y se harán recomendaciones sobre partes a mejorar del mismo.

Glosario: definiciones y abreviaturas

Referencias

2. Estado del arte

Para comprender mejor la tecnología Blockchain deberemos remontarnos a sus cercanos orígenes. Blockchain nació como la tecnología subyacente a la plataforma Bitcoin, por ello para conocer su historia, habrá primero que conocer la historia de dicha plataforma.

Tras hacer un estudio para conocer los inicios que hicieron que las redes Blockchain aparecieran hasta que se han hecho presentes en el campo de la informática, se realizará una clasificación de los distintos tipos de plataformas que puede albergar este tipo de tecnología.

Por otro lado, durante este apartado se realizará un análisis del estado del arte de las plataformas Blockchain. Por ello, se realizará un estudio sobre cuatro de las plataformas que serás las candidatas para desarrollar una aplicación descentralizada, utilizando como caso de uso la gestión de los historiales médicos electrónicos. Asimismo, se realizará un análisis sobre los tipos de historiales médicos electrónicos más conocidos hasta la fecha.

2.1. Historia de Bitcoin

En el año 2008 [14], un individuo o grupo escribiendo bajo el nombre de Satoshi Nakamoto, un nombre que pertenece de forma anónima hasta la fecha, publicó un documento titulado "Bitcoin: A Peer-To-Peer efectivo sistema electrónico". Este artículo describe una versión de peer-to-peer del dinero electrónico que permitiría que los pagos en línea enviar directamente de una parte a otra sin pasar por una institución financiera.

Bitcoin es la primera realización de este concepto. Ahora la palabra “criptomonedas” es la etiqueta que se utiliza para describir todas las redes y medios de cambio que utiliza la criptografía para transacciones seguras, frente a los sistemas donde las transacciones se canalizan a través de una entidad de confianza centralizada.

Meses más tarde a la publicación del documento, apareció una plataforma de código abierto poniendo en práctica el protocolo sobre el cual se basaba el artículo. Dicha plataforma comenzó con un génesis (primer nodo de la red Blockchain) con 50 monedas.

La popularidad de Bitcoin nunca ha dejado de aumentar desde entonces. La tecnología de Blockchain subyacente encuentra ahora la nueva variedad de aplicaciones más allá de finanzas.

Ilustración 4: Timeline de Bitcoin

2.2. Clasificación de las redes Blockchain

En el último año el concepto de "Blockchain privado" se ha convertido en muy popular en el debate más amplio de la tecnología de Blockchain.

En esencia, en lugar de tener una red totalmente pública y sin control controlada por el valor de las criptomonedas, como veremos más adelante, también es posible crear un sistema donde permisos de acceso son más bien controlados, con derecho a modificar o incluso leer el estado de Blockchain restringido a unos cuantos usuarios.

Siguiendo estos principios, una de las clasificaciones que realiza Ethereum (una de las empresas que implementa una plataforma Blockchain) es la siguiente:

- **Blockchain pública:** es una red Blockchain en la que cualquier persona puede leer, puede enviar transacciones y esperar si han sido incluidas en la cadena, y puede participar en el proceso de consenso, proceso que determina qué bloques son incluidos en la cadena.
- **Blockchain de consorcio:** es una red Blockchain que, en el proceso de consenso, donde las operaciones son de escritura en la red, participan únicamente nodos preseleccionados. Por otro lado, los procesos de lectura pueden ser público o de acceso restringido. Estas redes pueden ser parcialmente descentralizadas.
- **Blockchain privada:** es una red Blockchain en la que los procesos de escritura están controlados por permisos y es una organización quien cede esos permisos. Sin embargo, los procesos de lectura pueden ser públicos o también restringido.

[15] Hoy en día, las redes Blockchain también pueden ser clasificadas mediante generaciones, teniendo en cuenta las funcionalidades que han ido siendo añadidas en función de su desarrollo y crecimiento. Esta clasificación puede verse de la siguiente manera:

- **Primera generación**: surgió con la aparición de la primera red Blockchain, Bitcoin. La idea principal de las redes de esta generación es en la creación de un sistema distribuido y

descentralizado que se utilice como un almacén de registros transparente a todos los participantes en la red.

- **Segunda generación**: toma como base la generación anterior e introduce un nuevo elemento en la red con el que interactuar, las criptomonedas. Este tipo de redes están totalmente orientadas al mundo financiero, en concreto a la funcionalidad de poder hacer transacciones monetarias entre dos entidades.
- **Tercera generación**: en este tipo de redes aparecen los smart contracts o contratos inteligentes, la base para la creación de aplicaciones descentralizadas. Este tipo de contratos son autoejecutables, están almacenados en la red Blockchain (por lo que nadie puede interferir en ellos) y, por lo tanto, todo el mundo puede confiar.

Los contratos inteligentes son piezas de código, que se despliegan en una red Blockchain, capaces de verificar sus propias condiciones utilizando datos y auto-ejecutables. Su principal característica es que son a prueba de manipulaciones.

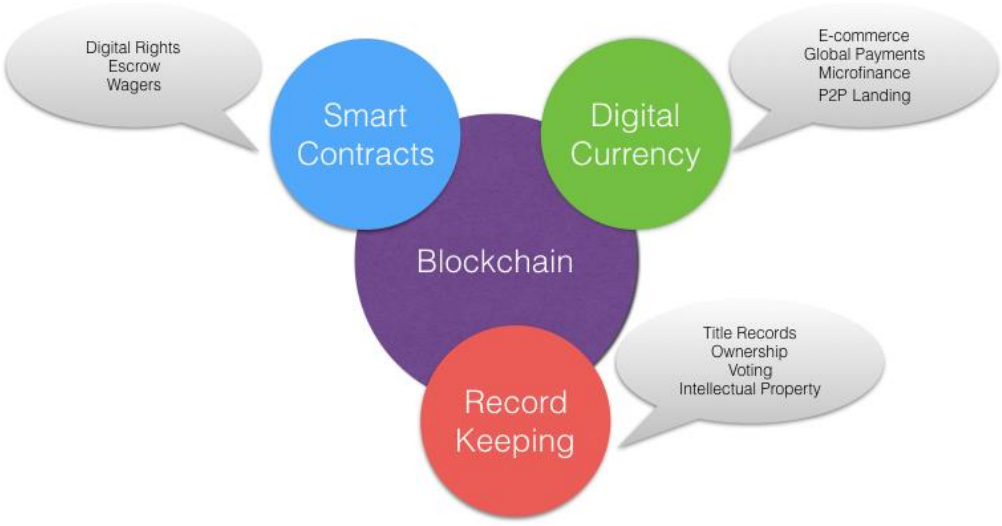


Ilustración 5: Clasificación de redes Blockchain

2.3. Análisis de redes Blockchain

Con motivo de proseguir con el estudio del arte actual de las redes Blockchain, se procederá a hacer un análisis de las redes más significativas a la hora de montar una red de Blockchain propia y poder desarrollar una aplicación descentralizada sobre ellas.

Montar una red Blockchain propia en ocasiones está limitado. Esta limitación viene dada en función de la licencia del código y la complejidad que supone poder crearla.

Las redes Blockchain que han sido escogidas para la realización del estudio son Ethereum, Eris, Ripple y Hyperledger.

2.3.1. Ethereum

Ilustración 6: Logo de Ethereum

La plataforma Ethereum [16] está implementada por un equipo de desarrolladores de EthDev en nombre de Ethereum Foundation.

Se trata de una plataforma de código abierto cuya intención es permitir desarrollar aplicaciones descentralizadas (Dapps) utilizando la tecnología blockchain. Además, posee su propia criptomoneda (Ether) que se utiliza para recompensar a los nodos de la red como pago por el coste computacional realizado. Del mismo modo, permite la transferencia de Ethers entre diferentes cuentas de la red.

Ethereum no es como la mayoría de las redes de criptomonedas existentes ya que, además de reflejar las transacciones monetarias, se utiliza para desplegar y ejecutar contratos inteligentes sin posibilidad de que haya algún tipo de censura, fraude, intervención de terceras partes ni caída del sistema.

Las características más importantes de la plataforma son:

- Libro contable con el estado de la red y con la lista de las transacciones realizadas.
- Moneda propia (Ether)
- Ejecución de smart contracts dentro de la Ethereum Virtual Machine(EVM)

Según la clasificación realizada anteriormente, Ethereum podría considerarse como una red Blockchain de segunda generación.

2.3.1.1. Casos de uso

De forma general, se pueden identificar los siguientes casos de uso donde la plataforma Ethereum podría utilizarse según la siguiente clasificación:

- Organizativo: posesión de bienes, acuerdos de accionistas, mercados de predicción, sistemas de votación o registros de dominio.

- Finanzas de igual a igual: crowdfunding, productos derivados, coberturas y seguros.
- Aplicaciones descentralizadas enfocadas al consumidor: garantías, tienda de activos personales, propiedades inteligentes, intercambios financieros, cuentas de ahorro, testamentos y propiedad intelectual.

De forma concreta, hay múltiples proyectos en vuelo que utilizan Ethereum para su funcionamiento:

- Airlock: sistema de cerraduras descentralizado.
- Augur: sistema de predicción de mercados.
- CubeSpawn: sistema de fabricación flexible que ofrece automatización de grandes fábricas a tiendas pequeñas.
- Gnosis: sistema de predicción.
- BoardRoom: sistema de gobernanza.
- Colony: sistema para crear empresas.
- Provenance: sistema para seguir el rastro de un producto de origen a destino.
- Slockit: sistema para vender, alquilar o compartir cualquier cosa sin necesidad de intermediarios.
- Digix: sistema de activos inteligente.
- WeiFund: sistema de crowdfunding.
- Maker: sistema de crédito autónomo.
- HitFin: sistema para comerciar con productos derivados.

2.3.1.2. Empresas relacionadas

La Ethereum Foundation no tiene ninguna relación pública con otras empresas a nivel de colaboración, ni de partnership o clientes relevantes en el momento actual.

La única relación con empresas es con Microsoft (asociada con Consensys) en la medida de que la plataforma de Ethereum es incluida en la plataforma Azure Blockchain as a Service, y con Wanxiang Group Corporation a través de Blockchain Labs.

Esta última empresa sin ánimo de lucro realizó una compra de medio millón de dólares en Ethers para apoyar dicha tecnología, y asimismo creó un fondo de capital de riesgo por valor de 50 millones de dólares.

Por otro lado, Slock.it ha anunciado la creación de un dispositivo IoT que llevará Ethereum integrado.

2.3.1.3. Arquitectura general de la plataforma

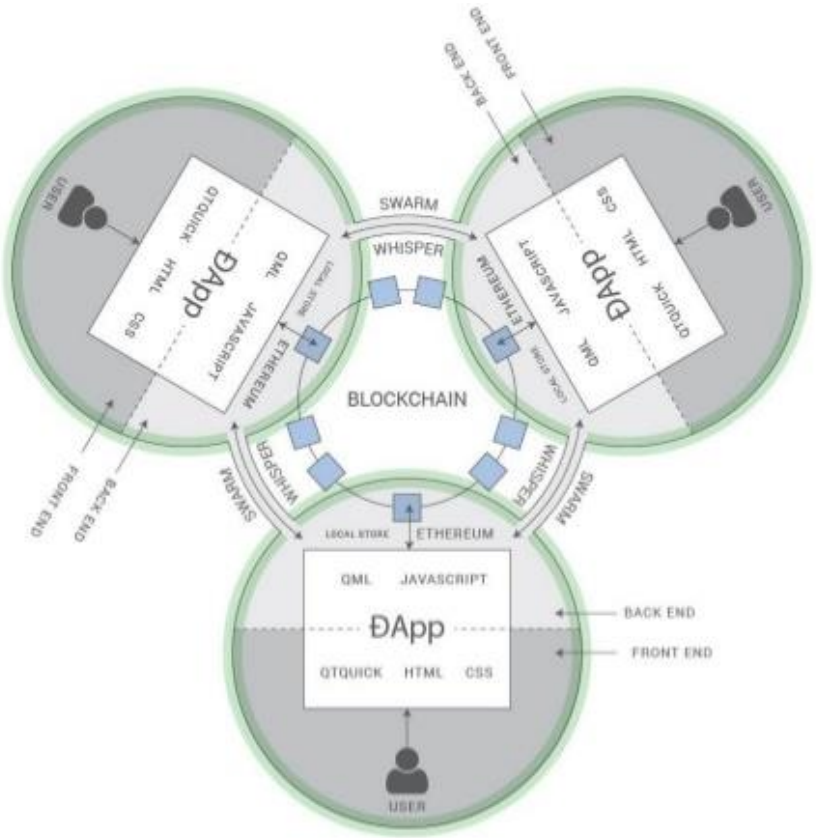


Ilustración 7: Arquitectura general de Ethereum. Fuente: Ethereum] [16]

La plataforma completa de Ethereum está formada por:

- **Swarm:** es el componente encargado del almacenamiento y transmisión de ficheros.
- **Whisper:** es el componente encargado de la mensajería entre los nodos.
- **Ethereum:** es el componente que se encarga de la ejecución de las transacciones y de los contratos gestionando el blockchain.

2.3.1.4. Implementaciones y APIs

Existen tres implementaciones oficiales de la plataforma realizadas en los siguientes lenguajes:

- C++
- Go
- Python

Y, por otro lado, a la hora de interactuar la plataforma, existen dos APIs para la comunicación con el core de Ethereum:

- Web3 JavaScript API: es el JavaScript SDK a utilizar cuando se quiere interactuar con la API de los nodos.

- JSON RPC API: es la interfaz de bajo nivel JSON RPC 2.0 para interactuar con un nodo. Esta API es utilizada por Web3 JavaScript API.
- IPC: es la API utilizada para realizar llamadas vía socket.

2.3.2. Eris

Ilustración 8: Logo de Eris

La plataforma de Eris [17] está desarrollada por parte de la compañía Monax Industries, compañía que comenzó siendo una start-up bajo el nombre de Eris Industries. Esta compañía es la principal valedora de la generación de esta plataforma y su mantenimiento. La plataforma ERIS es una plataforma de Blockchain que no está anclada en el mundo bancario y que propone un modelo más de propósito general, siendo sus principales puntos los siguientes:

- No dispone de criptomoneda propia.
- Soporta conectores para gestionar Blockchain basados en Ethereum o Bitcoin, entre otros.

Según la clasificación realizada anteriormente, Eris podría ser considerada como una red Blockchain de tercera generación.

ERIS es una plataforma que se basa en la gestión de diferentes conceptos y la relación entre los mismos, como son:

- Servicios (services): Se trata de un conjunto de servicios que se exponen como elementos auxiliares para la realización de aplicaciones basadas en smart contracts.
- Chains (chains): Son las propias cadenas donde se desplegará y ejecutarán los diferentes Smart Contracts que se desplieguen.
- Contratos (contracts): Son los propios smart contracts que contienen la lógica de ejecución que queremos aplicar.
- Acciones (actions).

Eris se basa en Docker como sistema central para gestionar el ciclo de vida de sus diferentes componentes, haciendo que cada servicio y cadena sea un contenedor de Docker que tiene una configuración centralizada, tal y como se puede ver en la siguiente imagen:

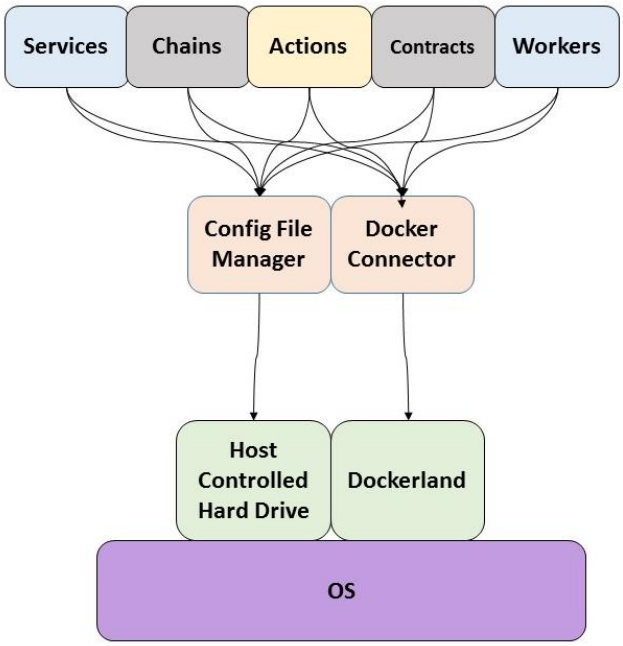


Ilustración 9 Base tecnológica de Eris [17]

2.3.2.1. Casos de uso

Monax Industries es una empresa orientada al propósito general y no principalmente centrada en banca por alguna de sus principales características, tal y como se ha comentado anteriormente.

Sin embargo, pese a eso, los casos de uso principales que remarcan en su web son referentes al sector bancario:

- Automatización de procesos de negocio back-office.
- Puesta en seguridad y transferencia del riesgo.
- Toma empresarial de decisiones y coordinación como resoluciones de bondholder o accionista.

Actualmente y tal y como se comentaba en el punto anterior, no existen referencias de proyectos públicos en su web con ningún cliente.

2.3.2.2. Empresas relacionadas

Monax Industries no tiene ninguna relación pública con otras empresas a nivel de colaboración, ni de partnership o clientes relevantes en el momento actual. La única relación con empresas es con Microsoft en la medida de que la plataforma Eris es incluida en la plataforma Azure Blockchain as a Service que la empresa de Redmond está montando desde el diciembre de 2015.

2.3.2.3. Arquitectura general

ERIS es una plataforma que se compone de los siguientes componentes lógicos a nivel estructural que son los siguientes:

- eris:cli
- eris:db
- eris:pm
- eris:legal

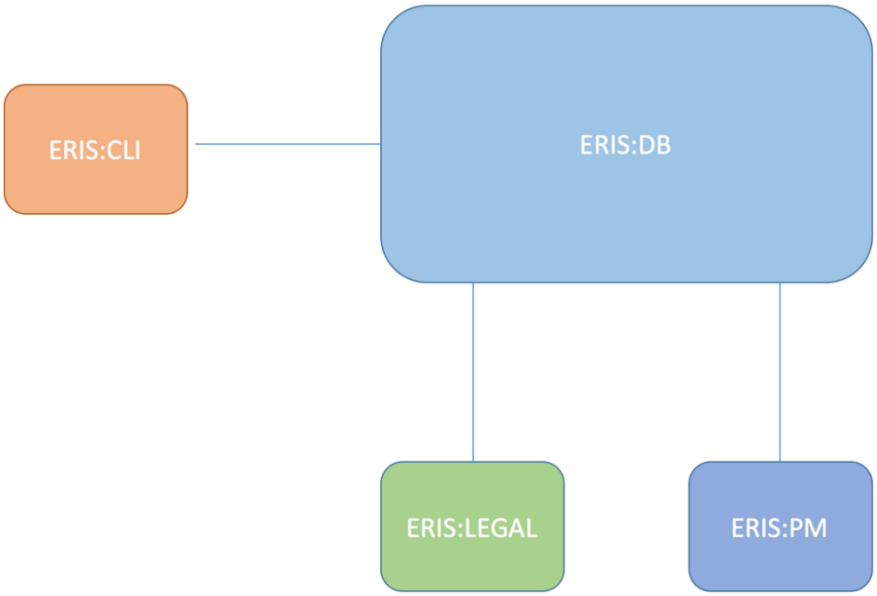


Ilustración 10: Arquitectura general de ERIS

Eris:cli

Se trata de un componente que sirve para facilitar la gestión de las aplicaciones basadas en smart contracts, y que se asienta sobre las dos bases técnicas de la propia plataforma que son Go y Docker.

Tal y como reconocen directamente los desarrolladores en su propio de repositorio, el cliente no deja de ser un wrapper sobre el API de Docker para facilitar la gestión de los diferentes componentes y conceptos asociados con las aplicaciones smart contracts dentro del mundo de Eris.

Eris:db

Es el servidor central de la plataforma Eris y a su vez se basa en una plataforma Blockchain conocido como Tendermint, que se trata de una plataforma abierta de blockchain que permite la ejecución de otras plataformas como es el caso del propio Eris, Ethereum o Bitcoin.

Se trata de un motor que cumple la especificación EVM (Ethereum Virtual Machine), y esto que hace que permita ejecuta cualquier smart contract que haya sido generado bien por el propio compilador de eris (eris:compiler) o bien cualquier otro compilador de Ethereum.

El hecho de basarse en Tendermint le proporciona una gran abstracción y compatibilidad a Eris con otras plataformas, siendo posible ejecutar conectores como Ethereum o Bitcoin.

Eris:pm

Se trata del gestor de contratos, similar y con funciones equivalentes a un gestor de paquetes propio de UNIX (yum, apt-get, pacman...) o un gestor de dependencias de los lenguajes de programación (Maven, Gradle).

Es el encargado de desplegar y testear los diferentes contratos, partiendo de un fichero de definición, conocido como fichero EPM.

Eris:legal

Es el nombre del módulo bajo el que se encuentra el módulo Eris Legal Markdown, el cual tiene la responsabilidad de enlazar contratos reales que pueden ser escritos bien usando Markdown u otras alternativas con smart contracts que corren dentro de los sistemas de blockchain distribuidos.

2.3.2.4. APIs

El propio servidor expone las siguientes API's:

- API's basadas en HTTP:
  - JSON-RPC
  - REST-lik.

- API's basada en WebSocket.

Adicionalmente a las propias API's existen un conjunto de librerías clientes que facilitan su integración con otro tipo de aplicaciones, como pueden ser la siguiente:

- eris-db.js: Librería js pensada para la integración con aplicaciones Node.

2.3.3. Ripple

Ilustración 11: Logo de Ripple

Ripple [18] es una red Blockchain de código abierto, destinada al entorno bancario. Permite realizar transacciones entre monedas diferentes incluida su propia criptomoneda. También permite utilizar una red pública o crear y utilizar una red privada.

Ripple también ofrece una serie de productos cliente de su red, aunque dichos productos no serán exhaustivamente analizados.

Las principales características de Ripple son:

- Utiliza un Libro Contable (ledger) que contiene el estado de la red en vez de una cadena con todas las transacciones, como Bitcoin
- Utiliza un sistema de validación de consenso para crear cada nueva versión "cerrada" del libro de cuentas.
- Tiene moneda propia
- Da soporte a monedas reales emitidas por alguien, asume que esas monedas están respaldadas por monedas reales en el banco de ese alguien.

2.3.3.1. Casos de uso

Ripple es una red que está totalmente volcada con la banca, por ello sus casos de uso son los siguientes:

- Pagos
- Pagos con cambio de divisa buscando las conversiones más ventajosas dentro de la red.

Se está estudiando la inclusión de contratos inteligentes, lo cual daría cabida a la comercialización de otros activos que no sean monedas, como podría ser Oro, Productos derivados, etc.

2.3.3.2. Empresas relacionadas

En el caso de Ripple, al ser una red más madura que las anteriores, son múltiples las empresas que tienen relación con el producto, tanto aquellas que tienen partnership como aquellas que estén participadas por otras empresas.

Inversores:

- Santander InnoVentures
- Google Ventures
- Seagate
- CME Group
- IDG Capital Partners
- Andreessen Horowitz

Partners

- Accenture
- CGI
- Earhport
- IntellectEU
- CBW bank
- Cross River Bank
- Fidor Bank

2.3.3.3. Arquitectura

Los componentes principales se engloban en tres tipos:

- Red Ripple, componentes que componen la red Ripple, construyen y mantienen el libro contable, realizan las validaciones y ofrecen los servicios que consumen todas las demás aplicaciones.
- Aplicaciones para clientes. Aplicaciones que permiten a los usuarios finales de la red Ripple realizar y recibir pagos.
- Aplicaciones Gateway (Ripple Stream y marketMaker). Aplicaciones para proveedores de internet, para mantener el libro de órdenes, e inyectar divisas en la red. Estas aplicaciones están orientadas a reproducir en el mundo real las transacciones de divisas que se producen en la red Ripple.

Ripple desarrolla tanto la red como una serie de aplicaciones subsidiares:

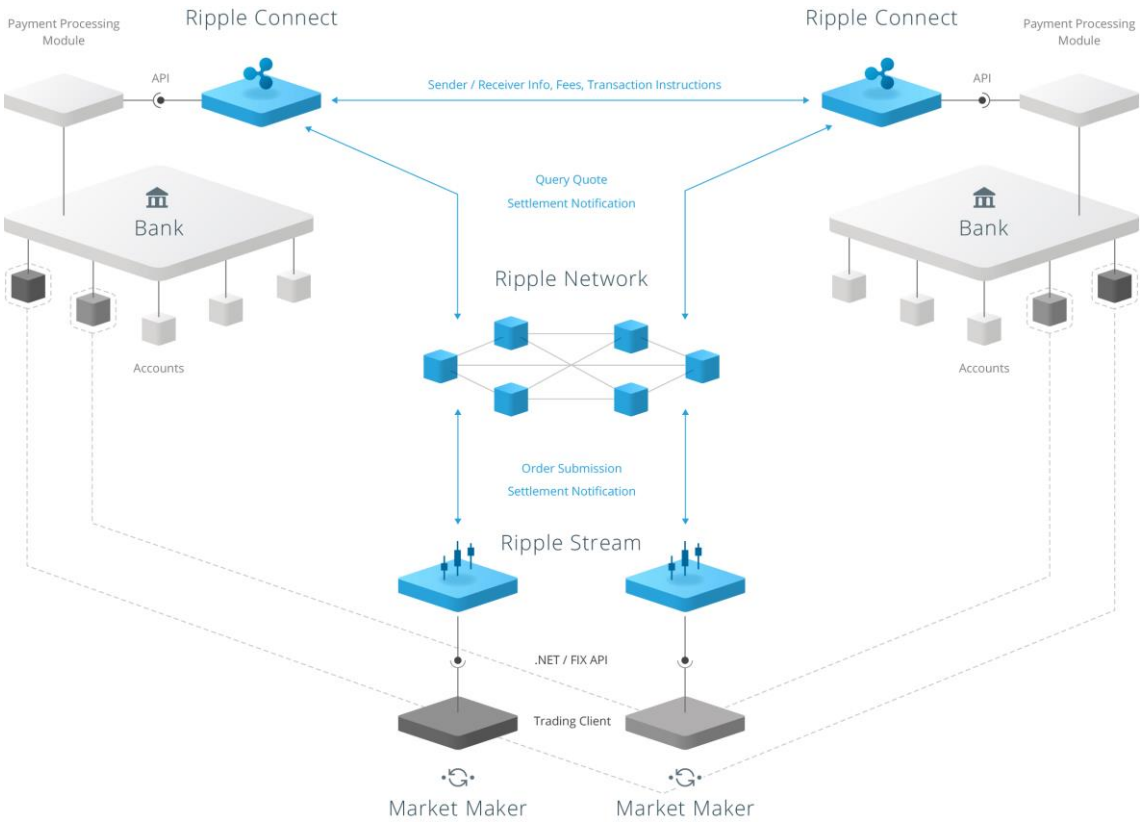


Ilustración 12 Arquitectura global de la red Ripple pública. Fuente: Ripple [18]

Dentro de la red Ripple, la arquitectura es como sigue:

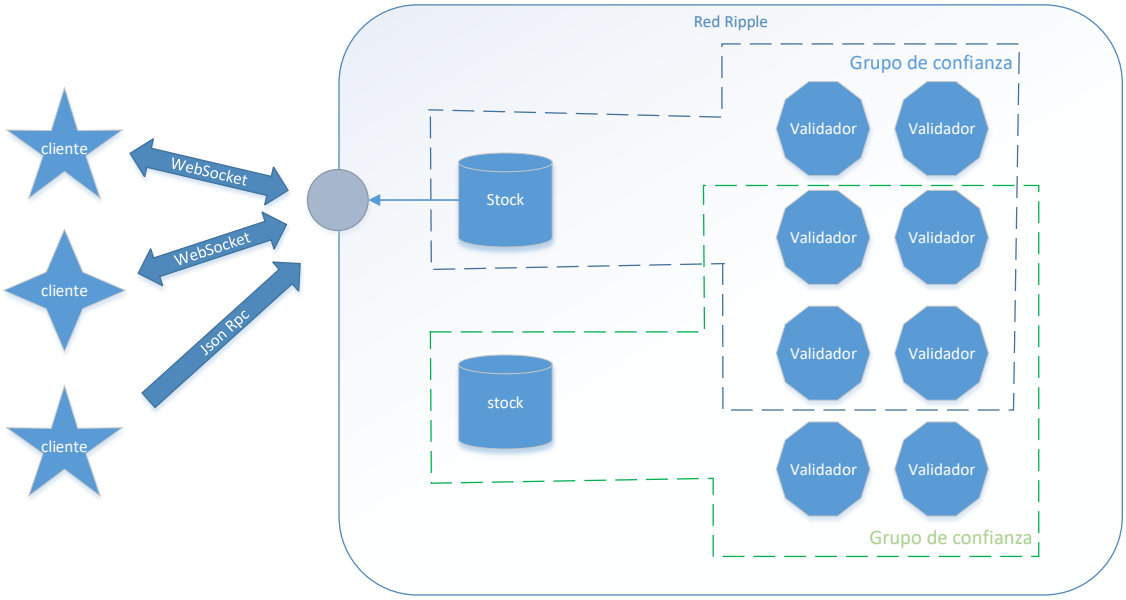


Ilustración 13 Arquitectura de la red de Ripple privada [18]

En una red Ripple existen una serie de nodos validadores que se encargan de validar mediante consenso las transacciones que les remiten. Una vez validadas se recogerán en el próximo libro contable cerrado.

Los servidores de tipo Stock cumplen varias funciones

- Guardan una copia del libro contable
- Definen subredes de confianza. Cada nodo de tipo Stock tiene que seleccionar un subconjunto de la red de validadores en los que confía. Desde un punto global, estos grupos deben de tener solapamientos de al menos dos nodos entre subredes o las redes podrían derivar en libros contables no-unificados. Las redes de confianza son mecanismo adicional de seguridad y una manera de conseguir velocidad
- Son los puntos de acceso públicos de la red. Un stock Server puede ser simplemente un repositorio (completo) pero puede ser un punto de la red donde se exponen las operaciones a los clientes de la red. En este caso, no necesitan tener una copia completa del histórico del libro contable.
- Inyectan las operaciones a sus validadores
- Se comunican entre sí y con los validadores para replicar los libros cerrados.

Tanto los validadores como los stocks servers se comunican con un protocolo llamado Ripple Payment Protocol (hablan Ripple) que es un protocolo distribuido, similar en su naturaleza al protocolo P2P de bittorrent.

Los clientes consumen los servicios expuestos por los stock-servers via json-rpc o via websocket. Siendo el api websocket la más completa de las dos.

Estos clientes son los mencionados en la arquitectura general, y pese a tener diferentes funcionalidades, usan el mismo api.

2.3.3.4. API

Para realizar una interacción con la red de Ripple la API existente es la siguiente:

- Ripple-REST: Interfaz RESTful entre la red y una aplicación Node.
- Ripple-lib: Impmentación para poder acceder a la API WebSocket de la red.
- rippledWebSocket API: API asíncrona construida mediante el protocolo WebSocket
- rippled JSON-RPC API: API síncrona construida siguiendo el convenio JSON-RPC.

2.3.4. Hyperledger

Ilustración 14: Logo del Proyecto Hyperledger

Hyperledger[19] es un proyecto de colaboración de Linux Foundation cuyo propósito es avanzar en la tecnología Blockchain identificando y abordando rasgos importantes para distintos sectores, consiguiendo así un estándar abierto de las transacciones comerciales que se llevan a cabo globalmente.

El ideal básico de esta plataforma es que el estándar abierto de Blockchain debe de estar basado en la modularidad. Por ello, la plataforma, debe de estar construida de forma en que las versiones de los diversos componentes de la cadena de bloque puedan ser intercambiados fuera de la voluntad de los desarrolladores. Es decir, mientras que algunos casos de uso necesitan un rápido consenso entre algoritmos confiables, otros pueden requerir menos velocidad, pero mucha más confianza. Por ejemplo, los smart contracts y el almacenamiento en base de datos debe ser del tipo “plug and play”.

Como objetivo de la plataforma es que Hyperledger contenga una API completa y fácil de usar y numerosos módulos principales que permitan el desarrollo y la interoperabilidad, pudiendo así abarcar numerosos casos de uso.

En resumen, Hyperledger debe ser una plataforma fácil de usar, útil y robusta para todos aquellos interesados en desarrollar software blockchain que use tanto código como núcleos.

2.3.4.1. Casos de uso

Actualmente hay del orden de 30 casos de uso aproximadamente entre los que destacan los siguientes:

Depósito de activos financieros

Los activos como títulos financieros deben tener la capacidad de ser desmaterializados en una red Blockchain, para que todos los stakeholders de un determinado activo puedan tener un acceso directo al mismo y obtener su información sin tener que pasar por capas intermediarias.

Asimismo, las transacciones deben de ser capaces de realizarse en un tiempo acordado entre dos partes, incluyendo si es necesario la capacidad de que se realice en tiempo real.

Ejemplo: El creador debería ser capaz de crear un activo de tal forma que el historial y el saldo actual del activo no esté disponible para otras terceras partes poseedoras de dicho activo.

Acción corporativa

Una empresa A anuncia un evento corporativo, sin embargo, la compañía necesita asegura que los detalles completos de éste sean enviados en tiempo real a los accionistas sin tener en cuenta cuántos intermediarios están involucrados en el proceso. Una vez que un accionista ha tomado una decisión, ésta será procesada en tiempo real.

Cadena de suministro

La plataforma blockchain debe proporcionar un medio para permitir a cada participante en una cadena de suministro introducir y rastrear el abastecimiento de materias primas, el registro de fabricación de piezas de telemetría, la procedencia de mercancías y el estado de su envío.

Gestión de datos maestros

Este tipo de datos, que contienen información comercial generalmente no transaccional son la clave y el componente principal de muchas industrias.

El objetivo es tener una versión confiable de estos datos donde las partes autorizadas pueden realizar cambios y designar validadores que aceptan dichos cambios, de este modo se resolverá la calidad de los datos y la integridad.

Economía compartida y el Internet de las cosas

La economía compartida generará nuevos tipos de ingresos en muchas industrias, incluyendo ciudades inteligentes, automoción, construcción, fitness, salud...etc.

Durante una transacción, las organizaciones no confían entre ellas. Implementándolo correctamente, una contabilidad blockchain distribuida podría resolver muchos de estos problemas de confianza.

2.3.4.2. Empresas relacionadas

Hay numerosas empresas relacionadas con el proyecto Hyperledger, haciendo diferencia entre miembros premier y general. Entre los primeros encontramos: Hitachi, IBM, Intel, Fujitsu, Accenture, CME Group, Deutsche Börse Group, J.P. Morgan, Digital Assert, Depository Trust & Clearing Corporation y R3, como miembros principales.

Como miembros generales encontramos Cisco, Blockchain, Redhat, VmWare…, entre otros.

2.3.4.3. Arquitectura

La arquitectura está compuesta por tres componentes principales:

- Servicios de identidad que gestiona las identidades de las entidades, de los participantes y los objetos transaccionales como los contratos inteligentes y activos.
- Blockchain que gestiona las transacciones a través de un protocolo de comunicación peer-to-peer. Las estructuras de datos están optimizadas para proporcionar esquemas eficientes para mantener el estado del mundo replicado en muchos de los participantes. Los algoritmos diferentes de consenso garantizan fuerte consistencia.
- Contratos inteligentes son una forma segura y ligera de aislar la ejecución de los contratos inteligentes sobre la validación de nodos.

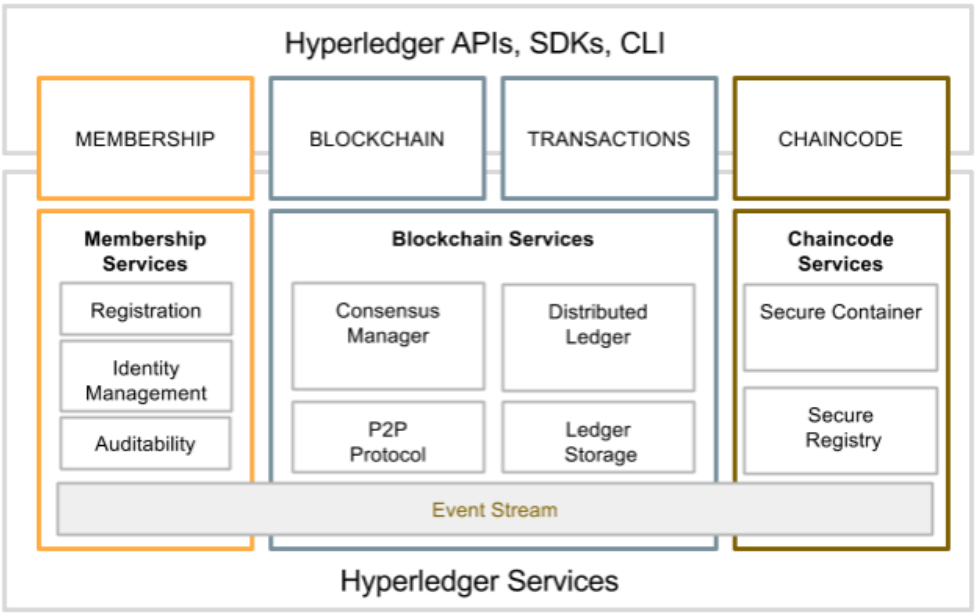


Ilustración 15: Arquitectura Hyperledger. Fuente: Hyperledger Project [19]

Servicio de identidad

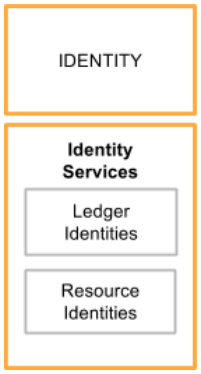


Ilustración 16: Servicio de identidad de Hyperledger

El servicio de identidad es un requisito generalizado para el protocolo de Hyperledger. El servicio controla las identidades de todos los participantes: organizaciones, validadores, transacciones, los objetos incluidos en las transacciones (activos y contratos inteligentes), y compontes del sistema de como redes, servidores, entornos de ejecución.

También incluye una representación de los diversos papeles que juegan en la contabilidad.

Los validadores, durante la configuración de red, pueden determinar el nivel de permiso que se requiere para realizar transacciones.

La configuración de la red también define la red como permisiva, lo que permite facilidad de acceso y apoyo para la adopción rápida y alta, o restrictiva para un ambiente más controlado.

Blockchain

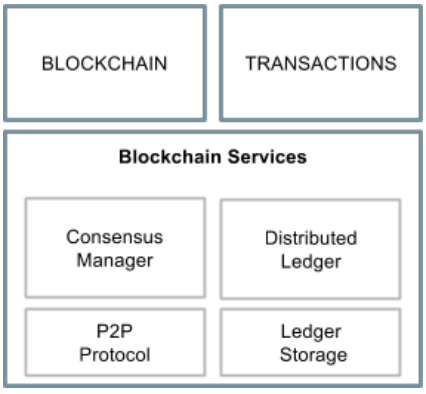


Ilustración 17: Servicio de Blockchain de Hyperledger

Consiste en tres componentes principales: el protocolo *peer-to-peer"",* el "distributed ledger" (la contabilidad distribuida), y el *"consensus manager"* (gestor de consenso).

- El protocolo P2P ofrece muchas capacidades incluyendo streaming bidireccional, control de flujo, y multiplexación de peticiones a través de una única conexión. Lo que es más importante, funciona con la infraestructura existente de Internet, incluyendo firewalls, proxies y seguridad.
- El *"distributed ledger"* maneja la red Blockchain y el estado mundial tratando y validando transacciones, actualizando y manteniendo el estado de objetos del libro mayor. Utiliza un almacén de datos para que persistan los datos, y construye una estructura interna de datos.
- El *"consensus manager"* es una abstracción que define la interfaz entre el algoritmo de consenso y los otros componentes de la Hyperledger. Recibe las transacciones y, dependiendo del algoritmo, decide cómo organizar y cuando para ejecutar las transacciones. La ejecución exitosa de las transacciones se traduce en cambios en la contabilidad. El algoritmo de consenso utilizado es PBFT (Practical Byzantine Faul Tolerance).

Contratos inteligentes (Chaincodes)

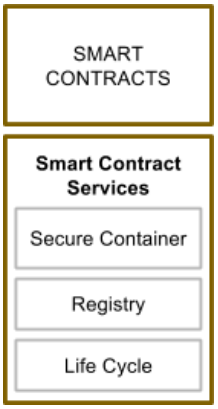


Ilustración 18: Contratos inteligentes en Hyperledger

Incluye un entorno de ejecución seguro, un registro de contratos inteligente y la gestión del ciclo de vida.

Estos contratos inteligentes (chaincode) tienen distintas políticas de acceso.

- Pública, utilizados por las transacciones públicas donde cualquier miembro de la red lo puede invocar.
- Confidencial, desplegado por transacciones confidenciales que solo pueden ser usados por miembros validadores.
- Acceso controlado, desplegado por transacciones confidenciales que utilizan los token de la transacción para identificar a los invocadores.

2.3.4.4. API

Actualmente la API de Hyperledger se encuentra en desarrollo, por lo que no todas sus funcionalidades están implementadas.

La API disponible utiliza el protocolo RPC, aunque está el proceso de la implementación de una librería para Node y Java.

Por lo tanto, en vista al breve análisis que hemos realizado, a la hora de realizar una aplicación basada en redes Blockchain hay que tener en cuenta cuál es el caso de uso sobre el que se va a basar la misma.

Si el caso de uso está relacionado con las transacciones financieras, es recomendable utilizar una red de segunda generación como Ethereum o Ripple. Sin embargo, si se quiere implementar una aplicación descentralizada basada en un smart contract se debería utilizar una red de tercera generación como Eris ya que el proceso madurativo es mucho mayor que Hyperledger y, por tanto, las prestaciones ofrecidas son mayores.

Actualmente, Hyperledger es una gran candidata a ser una de las redes Blockchain de tercera generación más potentes. Sin embargo, debido a su grado de desarrollo, es más recomendable utilizarla para desarrollos de contratos o implementación de pruebas de concepto.

2.4. Comparativa general de las plataformas Blockchain

	Ethereum	**ERIS**	**Ripple**	**Hyperledger**
Red pública	Si	No	Si	No
Tipos de claves	ECDSA ed22519 curve	ECDSA secp2k1 curve	ECDSA secp2k1 curve	ecdsa-with-SHA256
Criptomoneda	Ether	No	Ripple coin	No
Interacción WebSocket	Si	Sí	Si	No
Rest HTTP	Si	Si	Si	Si
Interfaz de comandos	Si	Si	Sí	Si

Tabla 1: Comparativa de plataformas. Parte 1.

	Ethereum	Eris	Ripple	Hyperledger
Lenguaje de implementación	Go	Go	C++	Go
Lenguaje de contratos inteligentes	Solidity	Solidity	No lo soporta	Go and JavaScript (en el futuro)
Red permisionada	No	Puede serlo	Sí, tiene un propio	Puede serle
Algoritmo de consenso	Proof of Stake	Proof of Work	RPCA (Ripple Consensus Algorithm)	PBFT (Practical Byzantine Fault Tolerance)
Licencia	Lesser General Public License 3.0	General Public License 3.0	ISC License	Apache License 2.0
Fecha de lanzamiento	Julio 2015	Agosto 2015	Septiembre 2012	Septiembre 2016

Tabla 2: Comparativa de plataformas. Parte 2.

2.5. Limitaciones de las redes Blockchain

Debido al hecho de que la industria Blockchain todavía está en sus primeras etapas de desarrollo, hay muchos tipos diferentes de posibles limitaciones. Las clases de limitaciones son tanto internas como externas, e incluyen aquellas relacionadas con problemas técnicos con la tecnología subyacente, la industria creciente de robos y escándalos, la percepción de la opinión pública, la regulación gubernamental, y la corriente dominante en la adopción de la tecnología. [20]

Entre las limitaciones técnicas destacan las siguientes:

1. **Capacidad de procesamiento:**

Las redes blockchain tienen una capacidad de procesamiento, por regla general, de 1 transacción por segundo (tps), con un máximo teórico de 7 tps. Una forma de solucionarlo sería haciendo un tamaño de bloque más grande.

2. **Desperdicio de recursos**

La actividad de minar bloques en las redes Blockchain pierde una gran cantidad de energía. Se estima que se desperdician 15 millones de dólares al día.

3. **Latencia**

En el caso de Bitcoin, cada bloque tarda 10 minutos en ser procesado, por consiguiente, la validación de una sola transacción tardaría dicho tiempo. Este suceso ocurre en la mayoría de las redes Blockchain.

4. **Seguridad**

Hasta la fecha hay varios problemas de seguridad en las redes Blockchain, por ejemplo, en los smart contracts las variables son almacenadas en claro. De este modo, si se realiza un ataque de fuerza bruta hasta conseguir el hash del contrato, se podría acceder a información que éste almacene.

5. **Versiones, hard forks y múltiples cadenas**

Algunas otras cuestiones técnicas tienen que ver con la infraestructura. Un problema es la proliferación de cadenas de bloques. Al haber tantas de ellas diferentes, puede ser fácil de implementar un ataque para lanzarlo al 51% de las cadenas más pequeñas.

Otro problema es que cuando las cadenas se dividen por cuestiones administrativas o versiones no hay manera de fusionarlas con otras cadenas.

2.6. Beneficios de las redes Blockchain

Sin embargo, a pesar de las limitaciones, algo común para una tecnología tan sumamente joven como Blockchain, hay numerosos beneficios que pueden marcar un antes y un después en nuestra concepción de como actuamos con el mundo.[21] Los beneficios son los siguientes:

1. **Desintermediación e intercambio "trustless"**

Dos partes son capaces de hacer un intercambio sin la supervisión o la intermediación de un tercero, reduciendo fuertemente o, incluso, eliminando el riesgo.

2. **Da poder a los usuarios**

Los usuarios tienen el control de toda su información y de sus transacciones

3. **Alta calidad de los datos**

En una red Blockchain los datos son completos, consistentes, precisos y altamente disponibles.

4. **Durabilidad, longevidad y fiabilidad**

Debido a las redes descentralizadas, Blockchain no tiene una entidad centralizada y, por ello, está mejor preparado para resistir ataques malintencionados.

5. **Integridad de procesos**

Los usuarios pueden confiar en que las transacciones se ejecutarán exactamente con los comandos de protocolo, eliminando la necesidad de un tercero de confianza.

6. **Transparencia**

Los cambios en la red Blockchain son públicamente visibles por todas partes creando transparencia, y todas las transacciones son inmutables, lo que significa no puede ser alterados o eliminados.

7. **Simplificación del sistema**

Al agregar todas las transacciones en un único sistema de contabilidad pública se reducen el desorden y las complicaciones.

8. **Transacciones más rápidas**

Las transacciones entre bancos físicos normalmente suelen tardar días. Las transacciones mediante Blockchain pueden ser reducidas temporalmente a minutos.

9. **Costes de transacción más bajos**

Eliminando el intermediario que funciona como tercera parte, se eliminan los costes de intercambios.

2.7. Casos de uso comunes de las redes Blockchain

1. **Préstamos y pagos de punto a punto**

Hasta la fecha todos los tipos de acciones bancarias conllevan demasiado tiempo de ejecución. Por ejemplo, si una entidad A quiere hacer un envío de dinero a una entidad B, debido a la situación actual, este proceso tardaría días en ejecutarse.

Este retraso, se debe principalmente en las terceras partes de las cuales depende la transacción.

Utilizando blockchain se eliminarían los intermediarios de modo que se reduciría considerablemente el tiempo y, por otro lado, los costos asociados también descenderían.

2. **Internet of Things**

Los sistemas IoT actuales están basados en modelos de comunicación centralizada, utilizando el paradigma conocido como cliente-servidor.

Todos los dispositivos se identifican, autentican y se conectan a servidores en la nube que tienen grandes capacidades de procesamiento. Sin embargo, a pesar de las capacidades de éstos, pueden producirse colapsos o pérdida de funcionalidad debido a una caída del servidor usado.

Utilizando un modelo descentralizado usando el protocolo de punto a punto, solucionaría este tipo de problemas y reduciría significativamente los costes asociados con la instalación y el mantenimiento de grandes centros de datos centralizados. Sin embargo, el uso de esta tecnología presenta una serie de inconvenientes de tipo de seguridad.

Puesto que el tema de seguridad es muy importante en IoT, debido a la sensibilidad de los datos que se procesan, usar Blockchain permitiría abordar el uso del protocolo punto a punto sin tener dichas dificultades.

3. Sistemas de votación

Los sistemas de votación electrónico sufren un gran defecto de diseño: utilizan un sistema centralizado. Esto implica que solo existe un proveedor que controla el código, la base de datos y las salidas del sistema.

Por otro lado, debido a la carencia de un código abierto independiente que compruebe los resultados, es difícil adquirir la confiabilidad requerida por los votantes y los organizadores de la elección.

Utilizando Blockchain como una base de datos segura y transparente se solucionaría este problema y se podrían registrar los votos de una manera confiable.

4. Mercados Descentralizados

Un mercado descentralizado es una estructura de mercado que consta de una red de diferentes dispositivos que permiten crear un mercado sin una ubicación centralizada. De este modo, la tecnología ofrece a los compradores el acceso a varias ofertas, haciendo posible el trato directo con otros distribuidores.

En un futuro, los mercados descentralizados acabarán sustituyendo al comercio electrónico actual. Esto será posible debido a que dos personas podrán comerciar la una con la otra sin depender de cualquier institución.

5. Registro de la propiedad

Actualmente los registros de la propiedad tienen algunas lacras como el fraude y la corrupción.

Por otro lado, en ocasiones, el precio para que un habitante pueda conseguir el registro de su vivienda puede ser excesivo.

La justificación para la aplicación de Blockchain a este campo implica, disminuir los costes que suponen realizar el registro de la propiedad, conseguir transparencia y seguridad, eliminar la corrupción y fraude del proceso y, el más importante, proporcionar inalterabilidad al título.

6. Digitalización de documentos y contratos

Con el aumento de tipos de datos y formatos respectivos, la necesidad de integrarse y compartir datos a través de sistemas se ha hecho esencial. Para la mayor parte de organizaciones, esto implica el equilibrio delicado de los procesos que mueven datos entre sistemas.

Blockchain juega un importante rol en un enfoque integral de la innovación y la estrategia de gestión de riesgos, incluidos los conceptos de cyber-responsabilidad, gran cantidad de datos y telemática

7. **Autenticación de bienes**

La vida futura de bienes puede ser drásticamente modificada, a través de la existencia de un registro que contenga su ciclo de vida y permita su rastreo de la cadena de suministro.

Actualmente, esto es posible gracias a la tecnología Blockchain.

Sin embargo, aunque los objetos de lujo como oro, diamantes y relojes sean ejemplos interesantes, los trastornos de la utilización de esta tecnología dentro de la identificación y autenticación radica en la industria de la salud.

Según la Interpol, más de 200,000 personas mueren anualmente por todo el mundo debido a la ingestión antihistamínicos falsificados.

Utilizando Blockchain como registro de los bienes ayudaría de forma significativa a la lucha contra la falsificación.

2.8. Clasificaciones del historial médico electrónico

En la actualidad no existe un estándar para generar historiales médicos electrónicos, sino que hay múltiples formatos de los mismos. Entre ellos destacan:

2.8.1. HL7

Ilustración 19: HL7

HL7 (Health Level Seven; Nivel 7 de Salud) [1] es una organización sin fines de lucro, acreditada la organización ANSI de desarrollo de normas organización dedicada a proporcionar un marco amplio y las normas para el intercambio, la integración, el intercambio y la recuperación electrónica de información de salud que apoya la práctica clínica, la gestión, la ejecución y la evaluación de servicios de salud.

"Level 7" se refiere al séptimo nivel de la Organización Internacional para la Estandarización (ISO) siete comunicaciones de capa de modelo de Interconexión de Sistemas Abiertos (OSI) - el nivel de aplicación. El nivel de la aplicación interactúa directamente y realiza servicios de aplicación comunes para los procesos de la aplicación. Aunque otros protocolos han sustituido en gran parte, el modelo OSI sigue siendo valioso como un lugar para comenzar el estudio de arquitectura de red.

2.8.2. UMLS

Ilustración 20: UMLS

UMLS (Unified Medial Language System; Sistema de lenguaje médico unificado) [2] es un estándar creado por la Librería Nacional de Medicina (NLM) con el objetivo de facilitar el desarrollo de sistemas informáticos que se comportan como si "entendieran" el significado del lenguaje de la biomedicina y la salud.

Como parte de la UMLS, la NLM produce y distribuye las fuentes de conocimiento (bases de datos) de la UMLS y las herramientas de software. Con el objetivo de dar un uso por parte de los desarrolladores de sistemas en la construcción o el mejoramiento de los sistemas de información electrónica que crean, procesan, recuperan y/o integran un conjunto de biomedicina y datos de salud.

Por el diseño, las fuentes de conocimiento UMLS no son limitadas con aplicaciones particulares, sino que son multiuso. Los desarrolladores pueden aplicar estas fuentes en sistemas que realizan una variedad de funciones que implican uno o varios tipos de la información, p.ej., archivos pacientes, pautas, datos de la salud pública…, etc.

2.8.3. SNOMED CT

Ilustración 21: SNOMED CT

SNOMED Clinic Terms [3]es un equipo sistemáticamente organizado y de procesado que incluye una recopilación de términos médicos que permite proporcionar códigos, términos, sinónimos y definiciones utilizadas en la documentación clínica y la presentación de informes. SNOMED CT es considerada la terminología en el mundo más completa y multilingüe de atención de salud.

SNOMED CT es mantenida y distribuida por SNOMED International, una organización de desarrollo de normas internacionales sin fines de lucro, ubicada en Londres, Reino Unido. SNOMED International es el nombre comercial de la “Health Terminology Standards Development Organisation” (IHTSDO; Organización de desarrollo de términos estándar de salud), establecido en 2007.

El SNOMED CT asegura el intercambio de información consecuente y es fundamental para un registro de la salud electrónico interoperable.

Proporciona un medio consistente para indexar, almacenar, recuperar y agregar datos clínicos a través de las especialidades y lugares de atención. También ayuda a organizar el contenido de los historiales médicos electrónicos los sistemas, reduciendo la variabilidad en la forma en que los datos son capturados, codificados y utilizados para la atención clínica de los pacientes y la investigación

2.8.4. ICD 11

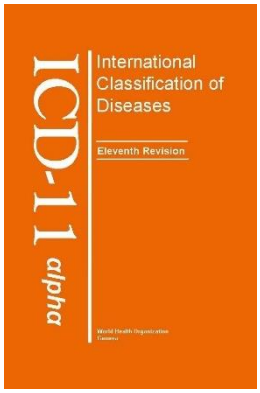


Ilustración 22: ICD-11

ICD (International Clasification of Desease); Clasificación Internacional de Enfermedades) [4] fue adoptada por el Instituto Internacional de estadística en 1893.

La clasificación es mantenida por la Organización Mundial de la Salud (OMS), la autoridad directiva y coordinadora de la acción sanitaria en el sistema de las Naciones Unidas.

Está diseñada como un sistema de clasificación de la salud, proporcionando un sistema de códigos de diagnóstico para la clasificación de enfermedades. Asimismo, incluye matrices de clasificaciones de una amplia variedad de signos, síntomas, hallazgos anormales, denuncias, circunstancias sociales y causas externas de lesión o enfermedad.

Es publicada por la OMS y se utiliza para calcular las estadísticas sobre la obesidad y la mortalidad. Este sistema está diseñado para promover la comparabilidad internacional en la colección, procesamiento, clasificación y presentación de estas estadísticas.

Actualmente se encuentra en la undécima revisión.

3. Algoritmos de consenso de las redes Blockchain

Los problemas de consenso provienen de la computación distribuida con el fin de lograr una fiabilidad total en presencia de una serie de procesos defectuosos. Esta dependencia por la fiabilidad es debida a que los procesos, en ocasiones, necesitan llegar a un acuerdo sobre los datos en algún momento. Con el fin de solucionar este problema, nacieron los algoritmos de consenso.

Los algoritmos de consenso cobran un gran papel en las redes Blockchain. Un algoritmo de consenso es un mecanismo seguro que permite la actualización de una transacción en la red. Mediante este mecanismo, los nodos que componen una red Blockchain realizan un acuerdo para poder marcar una transacción como válida o como inválida.

Cada red Blockchain puede tener un distinto algoritmo de consenso. De este modo, los algoritmos de consenso utilizados son*: Proof-of-Work* (PoW) en el caso de Ethereum, *Proof-of-Stake* (PoS) en el caso de Eris, Ripple Consensus Algorithm (RPCA) en el caso de Ripple y Practical Byzantine Fault Tolerance (PBFT) en el caso de Hyperledger. En el caso de esta última plataforma, aparte de utilizar PBFT, ésta ha sido implementada para poder seleccionar qué algoritmo de consenso utilizar.

Antes de centrarnos en los algoritmos, se ha de conocer el término “minar”. Dicho término es muy utilizado en esta tecnología.

El proceso de minar tiene como comienzo la creación de una transacción. Con el fin de poder crear un bloque que la contenga, el nodo que lo va a crear tiene como obligación crear un hash, problema matemático que define el nivel de dificultad del proceso de minado.

La dificultad del hash permite que sea casi imposible realizar ataques a la cadena, debido a su coste computacional.

Una vez que el nodo ha propuesto el hash, el resto de nodos tienen que resolver el problema. A la parte del nodo que realiza la parte computacional, se le conoce como minero.

De este modo se decide si un bloque es introducido en la cadena o no. Este proceso de cálculo y de decisión es diferente en función del algoritmo de consenso que posea la red.

3.1.1. Proof-of-Work

Proof-of-Work es el algoritmo más común y utilizado en las redes Blockchain. Fue utilizado por primera vez en la plataforma Bitcoin. Actualmente, entre las redes que hemos estado estudiando, es utilizado por Ethereum.

El algoritmo es el siguiente: el minero parte de la construcción de un bloque que ha de ser validado y que contiene transacciones. El minero calcula el hash de este boque, el cual está formado por la suma de los hashes de todas las transacciones que contienen el bloque, el hash del bloque anterior y un número, llamado *nonce*. El objetivo del resto de mineros es el cálculo del *nonce* para llegar a un acuerdo con el estado (válido o inválido) del bloque.

Este algoritmo tiene una alta carga computacional, principalmente proveniente del cálculo del *nonce*.

Está previsto que, en la próxima versión de Ethereum, el algoritmo de consenso sea cambiado *a Proof-of-Stake.*[22]

3.1.2. Proof-of-Stake

Proof-of-Stake, nació teniendo como base *Proof-of-Work* y, como hemos comentado anteriormente, es el algoritmo de consenso de Eris. Como veremos más adelante, la carga computacional de este algoritmo es mucho menor que la del algoritmo anterior. Esto es debido a que la principal motivación *de Proof- of-Stake* es que el sistema de minería se realiza en función de los *stakeholders* de la red.

A la hora de llegar a un acuerdo en el consenso, el nodo que tenga más poder de voto es el que tiene más influencia sobre la decisión de validar o no una transacción. De ahí que se realiza en función de los *stakeholde*rs de la red.

Este poder de voto es definido a la hora de crear la red. Y si se quiere, con el fin de mantener la descentralización, se puede definir que todos los nodos tengan el mismo poder de voto.

Este algoritmo, con vistas a un futuro lejano, podría dejar de cumplir el principio más importante de las redes Blockchain, la descentralización. Para que esto ocurriera, el nodo debería tener el 51% del poder, cosa prácticamente imposible hasta la fecha.

Debido a esto, el proceso de minado tiene mucha menos carga computacional, lo cual hace que este algoritmo de consenso posea una menor latencia.

3.1.3. Ripple Consensus Algorithm

Ripple desarrolló su propio algoritmo de consenso resistente a los fallos Bizantinos y es ejecutado cada pocos segundos en todos los nodos que componen la red.[23]

A diferencia con el resto de algoritmos, el algoritmo de consenso de Ripple es el que menos tiempo de latencia tiene debido a que es una red en producción con el uso de criptomonedas. En estos casos, es muy necesario conseguir el mínimo tiempo para llegar a un acuerdo entre las distintas partes.

Puesto que RPCA se repite iterativamente, en cada una de sus iteraciones realizan los siguientes pasos.

1. Los nodos cogen las transacciones que no han sido validadas y las hacen públicas en una lista de candidatas a validar.
2. Cada uno de los nodos vota la veracidad de esas transacciones.
3. Las transacciones que han sido marcadas como válidas pasarán a la siguiente iteración si han recibido un mínimo porcentaje de validez, y las que no lo hayan recibido serán descartadas.
4. Las rondas terminan si se ha llegado a un 80% como porcentaje de validez.

Una vez que se han terminado las iteraciones y se ha llegado a un resultado, las transacciones que han sido marcadas como totalmente válidas serán almacenadas en el libro de registros o "ledger".

3.1.4. Practical Byzantine Fault Tolerance

Finalmente, Hyperledger utiliza el algoritmo PBFT para entornos de producción, el que el usuario desee (en un futuro) y NOOPS (*No Operations*) para entornos desarrollo. El protocolo PBFT (*Practical Byzantine Fault Tolerance*) tiene como base el problema de los generales bizantinos, problema formulado por Leslie Lamport en el año 1982.[24]

Este problema es de tipo mental y fue creado con el objetivo de ilustrar el dilema de poder llegar a un consenso entre un conjunto de entidades con un fin común. Sin embargo, entre las entidades puede haber traidores, es decir, otro tipo de entidades que intentan impedir la realización del consenso.

Asimismo, se supone que el entorno es de tipo hostil y, por ello, las comunicaciones son inseguras y limitadas.

El problema, en el documento científico "Bitcoins y el problema de los generales bizantinos", se presenta de la siguiente manera:

"El problema se presenta como una analogía con un escenario de guerra, donde un grupo de generales bizantinos se encuentran acampados con sus tropas alrededor de una ciudad enemiga que desean atacar. Después de observar el comportamiento del enemigo los generales deben comunicar sus observaciones y ponerse de acuerdo en un plan de batalla común que permita atacar la ciudad y vencer. Para ello, los generales se comunican únicamente a través de mensajeros. Además, existe la posibilidad que algunos de los generales sean traidores y, por lo tanto, decidan enviar mensajes con información errónea con el objetivo de confundir a los generales leales. Un algoritmo que solucione el problema debe asegurar que todos los generales leales acuerdan un mismo plan de acción y que unos pocos traidores no pueden n conseguir que el plan adoptado por los generales leales sea equivocado"

El algoritmo requiere de tres pasos o fases: la primera para la difusión del bloque, y los dos últimos para la difusión de las firmas. Es decir, la difusión de si el bloque ha sido validado o no ha sido validado.[25]

Sin embargo, este algoritmo tiene dos problemas. El primero es que todas las partes que dan forma a la red tienen que estar de acuerdo con los participantes que colaboran en el proceso. El segundo, es que el ingreso en un sistema del acuerdo Bizantino es puesto por una autoridad central o negociaciones cerradas, esto podría repercutir en la descentralización.

La idea principal del algoritmo es que, para poder validar una transacción en n nodos, ha de ser validada por dos tercios de los nodos más uno para que esta sea correcta.

Actualmente, dado al estado de crecimiento de Hyperledger, su versión estable sólo incluye el modo para desarrolladores y sólo se puede utilizar el algoritmo de consenso NOOPS, en el que se toman todas las transacciones como válidas para incluirlas en la cadena.

En la siguiente tabla se muestra una comparativa de las plataformas en función a sus capacidades.

Algoritmo	*Descentralización*	*Baja latencia*	*Confiabilidad*
Proof-of-Stake	Sí		
Proof-of-Work	Sí	Puede tenerla	
PBFT		Sí	Sí
RPCA	Sí	Sí	Sí

Tabla 3: Comparativa de algoritmos. Fuente: Vukolik (2015) [13]

4. Análisis

En este apartado se realizará un estudio de aquello que se espera a diseñar. De este modo se podrán identificar las necesidades del producto a desarrollar. Entre estas necesidades se encuentran principalmente las funcionalidades requeridas en el sistema. En primer lugar, se plateará el problema a abordar sobre la gestión de historiales médicos electrónicos.

En segundo lugar, tras el planteamiento del problema, será necesario reunirse con el cliente y recoger los requisitos de usuario. Por otro lado, paralelamente, se especificarán los casos de uso con el fin de analizar el comportamiento del sistema.

En tercer lugar, se especificarán los requisitos software funcionales y los no funcionales.

4.1. Planteamiento del problema

Como se ha introducido en los apartados anteriores, la gestión de los historiales médicos hasta la fecha ha sido un proceso con lacras de seguridad y en el que el paciente no tiene derecho a la gestión de sus propios datos.

Asimismo, no existe un estándar universal que utilicen todas las entidades siguiendo el que haya sido definido, sino que, como hemos visto en el Estado del Arte, existen múltiples formatos que son utilizados en función de la demanda y de los datos que se quieran almacenar.

Por otro lado, existe una desconexión entre la sanidad privada y la sanidad pública. Son miles las personas que poseen un seguro médico público, dependiendo del país en el que habiten, y a su vez un seguro privado. Las comunicaciones entre ambas son prácticamente nulas. Y, por otro lado, los pacientes tienen sus datos distribuidos entre las distintas entidades.

Se propone hacer un sistema en el que el paciente pueda almacenar los datos de su historial médico y poder compartirlos únicamente con las entidades o personas que éste desee.

El principal objetivo del sistema es garantizar que el paciente pueda gestionar sus propios datos y mantenerlos de forma única y unificada.

Por otro lado, con vistas a un futuro, se podría definir la aplicación como un estándar a utilizar entre las diversas entidades relacionadas con la sanidad, tanto pública como privada.

4.2. Requisitos de usuario

En ocasiones, entre el cliente y el desarrollador puede haber muchas diferencias debido a que es difícil satisfacerlo plenamente y cumplir con todos los requisitos impuestos. En otras palabras, en los procesos software es bastante complicado dar al cliente todo lo que espera si no se ha tenido un proceso de comunicación con él previamente.

Con el objetivo de satisfacer las exigencias del cliente y que no haya diferencias entre éste y el desarrollador, se han recogido los requisitos de usuario. Este tipo de requisitos describen en un lenguaje formal los requerimientos funcionales y no funcionales del sistema. De este modo, estos requerimientos se realizan de una forma comprensible por el usuario debido a su falta de carácter técnico detallado.

Al igual que veremos con el resto de requisitos, los requisitos de usuario serán recogidos en la tabla que aparece a continuación.

IDENTIFICADOR: RU-XX

TÍTULO	
PRIORIDAD	
DESCRIPCIÓN	

Tabla 4: Definición de requisitos de usuario

Donde el significado de los campos es el siguiente:

- Identificador: Permite identificar a cada requisito de forma única.
- Título: Título asignado.
- Prioridad: Relevancia considerada por el cliente a la hora de tener en cuenta el requisito. Puede ser alta, media o baja.
- Descripción: Explicación sobre el requerimiento.

IDENTIFICADOR: RU-01

TÍTULO	Almacenamiento de datos
PRIORIDAD	Alta
DESCRIPCIÓN	El usuario podrá almacenar su historial médico electrónico en el sistema.

Tabla 5: RU-01 Almacenamiento de datos

IDENTIFICADOR: RU-02

TÍTULO	Recuperación de datos
PRIORIDAD	Alta
DESCRIPCIÓN	El usuario podrá recuperar su historial médico electrónico.

Tabla 6: RU-02 Recuperación de datos

IDENTIFICADOR: RU-03

TÍTULO	Modificación de los datos
PRIORIDAD	Alta
DESCRIPCIÓN	El usuario no podrá modificar los datos almacenados previamente.

Tabla 7: RU-03 Modificación de los datos

IDENTIFICADOR: RU-04

TÍTULO	Adición de los datos
PRIORIDAD	Alta
DESCRIPCIÓN	El usuario podrá añadir nuevos datos a su historial médico electrónico.

Tabla 8: RU-04 Adición de datos

IDENTIFICADOR: RU-05

TÍTULO	Idioma de la interfaz
PRIORIDAD	Alta
DESCRIPCIÓN	El sistema mostrará la interfaz en inglés.

Tabla 9: RU-05 Idioma de la interfaz

IDENTIFICADOR: RU-06

TÍTULO	Aplicación web
PRIORIDAD	Alta
DESCRIPCIÓN	La aplicación web se adaptará a cualquier dispositivo en el que sea utilizada.

Tabla 10: RU-06 Aplicación web

IDENTIFICADOR: RU-07

TÍTULO	Disponibilidad de los datos
PRIORIDAD	Alta
DESCRIPCIÓN	Los datos almacenados en el historial médico estarán siempre disponibles

Tabla 11: RU-07 Disponibilidad de los datos

IDENTIFICADOR: RU-08

TÍTULO	Propiedad de los datos
PRIORIDAD	Alta
DESCRIPCIÓN	El usuario tendrá control sobre sus datos y decidirá con quién compartirlos

Tabla 12: RU-08 Propiedad de los datos

4.3. Casos de uso

Vistos los requisitos de usuario, se analizarán los casos de uso del sistema. De este modo se podrá estudiar la posible secuencia de interacciones entre los actores del sistema y el propio sistema.

Un actor, es un usuario externo al sistema que requiere una funcionalidad y guarda una relación con el mismo. En el caso de este sistema, van a existir dos tipos de actores:

- Paciente: Rol asociado al usuario que interactúa contra la parte del sistema referida a la gestión de historiales médicos electrónicos. Es necesario que el usuario disponga de un dispositivo conectado a internet para poder interactuar con el sistema.
- Aseguradora: Rol asociado al usuario que puede visualizar el historial médico electrónico, siempre y cuando un usuario de tipo paciente le haya dado acceso al mismo. En este caso, también es necesario que el usuario disponga de un dispositivo conectado a internet para poder interactuar con el sistema.

A la hora de analizar los casos de uso se realizará un diagrama sobre el caso de uso y después se especificará mediante la descripción del mismo.

4.3.1. Diagramas de casos de uso

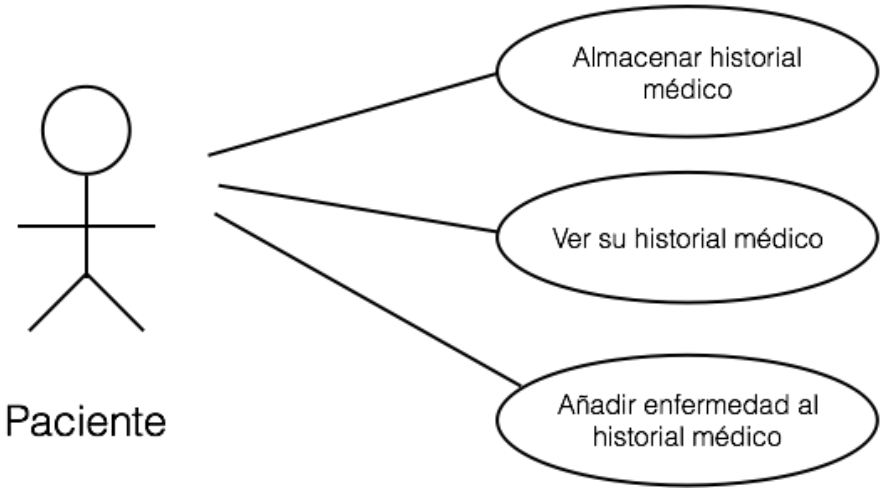


Ilustración 23: Diagrama de caso de uso. Paciente.

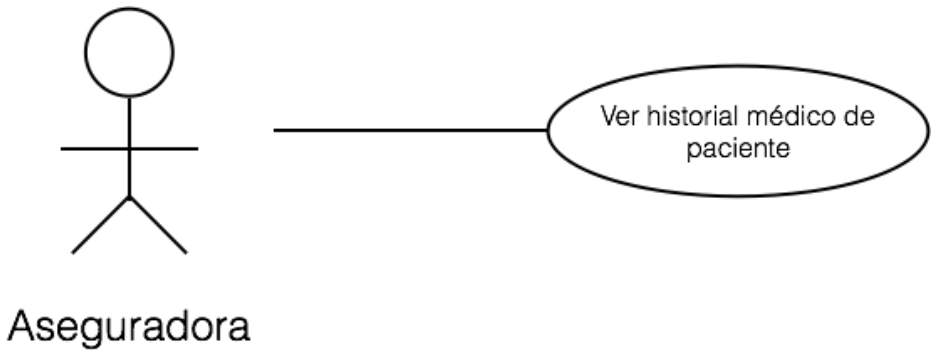


Ilustración 24: Diagrama de caso de uso. Aseguradora

4.3.2. Descripción de los casos de uso

Con el fin de hacer una explicación detallada de los casos de uso se ha utilizado la siguiente tabla:

IDENTIFICADOR: CU-XX

TÍTULO	
ACTOR	
OBJETIVO	
PRECONDICIONES	
POSTCONDICIONES	
ESCENARIO BÁSICO	

Tabla 13: Descripción de casos de uso

Donde los atributos de la tabla se refieren a lo siguiente:

- Identificador: Identifica únicamente al caso de uso mediante las siglas CU y XX, donde lo último es el número del caso de uso.
- Título: Título asignado al caso de uso.
- Actor: Usuario externo al sistema que realiza la interacción.
- Objetivo: Caso de uso, definido en el diagrama.

- Precondiciones: Condición o condiciones necesarias antes de haber realizado la acción que refiere al caso de uso.
- Postcondiciones: Condición o condiciones que han de ser cumplidas cuando se ha realizado la acción.
- Escenario básico: Explicación detallada del caso de uso.

A continuación, se muestra la descripción de los casos de uso definidos en el apartado anterior. De forma resumida, los casos de uso serán los siguientes:

- Almacenar el historial médico
- Recuperar el historial médico
- Añadir enfermedad al historial médico
- Ver historial médico de un paciente

IDENTIFICADOR: CU-01

TÍTULO	Almacenamiento del historial médico.
ACTOR	Paciente.
OBJETIVO	Almacenamiento del historial médico de un paciente de forma electrónica.
PRECONDICIONES	El paciente ha proporcionado sus datos al desarrollador.
POSTCONDICIONES	Los datos han sido almacenados en el sistema.
ESCENARIO BÁSICO	• Acceder a la aplicación web • Los datos han sido almacenados de forma automática

Tabla 14: CU-01 Almacenar historial médico

IDENTIFICADOR: CU-02

TÍTULO	Recuperar el historial médico.
ACTOR	Paciente.
OBJETIVO	Los datos del historial médico de un paciente son mostrados al mismo.
PRECONDICIONES	Los datos fueron almacenados en el sistema.
POSTCONDICIONES	Mostrar los datos en la aplicación.
ESCENARIO BÁSICO	• Acceder a la aplicación web • Ir a la pantalla donde se muestran los datos del paciente

Tabla 15: CU-02 Recuperar el historial médico

IDENTIFICADOR: CU-03

TÍTULO	Añadir enfermedad al historial médico.
ACTOR	Paciente.
OBJETIVO	Introducir una nueva enfermedad al historial médico electrónico.
PRECONDICIONES	Los datos fueron almacenados en el sistema.
POSTCONDICIONES	Mostrar el historial médico actualizado.
ESCENARIO BÁSICO	• Acceder a la aplicación web • Introducir la nueva enfermedad al historial médico. • Mostrar el historial médico donde aparece la enfermedad introducida anteriormente.

Tabla 16: CU-03 Añadir enfermedad al historial médico

IDENTIFICADOR: CU-04

TÍTULO	Ver historial médico de un paciente.
ACTOR	Aseguradora.
OBJETIVO	Mostrar el historial médico de un paciente.
PRECONDICIONES	El paciente ha dado acceso a sus datos a la aseguradora.
POSTCONDICIONES	Mostrar los datos del historial médico.
ESCENARIO BÁSICO	• Acceder a la aplicación web • Introducir el identificador de los datos del paciente. • Mostrar los datos del paciente asociado al identificador.

Tabla 17: CU-04 Ver historial médico de un paciente

4.4. Requisitos funcionales

En vista a las necesidades del cliente, se ha tomado como decisión utilizar una red Blockchain para almacenar su historial médico electrónico. De esta manera se satisfarán sus necesidades de la forma más completa posible.

Como justificación previa, antes de entrar en detalle con los requisitos no funcionales, una red Blockchain permite al usuario tener el pleno control del manejo de sus datos y éstos estarán siempre disponibles.

Por otro lado, como justificación sobre la decisión de almacenar dichos datos en la red Blockchain, visitar el apartado anteriormente descrito*: "2.6. Beneficios de las redes Blockchain"*

Los requisitos funcionales son definiciones de las funcionalidades que ha detener el sistema. Estos requerimientos provienen de los requisitos de usuario y de los casos de uso. Por otro lado, serán complementados por los requisitos no funcionales en el siguiente apartado.

A la hora de hacer esta definición se rellenará la siguiente tabla:

IDENTIFICADOR: RF-XX

TÍTULO	
PRIORIDAD	
ESTABILIDAD	
DESCRIPCIÓN	

Tabla 18: Descripción de los requisitos funcionales

Donde los campos de la tabla poseen el siguiente significado:

- Identificador: Identifica unívocamente al requisito funcional mediante la sigla RF (Requisito Funcional) y XX (número del requisito funcional).
- Título: Título asociado al requisito funcional.
- Prioridad: Fase del diseño técnico de la aplicación en el que se debe incluir la funcionalidad del requisito.
- Estabilidad: Posibilidad de que el requisito pueda ser modificado.
- Descripción: Explicación detallada del requisito.

IDENTIFICADOR: RF-01

TÍTULO	Almacenamiento del historial médico electrónico
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	El sistema podrá almacenar el historial médico de forma electrónica del paciente.

Tabla 19: RF-01 Almacenamiento del historial médico electrónico

IDENTIFICADOR: RF-02

TÍTULO	Adición de enfermedades
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	El sistema permitirá añadir nuevas enfermedades al historial médico electrónico.

Tabla 20: RF-02 Adición de enfermedades

IDENTIFICADOR: RF-03

TÍTULO	Mostrar historial médico
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	El sistema permitirá mostrar el historial médico almacenado.

Tabla 21: RF-03 Mostrar historial médico

IDENTIFICADOR: RF-04

TÍTULO	Nodos validadores
PRIORIDAD	Media
ESTABILIDAD	Alta
DESCRIPCIÓN	El sistema mostrará los nodos validadores de la red Blockchain activos

Tabla 22: RF-04 Nodos validadores

IDENTIFICADOR: RF-05

TÍTULO	Dirección del historial médico
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	El sistema mostrará la dirección en la cual se encuentra el historial médico electrónico en la red Blockchain

Tabla 23: RF-05 Dirección del historial médico

IDENTIFICADOR: RF-06

TÍTULO	Tamaño de la cadena Blockchain
PRIORIDAD	Baja
ESTABILIDAD	Alta
DESCRIPCIÓN	El sistema mostrará cuál es el último bloque añadido a la red Blockchain.

Tabla 24: RF-06 Tamaño de la cadena Blockchain

IDENTIFICADOR: RF-07

TÍTULO	Direcciones de los nodos validadores
PRIORIDAD	Baja
ESTABILIDAD	Estable
DESCRIPCIÓN	El sistema mostrará las direcciones o "address" de los nodos validadores que conforman la red Blockchain.

Tabla 25: RF-07 Dirección de los nodos validadores

IDENTIFICADOR: RF-08

TÍTULO	Autenticación del usuario
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	El sistema permitirá autenticar al usuario para entrar a la aplicación web.

Tabla 26: RF-08 Autenticación del usuario

IDENTIFICADOR: RF-09

TÍTULO	Modificación de los datos
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	El sistema no permitirá modificar los datos una vez almacenados en el sistema.

Tabla 27: RF-09 Modificación de los datos

IDENTIFICADOR: RF-10

TÍTULO	Búsqueda de enfermedad
PRIORIDAD	Baja
ESTABILIDAD	Estable
DESCRIPCIÓN	El sistema permitirá introducir enfermedades utilizando un buscador de las mismas.

Ilustración 25: RF-10 Búsqueda de enfermedad

IDENTIFICADOR: RF-11

TÍTULO	Almacenamiento dirección historial médico electrónico
PRIORIDAD	Baja
ESTABILIDAD	Estable
DESCRIPCIÓN	El sistema permitirá almacenar la dirección del historial médico electrónico con el objetivo de recuperarlo.

Ilustración 26: RF-11 Almacenamiento historial médico electrónico

4.5. Requisitos no funcionales

Algunos de los atributos propios que posee el sistema no pueden determinar características de funcionalidad, de ahí que existan los requisitos no funcionales.

Los requisitos no funcionales, también conocidos como atributos de calidad, son requerimientos que detallan los criterios que pueden ser utilizados para juzgar un comportamiento específico del sistema. Este tipo de comportamiento está estrechamente correspondido con uno o varios requisitos funcionales.

En otras palabras, los requisitos no funcionales imponen restricciones al diseño, restricciones a la implementación y definen los estándares de calidad que éste debe cumplir.

Al igual que con los requisitos funcionales, se ha definido una tabla para detallar este tipo de requisitos:

IDENTIFICADOR: RNF-XX

TÍTULO	
PRIORIDAD	
ESTABILIDAD	
DESCRIPCIÓN	

Tabla 28: Descripción de los requisitos funcionales

Donde los campos de la tabla poseen el siguiente significado:

- Identificador: Identifica unívocamente al requisito funcional mediante la sigla RF (Requisito No Funcional) y XX (número del requisito no funcional).
- Título: Título asociado al requisito no funcional.
- Prioridad: Fase del diseño técnico de la aplicación en el que se debe incluir la funcionalidad del requisito.
- Estabilidad: Posibilidad de que el requisito pueda ser modificado.
- Descripción: Explicación detallada del requisito.

IDENTIFICADOR: RNF-01

TÍTULO	Idioma de la interfaz
PRIORIDAD	Media
ESTABILIDAD	Estable
DESCRIPCIÓN	El sistema mostrará la interfaz en inglés

Tabla 29: RNF-01 Idioma de la interfaz

IDENTIFICADOR: RNF-02

TÍTULO	Interfaz adaptativa
PRIORIDAD	Media
ESTABILIDAD	Estable
DESCRIPCIÓN	La interfaz se adaptará a cualquier dispositivo que la utilice.

Tabla 30: RNF-02 Interfaz adaptativa

IDENTIFICADOR: RNF-03

TÍTULO	Disponibilidad del historial médico electrónico
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	El historial médico estará disponible siempre que esté la red con todos los nodos activos.

Tabla 31: RNF-03 Disponibilidad del historial médico.

IDENTIFICADOR: RNF-04

TÍTULO	Replicación historial médico electrónico
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	El historial médico se encontrará replicado al menos tres veces.

Tabla 32: RNF-04 Replicación del historial médico

IDENTIFICADOR: RNF-05

TÍTULO	Aprendizaje del sistema
PRIORIDAD	Baja
ESTABILIDAD	Estable
DESCRIPCIÓN	El usuario tardará menos de media hora para aprender cómo utilizar el sistema.

Tabla 33: RNF-05 Aprendizaje del sistema

IDENTIFICADOR: RNF-06

TÍTULO	Acceso al historial médico electrónico
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	El historial médico electrónico sólo será accedido por la persona que posea la dirección del mismo

Tabla 34: RNF-06 Acceso al historial médico electrónico

IDENTIFICADOR: RNF-07

TÍTULO	Calidad de los datos
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	Los datos almacenados serán completos, consistentes, precisos y altamente disponibles.

Tabla 35: RNF-07 Calidad de los datos

IDENTIFICADOR: RNF-08

TÍTULO	Integridad de los datos
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	La transacción de almacenar datos se ejecutará exactamente con los comandos de protocolo de la red Blockchain.

Tabla 36: RNF-08 Integridad de los datos

IDENTIFICADOR: RNF-09

TÍTULO	Tiempo de las transacciones
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	El tiempo de ejecución de las transacciones, almacenar y recuperar historial médico electrónico, será menor a 1 segundo.

Tabla 37: RNF-09 Tiempo de las transacciones

IDENTIFICADOR: RNF-10

TÍTULO	Transparencia de las modificaciones
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	Las modificaciones se realizarán de forma transparente para el usuario.

Tabla 38: RNF-10 Transparencia de las modificaciones

IDENTIFICADOR: RNF-11

TÍTULO	Tasa de errores cometida por el usuario
PRIORIDAD	Media
ESTABILIDAD	Inestable
DESCRIPCIÓN	La tasa de errores que el usuario cometa deberá ser menor del 10% de las transacciones totales realizadas en el sistema.

Tabla 39: RNF-11 Tasa de errores cometida por el usuario

IDENTIFICADOR: RNF-12

TÍTULO	Alertas en la aplicación
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	La aplicación mostrará mensajes de alertas cuando se complete una transacción.

Tabla 40: RNF-12 Alertas en la aplicación

IDENTIFICADOR: RNF-13

TÍTULO	Mensajes de error
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	La aplicación mostrará diálogos de error en el caso de que el usuario cometa un error.

Tabla 41: RNF-13 Mensajes de error

IDENTIFICADOR: RNF-14

TÍTULO	Tolerancia a fallos
PRIORIDAD	Alta
ESTABILIDAD	Estable
DESCRIPCIÓN	La red Blockchain poseerá tolerancia a fallos.

Tabla 42: RNF-14 Tolerancia a fallos

5. Diseño de la aplicación

En este apartado se determina el funcionamiento general del sistema mediante la arquitectura escogida. En este caso, la elegida ha sido una arquitectura de microservicios.

5.1. Diseño de componentes

Un diagrama de componentes es de tipo UML (Lenguaje Unificado de Modelado). El diagrama de componentes tiene como propósito principal la modelación del sistema y todas sus partes que serán implementadas en la fase de implementación. Por ello, se ha realizado el diagrama que veremos a continuación.

Con este diagrama se pueden mostrar los elementos de diseño, viendo así cómo es la estructura a alto nivel de lo que será la aplicación posteriormente. En otras palabras, los diagramas de componentes son usados para hacer una representación de la vista estática y dinámica de un sistema.

Otro de sus usos es el análisis de qué componentes se pueden compartir entre diferentes sistemas, tantos ajenos como internos del mismo.

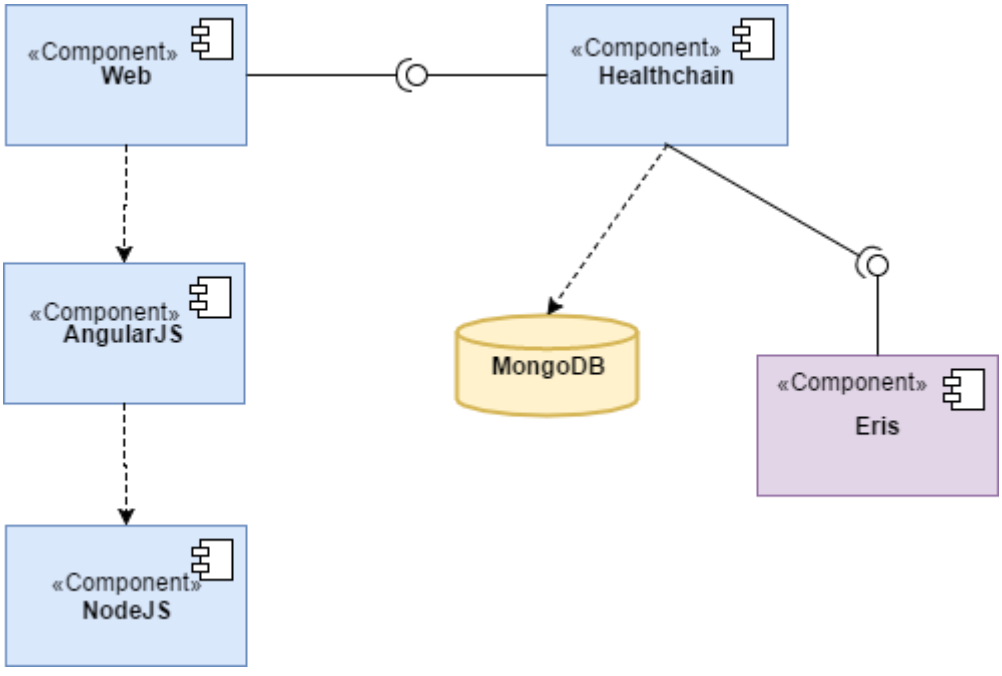


Ilustración 27: Diagrama de componentes

Como podemos comprobar en la imagen, se sigue el modelo Cliente-Servidor. De este modo, el componente Web requiere de la interfaz del componente Healthchain para poder comunicarse con el mismo y poder ofrecer la lógica de negocio de la aplicación a la entidad que interaccione con el mismo.

Los componentes que aparecen en el diagrama son los siguientes:

- Componente Web: Es el encargado de proveer la capa de vista de la aplicación. Interacciona con el componente Healthchain. Tiene como dependencias el componente AngularJS que depende a su vez del componente NodeJS.
- Componente AngularJS: Encargado de proveer el entorno operacional al componente Web.
- Componente NodeJS: Componente que se encarga de proporcional el entorno operacional al componente AngularJS.
- Componente Healthchain: es el encargado de proveer la lógica de negocio a la aplicación. Expone su interfaz de forma pública para que cualquier entidad pueda beneficiarse de las funcionalidades que expone.
- Componente Eris: es el componente que recoge a la red Blockchain. Como hemos analizado previamente y se estudiará con mayor profundidad en el siguiente apartado, Eris posee una interfaz que se expone de manera pública con el objetivo de poder ser utilizada y poder servirse de su funcionalidad. Esta funcionalidad se basa en ser un libro de registros que almacene la información que se quiera mantener la red de forma íntegra, consistente y segura.

5.2. Diseño del entorno operacional

La arquitectura de microservicios, es un nuevo enfoque utilizado a la hora de desarrollar una aplicación. Este tipo de enfoque es debido a que la aplicación software se realiza utilizando varios servicios que se ejecutan autónomamente y comunicándose entre sí, utilizando peticiones del tipo HTTP.

Cada microservicio debe ser independiente al resto y su despliegue no debe afectar de ninguna forma al resto de microservicios que componen la aplicación. Esto, entre otras cosas, quiere decir que una aplicación puede estar construida en distintos servicios desarrollados en diferentes lenguajes.

Al haber escogido microservicios la aplicación se beneficiará de las siguientes características:

- Simplicidad de los servicios que componen la aplicación.
- Interdependencia.
- Reducción en el tiempo del desarrollo de la aplicación.
- Multitecnología.

- Unidades de despliegue pequeñas.
- Tolerancia a fallos.
- Gestión de la configuración.
- Localización de los servicios.
- Fácil escalado horizontal.
- Descentralización.
- Reutilización de componentes.

De esta manera, cada microservicio tiene como función principal la implementación de una función específica en el negocio.

Cabe señalar que la característica de interdependencia requiere de los microservicios que sean diseñados de forma "stateless" o sin estado. Así, la recuperación de un fallo se realiza de forma sencilla y no se realizan sobrecargas de peticiones a los servidores. De la misma forma, gracias a esta característica se pueden multiplicar el número de servidores y poder atender de forma dinámica la demanda de los mismos.

Por otro lado, en cuanto al almacenamiento de datos sus principales propósitos a conseguir son la continuidad operativa y el rendimiento. Por esta razón, son utilizados patrones de acceso en forma de consultas mediante el uso de memorias cache.

Por todas estas razones, el diseño del entorno operacional de la aplicación desarrollada tiene la siguiente estructura:

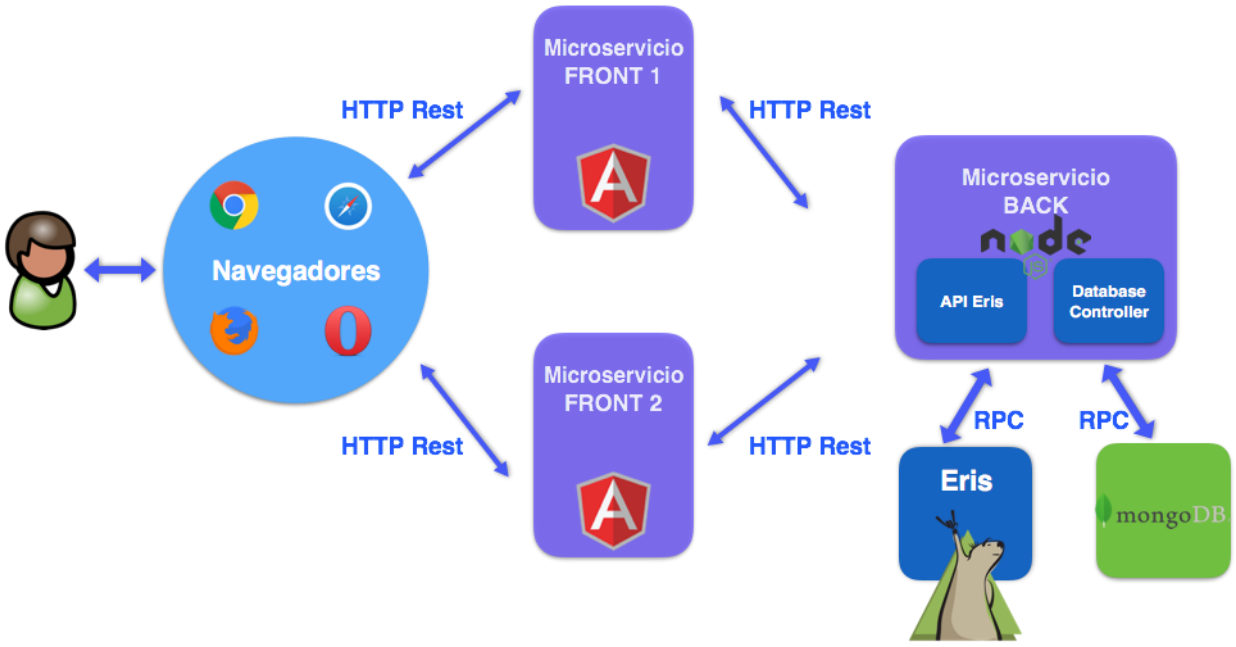


Ilustración 28: Diseño entorno operacional

Como podemos observar en la imagen, la arquitectura constará de los siguientes componentes:

- Microservicios de front-1 y front-2: Corresponderán a la implementación de las interfaces de la aplicación. Puesto que existen dos actores en los casos de uso, se hará una diferenciación entre los mismos realizando dos frontend diferentes.
- Microservicio de back: Implementación del backend de la aplicación. Existirá un servidor que recogerá las peticiones de los microservicios comentados anteriormente y realizará las funcionalidades de negocio correspondientes.
- MongoDB: Base de datos utilizada para almacenar los datos que requiera la aplicación.
- Eris: Componente de la red Blockchain.

Cabe destacar que las comunicaciones entre los microservicios se realizan mediante HTTP Rest. Esto es debido a la API Rest que tienen los mismos y exponen hacia al exterior. Por el contrario, entre el microservicio de backend y los componentes de Eris y MongoDB se realiza una conexión utilizando el protocolo RPC (Remote Procedure Call).

Finalmente, se puede observar fácilmente la característica de reutilización de microservicios. Si analizamos la imagen se puede ver que los microservicios front-1 y front-2 son interdependientes entre ellos. Sin embargo, hacen uso del mismo recurso: el microservicio backend.

En otras palabras, se puede decir que el diseño de la aplicación cumple con el modelo vista controlador. Este modelo es un tipo de patrón utilizado en las arquitecturas software para separar los datos y la lógica del negocio de la interfaz utilizada por los usuarios.

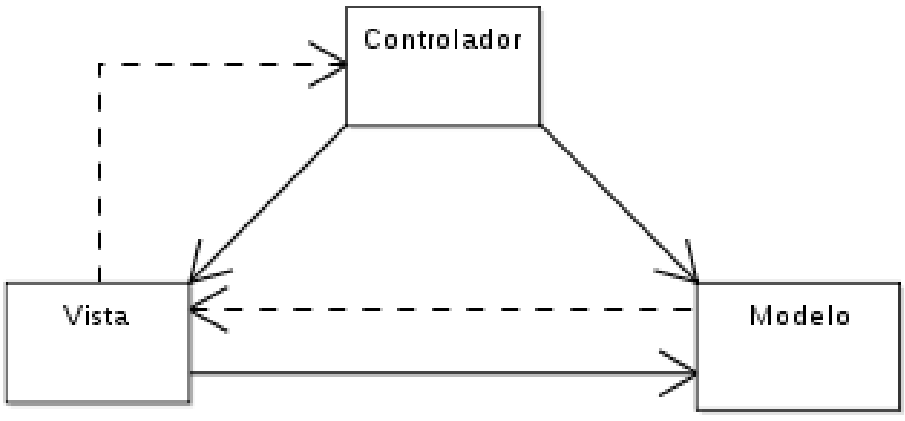


Ilustración 29: Modelo Vista Controlador

De forma general componentes de este tipo de diagrama se pueden definir como los siguiente:

- Modelo: Es la representación de toda la información que es utilizada en la aplicación. Por otro lado, es utilizado de forma directa por el controlador. En nuestro caso, el modelo correspondería con la base de datos de MongoDB. Asimismo, podría decirse que también correspondería con la red Blockchain. En otras palabras, en este tipo de patrón se establece que el controlador "actualiza" y "manipula" la información de forma que podría desaparecer del sistema y no dejar ningún tipo de rastro. Esto en las redes Blockchain es imposible que suceda, porque una vez que se almacena un dato en la cadena no puede ser borrado hasta desaparecer, si no que deja un rastro.
- Controlador: responde a los eventos y comunica la vista con el modelo. Es el principal responsable de gestionar todas las peticiones. En nuestro caso correspondería con el microservicio de backend.
- Vista: presenta de forma visual la lógica del negocio y permite al usuario interactuar con la aplicación. En este caso se correspondería con los microservicios frontend-1 y frontend-2

Tras esta introducción de los componentes que dan forma a la aplicación, se realizará un estudio más detallado.

5.2.1. Front

Debido a la existencia de dos actores, los cuales han sido definidos en los casos de uso, es necesario hacer dos frontend en la aplicación. Uno, que será accedido por los pacientes que quieran almacenar el registro médico electrónico y otro, que será accedido por las aseguradoras para poder almacenar el historial médico de un paciente, siempre y cuando el paciente le haya dado acceso.

Esta pieza es clave en el sistema debido a que es la forma en la que los usuarios pueden interactuar con la aplicación.

Ilustración 30: Microservicio frontend

5.2.2. Back

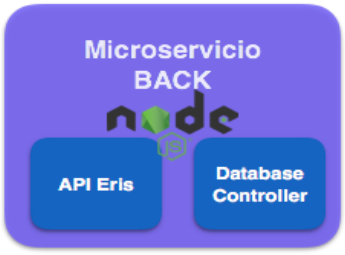


Ilustración 31: Microservicio backend

El componente Back será el núcleo de la aplicación donde se realizará toda la lógica del negocio y funcionará como intermediario entre los frontend y la base de datos junto con la red Blockchain

Este componente poseerá un servidor en NodeJS que recibirá las peticiones de los microservicios front-1 y front-2 y las procesará.

Por otro lado, el servidor deberá contener dos controladores para comunicarse con los componentes Eris y MongoDB.

El controlador de la red Eris consumirá la API de ésta, utilizando como protocolo de comunicación RPC (Remote Procedure Calls).

Del mismo modo, el controlador de la base de datos se comunicará con la misma utilizando el mismo protocolo de implementación.

Cabe destacar que, como el lenguaje elegido es NodeJS, los controladores deberán utilizar las librerías asociadas a dichos recursos.

Finalmente, es de gran relevancia recordar que, gracias a la arquitectura elegida de microservicios, el back es reutilizado entre los dos microservicios de frontend y, además, en el caso de que se quisiera utilizar otra aplicación que consuma este microservicio, los cambios que habría que realizar serían los mínimos.

5.2.3. MongoDB

Ilustración 32: MongoDB

MongoDB es el componente referente a la base de datos que será utilizada para almacenar todos los datos que se requieran a la hora de interactuar con la aplicación.

Entre estos datos se deberán encontrar:

- Las credenciales del usuario.
- La dirección del historial médico electrónico para poder recuperarlo a la hora de acceder a la red Blockchain.

5.2.4. Eris

Eris es el componente referente a la red Blockchain que será utilizado por la aplicación web. El propósito principal en la red es poder almacenar el historial médico electrónico para que se beneficie de las principales características de las redes Blockchain: seguridad, transparencia, disponibilidad, longevidad e integridad, entre otras.

Hasta el momento el almacenamiento de los historiales médicos se realizaba en bases de datos convencionales. Además, como parte negativa, los pacientes no podían gestionar su propio historial electrónico, si no que dependía de terceras partes.

Utilizando Blockchain, el historial médico electrónico ahora pertenece a los pacientes y son ellos los encargados de gestionarlo. Debido a esto, es muy necesario realizar un buen diseño de la composición de la red.

Ilustración 33: Eris

A la hora de hacer la composición de la red Blockchain, la plataforma que se escogió fue Eris. Esta elección se basó en, como hemos comentado anteriormente, la orientación de la plataforma: una plataforma de tercera generación y enfocada a ser utilizada mediante aplicaciones descentralizadas.

Entre las plataformas comentadas que también estaban focalizadas al desarrollo de smart contracts se encontraba Hyperledger. Sin embargo, debido a que ha sido la plataforma que menos tiempo tiene de vida y que se encuentra en estado de desarrollo, al hacer pruebas con la misma se verificó que no tenía aún implementada la funcionalidad que se requería.

Antes de focalizarnos en el despliegue de la red y en un estudio de la composición de la misma, hay que tener en cuenta el tipo de nodos que vayan a componer la misma.

En algunas redes Blockchain, como por ejemplo Ethreum o Hyperledger, los nodos que dan vida a la red y que realizan la validación de las transacciones son del mismo tipo. Sin embargo, en Eris, hay distintos tipos de nodos.

La implementación de distintos tipos de nodos está basada en un sistema de permisos. Gracias a los mismos, los nodos pueden realizar determinadas funciones en la red y así poder optimizar, a su manera, las funciones a realizar.

El sistema de permisos reduce los problemas de autenticación en las cadenas de bloques. Por ejemplo, la plataforma Hyperledger está basada en un sistema de permisos de manera que se pueden administrar los privilegios de la red y requiere de una serie de pasos previos de autenticación bastante complejos

Por el contrario, en Eris, la manipulación de la red a la hora de hacer transacciones es sencilla y en ningún momento se ponen en riesgo los datos.

Cada nodo tiene asociado una cuenta, que toma forma con un tipo "*address*". Los tipos diferentes de cuentas son los que hacen que los nodos sean diferenciados. Podemos distinguir las siguientes:

- **Participante**: no tiene permisos de validación de la cadena, su principal función es distribuir las transacciones por la red.
- **Validador:** tras realizar el algoritmo de consenso, validan las transacciones.
- **Root:** poseen todos los permisos y pueden realizar todo tipo de interacciones en la cadena, validaciones, despliegue de contratos…

- **Full:** poseen todos los privilegios, sin embargo, sus permisos están asociados al nodo génesis. En ocasiones suele ser un nodo problemático, ya que suele tomar el poder de voto de otros nodos y se valida la transacción en función de su resultado, sin tener el resto de nodos en cuenta. Permite validar transacciones.

En nuestro caso, puesto que sólo una aplicación está haciendo uso de la red y puesto que la misma no ha pasado a entornos de producción, no es necesario hacer una red compleja y con mucha tolerancia a fallos. Esto es debido a que el riesgo de sobrecarga de la misma es mínimo, por no decir nulo.

Por la misma razón, solo es necesario tener un único nodo que haga de contacto con la red Blockchain y la aplicación. Éste nodo debería ser de tipo *root o full*. Sin embargo, la plataforma recomienda que en caso de redes pequeñas el nodo que esté enlazado a la aplicación sea el nodo full.

La composición de la red sería la siguiente:

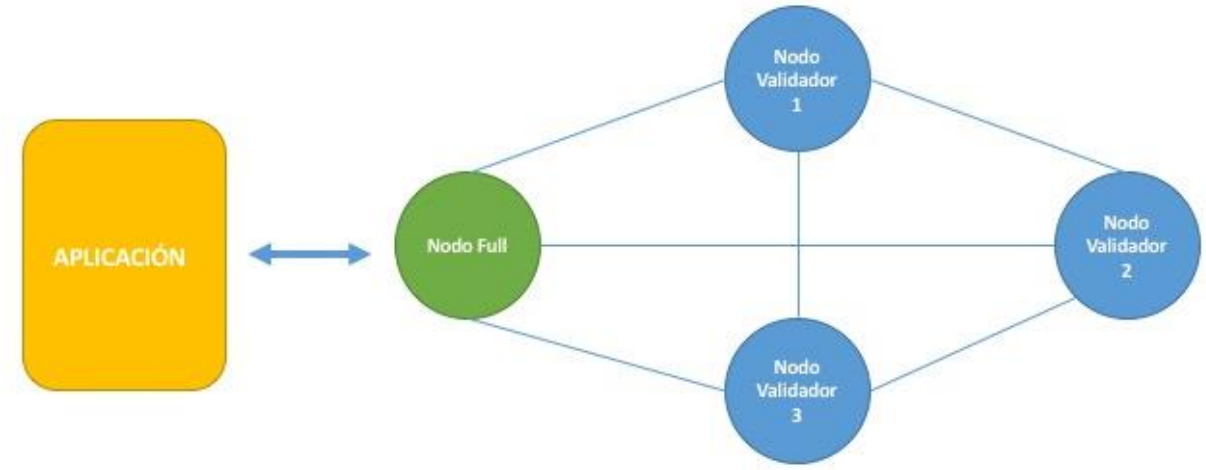


Ilustración 34: Composición de la red

Como se ha comentado anteriormente, para hacer consenso al menos se necesitan 2/3 de la red. Esto quiere decir, que con tres nodos que hicieran validaciones la red podría funcionar. Sin embargo, por tolerancia a fallos, se ha tomado la decisión de incluir un nodo más que pueda mantener la red en el caso de que uno de los nodos falle.

Por otro lado, a la hora de configurar la red, se ha tenido en cuenta el algoritmo de consenso de Eris: *Proof-of-Stake*. Resumidamente, y a modo recuerdo, los nodos tienen un poder de voto que influye sobre la decisión a la hora de validar una transacción. Esto quiere decir que, si un nodo tiene más poder de voto que otro, su decisión cobrará más valor que la del otro nodo. Por esta razón y con el fin de tener

un sistema totalmente descentralizado, a la hora de generar los nodos para crear la red se les ha dado el mismo poder de voto.

Finalmente, cabe destacar que para entornos de producción habría que hacer una red con mayor número de nodos, teniendo en cuenta que cumpliese la siguiente ecuación.

$$Poder\ de\ voto\ de\ la\ red = \frac{2}{3} * Poder\ de\ voto\ total + 1$$

A la hora de generar esta norma, que debe ser cumplida para que haya lo mínimo necesario para poder realizar consenso entre los nodos, se ha tenido en cuenta que los nodos pueden tener distinto poder de voto.

Por otro lado, y con motivo de optimizar la red y que el consenso se haga de una manera más rápida, y con el fin de no tener una red centralizada, los nodos no tienen por qué tener el mismo poder de voto. Sin embargo, es reseñable que ningún nodo debería tener el poder mínimo de voto de la red reflejado en la fórmula anterior, si no se perdería la característica más importante de Blockchain: la descentralización.

A la hora de componer la red se han utilizado contenedores de Docker para poder desplegar los nodos. Sin embargo, es necesario el componente eris:cli del cual hemos hablado anteriormente para generar los nodos y sus cuentas necesarias.

Una vez que los nodos han sido generados y configurados, existen imágenes de Docker que permiten el despliegue de la red.

5.3. Matriz de trazabilidad

Una vez vistos los componentes que darán cabida al sistema a desarrollar, se procederá a la realización de una matriz de trazabilidad de requisitos y componentes. De esta manera se podrá realizar un análisis mediante el mapeo de requisitos y de componentes, viendo así si los requisitos quedan totalmente cubiertos y se puede ofrecer la funcionalidad completa de la aplicación.

De forma más genérica, la matriz de trazabilidad es una herramienta utilizada en la ingeniería del software que permite asegurar que cada requisito del sistema que agrega valor al negocio se vincule a los componentes de negocio de la aplicación.

Por otro lado, también se puede realizar un seguimiento y/o monitorización de la vida del proyecto y sus respectivos cambios.

La matriz de trazabilidad del proyecto es la siguiente:

	FRONT-1	FRONT-2	BACK	ERIS	MONGODB
RF-01			x	x	x
RF-02	x		x	x	x
RF-03	x	x	x	x	x
RF-04				x	
RF-05	x			x	
RF-06	x			x	
RF-07	x			x	
RF-08	x				x
RF-09	x			x	
RF-10	x		x		
RF-11			x	x	x
RNF-01	x	x			
RNF-02	x	x			
RNF-03				x	
RNF-04				x	
RNF-05	x	x			
RNF-06	x	x	x		
RNF-07				x	
RNF-08				x	
RNF-09				x	
RNF-10			x		
RNF-11	x	x			
RNF-12	x				
RNF-13	x				
RNF-14				x	

Tabla 43: Matriz de trazabilidad

6. Implementación y pruebas

Tras haber hecho el diseño de la aplicación se procederá a la implementación de la misma.

En este apartado se detallará como se ha realizado dicho proceso y, una vez terminado, se detallarán las pruebas software a realizar con el objetivo de comprobar la calidad de la implementación y de si se cumple el comportamiento esperado.

6.1. Implementación de la red Blockchain

Tal y como se ha ido anticipando en los apartados anteriores, para la implementación de la red Blockchain es necesario utilizar el programa nativo eris:cli, disponible para distribuciones Linux y posteriormente el software Docker.

Mediante eris:cli se crea la configuración de la red, cuántos nodos se requieren y qué tipos de cuentas van a ir asociadas a los mismos.

Tras haberla creado se genera un directorio con los ficheros necesarios para la configuración, en los que se encuentran las claves privadas y públicas de los nodos, las direcciones de los nodos y los ficheros para configurar los nodos para que puedan estar interconectados entre ellos.

Los ficheros obtenidos son los siguientes:

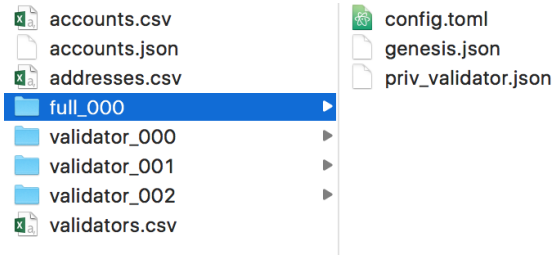


Ilustración 35: Ficheros de configuración de la red de Eris

Donde cada fichero es lo siguiente:

- **accounts.csv y accounts.json:** ficheros que contienen las claves privadas y públicas de la red Blockchain.
- **adresses.csv:** fichero que contiene las cuentas de los nodos.
- **Validators.csv:** fichero que contienen las direcciones de todos los nodos validadores.
- **Carpetas full_000, validator_000, validator_001, validator_002:** directorios que contienen los ficheros config.toml, donde se configura el node, priv_validator.json, fichero que contiene las claves del nodo, y genersis.json, fichero que contiene el primer bloque del nodo.

Una vez obtenidos dichos ficheros, es necesario cambiar en el fichero config.toml de los nodos validadores la ip y puerto donde se encuentra el nodo validador. Este cambio se realiza mediante el parámetro “seeds”:

```
moniker = "node"
seeds = "192.168.1.196:46656"
fast_sync = false
db_backend = "leveldb"
log_level = "debug"
node_laddr = "0.0.0.0:46656"
rpc_laddr = "0.0.0.0:46657"
vm_log = false
```

Ilustración 36: Configuración de un nodo

Realizados los cambios el siguiente paso es realizar un docker-compose para poder levantar tantos contenedores de Docker como nodos haya en la red.

Finalmente, una vez realizado se puede levantar la red mediante el comando:

```
$ docker-compose up -d
```

6.2. Implementación de Healthchain

Para la implementación de la aplicación web se han realizado tres directorios distintos:

- Back: aplicación en NodeJS que funciona como backend de la aplicación
- Front-1: aplicación en HTML5 y AngularJS que funciona como el frontend del paciente de la aplicación.
- Front-2: aplicación en HTML5 y AngularJS que funciona como el frontend de la aseguradora de la aplicación.

La estructuración de los ficheros de Front-1 y Front-2 es muy parecida. Inicialmente se comenzó realizando únicamente el Front-1 de forma estática y posteriormente se le añadió el controlador de AngularJS para comunicarse con el backend de la aplicación.

Finalmente, en la parte de la capa visual, como mejora a la aplicación se utilizó el software Gulp para automatizar su despliegue. Gulp es un sistema de construcción que permite la automatización de tareas relacionas con el desarrollo de aplicaciones.

Por el contrario, en la parte del backend es necesario destacar que se ha incluido un script de inicialización que inicializa el contendor de la base de datos, creando así las tablas necesarias.

Por otro lado, también se ha creado un fichero de configuración para conectarse con la red Blockchain e inicializarla cuando se despliega la aplicación.

6.3. Tecnologías utilizadas para la implementación

Tras haber hecho un estudio de la aplicación a implementar, se analizarán las tecnologías escogidas con este fin.

La tecnología principal que ya ha sido analizada, y que es el motor principal del presente proyecto, es Blockchain. Y tras haber hecho un análisis de plataformas candidatas implementadas, se tomó como decisión de escoger la plataforma creada por Monax Industries, Eris. Por esta razón, y por no quitar protagonismo al resto de tecnologías que han dado vida a la aplicación, se analizarán directamente.

Estas tecnologías no han sido meramente escogidas al azar. Sin embargo, cabe añadir que con el fin de adquirir nuevos conocimientos y en lenguaje formal "cambiar la rutina", sí han tenido mayor poder de decisión a la hora de ser escogidas.

Por esta razón se hará un breve estudio de Docker, AngularJS, NodeJS, Git, MongoDB y Rest HTTP.

6.3.1. Docker

El principal motivo sobre la elección de esta plataforma se basó en la implementación de la plataforma Eris.

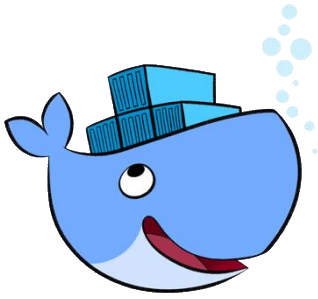


Ilustración 37: Logo de Docker

La plataforma puede ser montada en cualquier máquina de forma nativa gracias a eris:cli, componente que ya ha sido analizado. Sin embargo, a la hora de desplegar los nodos para que estén corriendo en una o más maquinas, es necesario esta tecnología.[5]

Docker es una plataforma utilizada para crear y ejecutar una aplicación en cualquier plataforma. Su código está implementado en el lenguaje de Google, Go. Docker es de tipo open source. Es muy utilizado por las empresas, por un lado, para simplificar y acelerar el proceso de desarrollo de aplicaciones y, por otro, el despliegue de las mismas. Esto es conseguido gracias a los contendores que son proporcionados gracias a Docker.

Los contenedores son ligeros y fácilmente intercambiables en función de la aplicación que les esté dando uso. Por otro lado, son interdependientes entre ellos y portables a través de los distintos entornos: desarrollo, pruebas, producción, etc.

Entre las características de Docker destacan las siguientes:

- Aislamiento entre aplicaciones
- Aislamiento entre la aplicación y su servidor.
- Portabilidad
- Fomentar la adopción del privilegio mínimo.

A modo paréntesis, el privilegio mínimo es una técnica proveniente de la seguridad mínima. Está basado en que un usuario, proceso o programa sólo puede ser capaz de acceder a los recursos mínimos que necesita para llevar a cabo su cometido.

En el núcleo de Docker se encuentra el Docker Engine, un motor de ejecución de aplicaciones. Este componente utiliza una arquitectura cliente-servidor. El cliente de Docker, aplicación instalada en la máquina, se comunica con el motor, el cual hace el trabajo pesado de construcción, transporte y ejecuta los contenedores de Docker para un servicio específico.

Las comunicaciones entre el cliente de Docker y el Docker engine se realizan via RESTFul, junto con una parte de seguridad que utiliza TLS (Seguridad de la capa de transporte).

El cliente de Docker puede ser ejecutado en cualquier sistema operativo: distribuciones de Linux, Windows, MacOS.

6.3.2. AngularJS

AngularJS[6] es un framework estructural de tipo open source desarrollado por Google. Permite la creación de páginas web dinámicas de tipo SPA (Single Page Applications) mediante el uso de JavaScript.

Ilustración 38: Logo de AngularJS

En el pasado, en la parte frontend del desarrollo de páginas web, el dinamismo era proporcionado mediante JavaScript nativo o JQuery. Asimismo, no se seguía ningún tipo de patrón, es decir, las funciones se implementaban en función a la demanda. Sin embargo, gracias a la aparición de AngularJS este problema quedaba solucionado.

AngularJS utiliza una inyección de dependencias propia que elimina gran parte de código que debería ser escrita por otro lado. De este modo, la parte cliente en una página web puede tener una parte servidora que funcionaría como backend. Consiguiendo así una aplicación de tipo SPA, como hemos comentado anteriormente.

Angular también permite a la aplicación seguir el modelo MVC (Modelo Vista Controlador), de esta forma el desarrollo y el testing se realiza de una forma sencilla y rápida.

AngularJS proporciona las siguientes ventajas a una aplicación:

- Directivas: son marcadores que indican al compilador que el elemento que los contienen poseen un comportamiento específico. Esto, asimismo, ofrece reusabilidad a partes de la implementación de la aplicación.
- Dos vías de enlace de datos: Todo cambio visual se realiza en tiempo real. De esta manera se evita que el desarrollador tenga que estar sincronizando continuamente los componentes vista y controlador.
- Una amplia comunidad: Hasta el momento, son miles de usuarios los que utilizan esta tecnología y la comunidad generada es bastante amplia. Asimismo, se cuenta con el respaldo de Google.

A pesar de las ventajas, AngularJS cuenta con un gran inconveniente: la seguridad. Si no es utilizado de manera correcta y no se tienen en cuenta los datos utilizados, la aplicación puede tener una brecha de seguridad. Esto es debido a que las llamadas se hacen desde la vista y pueden ser manipuladas o vistas desde la consola.

En la aplicación que ha sido desarrollada durante el proyecto, AngularJS funciona como la parte servidora de la aplicación y como enlace de comunicación entre la vista y el microservicio de backend.

6.3.3. NodeJS

Ilustración 39: Logo de NodeJS

NodeJS[7] es un entorno de ejecución y una biblioteca de librerías JavaScript que puede ser de tipo asíncrona y permite el manejo de eventos E/S.

En el pasado, JavaScript solo podía ser utilizado en la parte frontend para proporcionar dinamismo a las páginas HTML. Sin embargo, con la aparición de NodeJS, JavaScript también puede ser ejecutado en el lado servidor.

NodeJS fue desarrollado por Google, es de tipo open source y puede ser ejecutado en cualquier plataforma Windows, Linux o MacOS.

Este entorno proporciona las siguientes ventajas:

- Mayor experiencia de usuario: las aplicaciones desarrolladas son mucho más ligeras y rápidas.

- Disminución en costes de infraestructura. NodeJS requiere mucha menos infraestructura que cualquier otro lenguaje. Un ejemplo real es la infraestructura de Linkedin, la cual pasó a disminuir su número de servidores significativamente a 4.
- Los test unitarios se realizan de una forma más sencilla y rápida. Asimismo, hasta el momento hay múltiples librerías que permiten realizar el testing de aplicaciones.
- Permite la programación asíncrona de una forma sencilla. NodeJS posee múltiples formas que permiten incorporar funciones asíncronas a una aplicación: promesas, callbacks…
- Amplia librería. Existe una buena gestión y una grandísima variedad de paquetes NPM, los cuales permiten prácticamente poder hacer cualquier cosa en dicho lenguaje.
- Escalabilidad. Esta ventaja es debido a que NodeJS fue diseñado para ser utilizado en una arquitectura asíncrona, lo cual permite la escalabilidad de forma sencilla.
- Alto rendimiento, gracias al motor JavaScript V8.
- Está basada en el no-bloqueo, lo cual hace de NodeJS un entorno eficiente y ligero.

Esta tecnología ha sido escogida debido a su mayor funcionalidad: la programación asíncrona. Las llamadas que se realizan desde el backend a la plataforma Blockchain son de este tipo. Por otro lado, la plataforma Eris posee su librería npm oficial para comunicarse con la misma, lo cual simplificaba múchisimo el trabajo.

Por otro lado, con fines lectivos propios NodeJS es una tecnología muy utilizada en la actualidad y deseaba tener una toma de contacto con la misma.

6.3.4. Git

Ilustración 40: Logo de Git

Git[8] es un software utilizado para el control de versiones. Éste fue desarrollado por Linus Torvarlds en el año 2005 con el fin de mejorar la eficiencia y la confiabilidad al control de versiones aplicable tanto a aplicaciones pequeñas como a aplicaciones con un gran número de líneas y ficheros.

Inicialmente Git fue pensado como un motor de bajo nivel donde los usuarios escribirían el frontend de una aplicación. Sin embargo, desde entonces, Git se ha convertido en un sistema de control de versiones.

El control de versiones es un sistema que registra los cambios sobre un archivo o conjunto de archivos, de este modo se pueden recuperar versiones específicas más adelante.

Los cambios se pueden apreciar basándose entre las similitudes entre los archivos, de modo que los renombrados de los ficheros se realizan sin el menor peligro.

Las ventajas que presenta Git son las siguientes:

- Compartición selectiva: Git permite seleccionar a un usuario qué ficheros o conjunto de ficheros quiere compartir.
- Velocidad de acceso y de escritura.
- Ramificación: se pueden hacer diferentes ramas sobre un proyecto para poder así abarcar distintos aspectos del mismo de forma independiente.
- Flujo de trabajo adaptable, ya que permite el desarrollo ágil.
- Disminución de coste: en la actualidad hay múltiples softwares de tipo open source que permiten la manipulación del control de versiones.
- Gran rendimiento: las búsquedas entre los ficheros son muy eficaces lo cual favorece la velocidad de las mismas.
- Recuperación de ficheros. En ocasiones, durante el desarrollo, hay veces que se requiere volver a un paso anterior. Con Git es posible.
- Gestión distribuida. Git permite tener copias tanto en repositorios locales como en repositorios en la nube, permitiendo así la generación automática de seguridad del proyecto.

Como esta tecnología es proporcionada en Bitbucket, se utilizará un repositorio de dicha plataforma para realizar un control de versiones sobre el proyecto.

6.3.5. MongoDB

MongoDB[9] es la base de datos NoSQL más utilizada en el mercado. Ofrece escalabilidad, rendimiento, disminución de costes, crear nuevos tipos de aplicaciones y mejorar la experiencia del usuario. Es, por un lado, de tipo open-source y, por otro, posee licencias comerciales para su parte MongoDB Enterprise.

Ilustración 41: Logo de MongoDB

Al ser una base de datos NoSQL, los datos no se almacenan en registros si no que se almacenan en documentos en formato BSON, formato binario del estándar JSON.

MongoDB está implementado en C++, sin embargo, las consultas que se quieran realizar a la base de datos se realizan introduciendo un objeto de tipo JSON como parámetro.

A la hora de utilizar MongoDB se puede utilizar mediante su consola construida mediante JavaScript, o bien utilizar las librerías que hay implementadas para los siguientes lenguajes: C#, Java, Node.js, PHP, Python, Ruby, C, C++, Perl o Scala.

Las principales características de MongoDB son las siguientes:

- Están permitidas las consultas de tipo Ad hoc, es decir, búsquedas que incluyen expresiones regulares, búsqueda por campos y consultas de rangos.
- Indexación: cualquier campo en el documento almacenado en la base de datos puede estar indexado, lo cual reduce los tiempos de búsquedas y permite la escalabilidad.
- Balanceo de carga: el servidor de MongoDB divide de manera equitativa las peticiones que se realizan sobre la base de datos, evitando así los cuellos de botella. La base de datos puede ser escalada de forma horizontal mediante una clave, la cual determina la distribución de los datos en una colección.
- Replicación: MongoDB posee un tipo de replicación primario-secundario. Este tipo de replicación proporciona una alta disponibilidad y tolerancia a fallos de forma nativa. Asimismo, se basa en que hay una serie de instancias de MongoDB donde siempre hay un nodo primario y dos secundarios, como mínimo. Tras el fallo del nodo primario, se inicia un proceso de selección que busca entre los nodos restantes un sustituto que restituya las funcionalidades del primer nodo.

La elección de esta tecnología se ha basado en que los datos manejados, sin contar el historial médico que es almacenado en la red Blockchain, no tienen una estructura definida. Por ello, utilizar una base de datos de tipo SQL no habría sido óptimo para el desarrollo del proyecto.

6.3.6. REST

REST[10] es un estilo de arquitectura software dirigida a sistemas hipermedia distribuidos, sistemas que integran texto, imagen, video, audio, mapas y otros soportes de integración emergentes.

REST son las siglas de *REpresentational State Transfer*, es decir, transferencia de representación del estado. Esto quiere decir que un servicio que hace uso de esto es de tipo *stateless*, sin estado. Esto quiere decir que, si se hacen distintas peticiones, éstas son interdependientes y no guardan ningún tipo de relación.

REST cambió el enfoque de la ingeniería en el año 2000, ya que supuso una modificación en el enfoque de proyectos y produjo la aparición de los servicios web.

En la actualidad, REST es el estándar más utilizado, más lógico y más eficiente en la práctica del desarrollo de aplicaciones o de APIs.

Esta tecnología ha sido empleada como comunicación entre el microservicio del frontend con el del backend, proporcionando así escalabilidad entre ambas partes.

Esta escalabilidad ha sido conseguida debido a los siguientes aspectos de diseño:

- El protocolo cliente/servidor es de tipo sin estado. Debido a ello, no es necesario ningún tipo de almacenamiento en las distintas partes para seguir un registro de mensajes.
- Las operaciones están bien definidas. Las operaciones realizadas son las siguientes: GET (leer y consultar), PUT (editar), DELETE (borrar) y POST (crear).
- Se utiliza una estructura de capas, de manera que se sigue una estructura jerárquica entre los distintos componentes.

6.4. Pruebas realizadas

Las pruebas de software tienen como principal objetivo la comprobación de que el comportamiento de un programa, mediante un conjunto de casos de prueba, cumple con el resultado que se esperaba. De esta manera se puede obtener una información objetivo sobre el estado de la calidad de un producto.

De esta manera, en el presente apartado se detallarán las pruebas realizadas al software y cómo fue su respuesta.

Por ello, al igual que con la especificación de los requisitos, se ha realizado una plantilla que detallará las pruebas realizadas. La plantilla es la siguiente:

IDENTIFICADOR CP-XX

TÍTULO	
DESCRIPCIÓN	
RESULTADO ESPERADO	
DEPENDENCIAS	

Tabla 44: Casos de prueba

Donde los campos tienen el siguiente significado:

- Identificador CP-XX: Identificador que identifica unívocamente al caso de prueba. Formado por las siglas CP (Caso de Prueba) y XX que se refiere al número de caso de prueba.
- Título: Título del caso de prueba.
- Descripción: Detalle del caso de prueba a realizar.
- Resultado esperado: Resultado que se obtiene al realizar la prueba.

Dependencia: Requisito funcional o no funcional relacionado con el caso de prueba.

IDENTIFICADOR CP-01

TÍTULO	Login del paciente
DESCRIPCIÓN	Una vez que se ha iniciado la aplicación el usuario paciente tiene que introducir sus credenciales de acceso.
RESULTADO ESPERADO	El login se realiza correcto y se redirige a la página de inicio de la aplicación.
DEPENDENCIAS	RF-08, RNF-06

Tabla 45: CP-01 Login del paciente

IDENTIFICADOR CP-02

TÍTULO	Recuperación de los datos
DESCRIPCIÓN	Una vez que el usuario accede a la página inicial puede acceder a sus datos médicos.
RESULTADO ESPERADO	Se muestra el historial médico electrónico del usuario paciente.
DEPENDENCIAS	RF-01, RF-03

Tabla 46: CP-02 Recuperación de los datos

IDENTIFICADOR CP-03

TÍTULO	Introducción de nuevos datos
DESCRIPCIÓN	Se comprobará que los datos almacenados son recuperados posteriormente.
RESULTADO ESPERADO	El usuario tras haber accedido a la página de introducir datos volverá a la página de inicio y podrá ver su historial médico electrónico actualizado.
DEPENDENCIAS	RF-02

Tabla 47: CP-03 Introducción de nuevos datos

IDENTIFICADOR CP-04

TÍTULO	Búsqueda de enfermedad
DESCRIPCIÓN	Se comprobará que aparecen las enfermedades relacionadas al introducir una en el campo se búsqueda.
RESULTADO ESPERADO	El usuario tras haber añadido en el campo de búsqueda una enfermedad encontrarán las enfermedades relacionadas con sus correspondientes códigos.
DEPENDENCIAS	RF-10

Tabla 48: CP-04 Búsqueda de enfermedad

IDENTIFICADOR CP-05

TÍTULO	Búsqueda de historial médico electrónico.
DESCRIPCIÓN	Comprobación de la búsqueda de un historial médico electrónico por parte del usuario aseguradora.
RESULTADO ESPERADO	El usuario de tipo aseguradora introducirá en su frontend correspondiente la dirección de un historial médico electrónico y se mostrarán los datos médicos asociados al mismo.
DEPENDENCIAS	RF-05

Tabla 49: CP-05 Búsqueda del historial médico electrónico

IDENTIFICADOR CP-06

TÍTULO	Nodos validadores
DESCRIPCIÓN	Se comprobará que aparecen las direcciones de los nodos validadores.
RESULTADO ESPERADO	El usuario accederá al panel de control de la red Blockchain y observará que aparecen las direcciones de los nodos validadores.
DEPENDENCIAS	RF-07

Tabla 50: CP-06 Nodos validadores

IDENTIFICADOR CP-07

TÍTULO	Adaptación de la interfaz.
DESCRIPCIÓN	Se comprobará que la interfaz es adaptativa.
RESULTADO ESPERADO	Se modificará el tamaño de la ventana y se observará que la interfaz se adapta al mismo.
DEPENDENCIAS	RNF-02

Tabla 51: CP-07 Adaptación de la interfaz

IDENTIFICADOR CP-08

TÍTULO	Tamaño de la cadena
DESCRIPCIÓN	Se comprobará que se muestra el número de bloques validados en la cadena.
RESULTADO ESPERADO	Se accederá a la pantalla de control de la red Blockchain y tras varias actualizaciones de la misma se observará que el tamaño de bloque es mayor al anterior.
DEPENDENCIAS	RF-06

Tabla 52: CP-08 Tamaño de la cadena

IDENTIFICADOR CP-09

TÍTULO	Estado de la red
DESCRIPCIÓN	Se comprobará que la red está realizando consenso.
RESULTADO ESPERADO	Se accederá al puerto correspondiente de la red y se observará que los nodos están conectados entre ellos.
DEPENDENCIAS	RF-07

Tabla 53: CP-09 Estado de la red

IDENTIFICADOR CP-10

TÍTULO	Tolerancia a fallos
DESCRIPCIÓN	Se comprobará que la red al caerse un nodo sigue funcionando.
RESULTADO ESPERADO	Se parará un contenedor de bloque y se comprobará que el tamaño de la cadena sigue aumentando.
DEPENDENCIAS	RNF-14

Tabla 54: CP-10 Tolerancia a fallos

IDENTIFICADOR CP-10

TÍTULO	Replicación del historial médico
DESCRIPCIÓN	Se comprobará que el historial médico se encuentra replicado.
RESULTADO ESPERADO	Se accederá a al menos tres nodos de la red y se obtendrá que el historial médico electrónico en cada uno de ellos.
DEPENDENCIAS	RNF-4

Tabla 55: CP-10 Replicación del historial médico

IDENTIFICADOR CP-11

TÍTULO	Mensajes de error
DESCRIPCIÓN	Se comprobará que aparecen mensajes de error.
RESULTADO ESPERADO	Se cometerá un error que pueda cometer el usuario y se observará que aparece un diálogo de error.
DEPENDENCIAS	RNF-4

Tabla 56: CP-11 Mensajes de error

Una vez definidos los casos de prueba, ser realizarán las mismas sobre el comportamiento de la aplicación, donde los resultados generados son los siguientes:

CASO DE PRUEBA	RESULTADO
CP-01	ÉXITO
CP-02	ÉXITO
CP-03	ÉXITO
CP-04	ÉXITO
CP-05	ÉXITO
CP-06	ÉXITO
CP-07	ÉXITO
CP-08	ÉXITO
CP-09	ÉXITO
CP-10	ÉXITO

Tabla 57: Resultados casos de prueba

7. Marco regulador y entorno socioeconómico

En este apartado se realizará un breve análisis sobre la ley aplicable y marco regulador a los datos y servicios utilizados a lo largo del proyecto y a los datos que se tratarán en el caso de que la aplicación llegue a entornos de producción. Esto es debido a que el tipo de los datos es de carácter muy confidencial.

Por otro lado, se realizará un estudio sobre el impacto socioeconómico que supone Blockchain en el mundo que hasta ahora conocemos. Como hemos ido viendo a lo largo del documento, Blockchain es una tecnología candidata a hacer un giro de trescientos sesenta grados sobre la concepción de la tecnología y su uso hasta el momento.

7.1. Marco regulador y aspectos legales

En cuanto a los servicios utilizados, la plataforma Blockchain Eris, se han tenido en cuenta los aspectos legales de la misma con el fin de no disuadir la legalidad vigente. Asimismo, se afirma que los recursos que han sido utilizados, han sido de manera que no incumplen las leyes ni la normativa pertinentes a los mismos.

Tal y como se ha comentado, la plataforma es de código abierto y se permite su uso bajo la licencia General Public License 3.0. Dicha licencia está referida al software libre y fue creada por el GNU con el fin de promover el software de manera gratuita.[26]

Gracias a este tipo de licencia, cualquier usuario que haga uso del producto puede realizar cualquier modificación sobre el mismo siempre y cuando mantenga la licencia del mismo, es decir, el usuario no puede realizar modificaciones sobre el código fuente y lanzarlo al mercado de forma lucrativa.

Por otro lado, los datos que serán explotados en la aplicación y que, posteriormente, son almacenados en la red Blockchain son de carácter personal y relacionados con el estado de salud de un usuario, por lo que son altamente confidenciales.[27]

La protección de los datos personales es un derecho constitucional que aparece en el Artículo 18.4 de la Constitución Española y se presenta de la siguiente manera:

*"La ley **limitará el uso de la informática** para garantizar el honor y la **intimidad personal** y familiar de los ciudadanos y el pleno ejercicio de sus derechos"* .[27]

La legislación que se debería aplicar con el fin de mantener los datos sin una exposición pública sería la LOPD (Ley Orgánica de Protección de Datos de Carácter Personal). Ley Orgánica tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas.

Asimismo, se persigue el compromiso de poder dar a una persona el pleno control sobre los datos personales, su uso y su destino. De esta manera se podría impedir el tráfico ilícito y lesivo sobre los derechos de la persona afectada.

A la hora de desarrollar el proyecto este problema se ha tenido muy en cuenta puesto que la confidencialidad de los datos del usuario es altamente necesaria. Por ello, los datos están cifrados cuando son almacenados en las redes Blockchain por defecto y sólo el usuario decide que datos y con quién decide compartirlos como, por ejemplo, una institución sanitaria.

7.2. Análisis socioeconómico de Blockchain como tecnología

En el pasado, toda la información relativa a las transacciones y al tratamiento de bienes quedaba almacenada en libros de registros físicos e independientes entre ellos. [28]

Poco a poco, con la llegada del mundo digital y con el avance inminente de la tecnología, estos libros físicos fueron migrados a libros digitales. Sin embargo, la información no poseía ningún tipo de relación de dependencia de forma interna, y mucho menos de forma externa. Hasta el momento, toda información financiera sigue siendo mantenida en libros de registros. Sin embargo, a este tipo de digitalización de la información le queda poco tiempo de vida con la llegada de Blockchain.

Blockchain fue analizado en el foro económico mundial, realizado en 2016, dejándolo como una de las tecnologías más potentes y disruptivas hasta el momento. Por ello, Blockchain está visto como una tecnología que va a cambiar en el mundo la forma de realizar negocios.

Este cambio es muy importante ya que mejorará el coste y la seguridad en los negocios, partes altamente impactantes en los mismos.

El foro económico mundial tiene como objetivo principal aplicar un enfoque multisectorial para tratar temas que tengan un alto impacto global. Este enfoque se consigue con las participaciones con las diferentes comunidades: financiera, tecnológica, de innovación y el sector público, entre otras.

En el año 2015 tuvieron como proyecto futuro hacer una investigación sobre las innovaciones disruptivas en los servicios financieros hasta el momento, analizando así sus posibles impactos en la industria.

El estudio se realizó durante 12 meses teniendo como participantes líderes y expertos en la materia. Entre estos participantes encontramos en el sector empresarial: Deloitte, Zurich, ING Bank, Banco Santander, y en el sector tecnológico: Ripple, Hyperledger Project, Eris, Blockchain...etc. Tras estas investigaciones llegaron a la conclusión de que Blockchain abarcaba el ecosistema de los servicios financieros y que tenía un grandísimo potencial para hacer un cambio en la concepción de la economía hasta el momento de forma positiva.

Utilizando Blockchain, los procesos financieros y sus infraestructuras se simplificarán y ganarán mucha más eficiencia. Según el foro mundial de la economía, esto es debido a:

- Minimización del fraude. Debido a que es prácticamente alterar los bloques en la red y todo se realiza de forma transparente
- Reducción del riesgo entre tener partes involucradas. Blockchain es un sistema que no requiere confianza
- Simplificación operacional. Usando esta tecnología se eliminan los esfuerzos manuales para resolver problemas.
- Reducción de tiempo. Blockchain realiza las transacciones en tiempo real, de modo que, si se requiere de hacer una transferencia multinacional, ésta se realizará de manera inmediata y no en cuestión de días, como hasta el momento.
- Refuerzo del perfil digital. Al estar almacenada la identidad digital en un sistema Blockchain, se establece la confianza y la autenticación se realiza de una manera mucho más sencilla a la convencional de hoy en día, y es vulnerable.
- Mejora de la liquidez y el capital de las instituciones financieras. Blockchain reduce el bloque de capitales y ofrece transparencia en el abastecimiento de liquidez para los activos.

Aun así, Blockchain no tiene por qué estar orientado únicamente a productos financieros, sino productos que de los cuales dependen los mismos. Por ejemplo, las identidades digitales. Hasta el momento, el tema de la identidad digital es algo complejo y depende de muchas instituciones y terceras partes que por obligación requieren de la confianza. Utilizando esta tecnología se eliminarían las terceras partes y se haría de un sistema dependiente de la confianza, uno que no la requiere. Muchas empresas y startups se están dedicando al desarrollo de una infraestructura que lo soporte, como por ejemplo KYC-Chain.

Las aplicaciones relacionadas con Blockchain requieren una profunda colaboración entre titulares de activos, innovadores tecnológico y reguladores de mercado, añadiendo así una complejidad técnica alta.

Entre el amplio rango de casos de uso de las redes Blockchain, el foro económico mundial determinó los siguientes casos de usos orientados al sector financiero reflejados en la siguiente imagen.

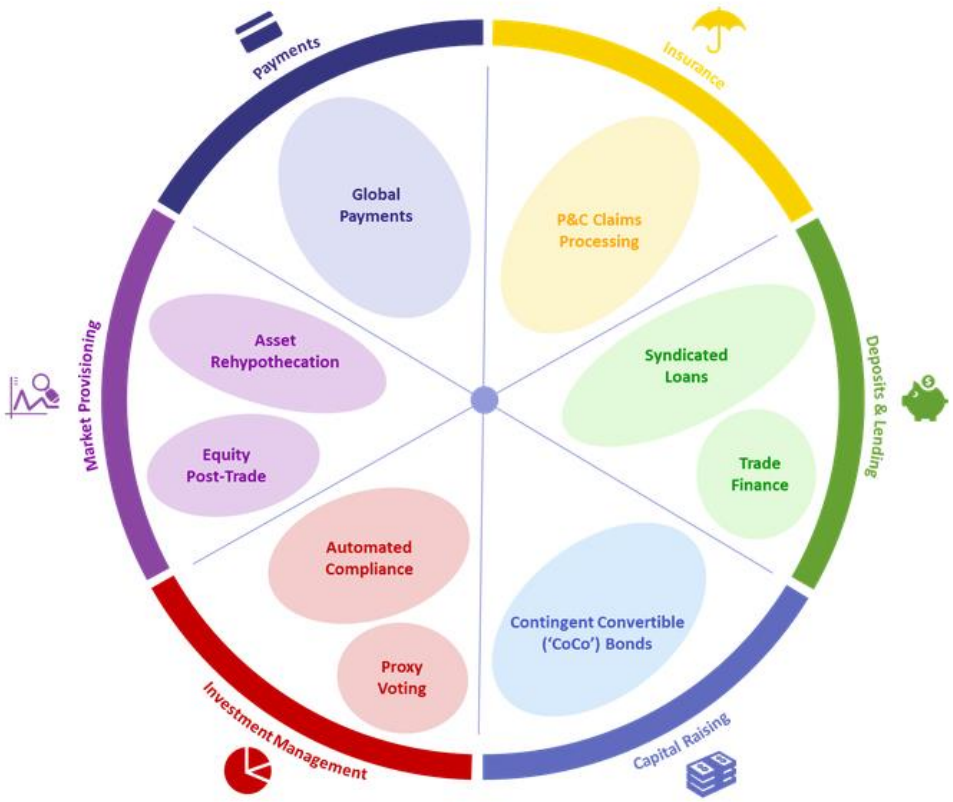


Ilustración 47: Casos de uso financieros

8. Planificación y presupuesto

La planificación antes de la realización de un proyecto cobra una gran relevancia debido a que en este proceso se pueden conocer las limitaciones y se puede hacer una estimación de la duración del mismo teniendo en cuenta los posibles retrasos que puedan suceder.

La estimación del proyecto debe realizarse teniendo en cuenta los plazos, la disponibilidad de los recursos, ya sean de personal o económicos.

El propósito principal de realizar una planificación de un proyecto software es realizar una estimación razonable de tiempo y de los recursos que se encontrarán implicados en el proyecto. De este modo, si ocurre cualquier imprevisto, los repercusiones e impacto que pueda suceder sobre el proyecto sea el mínimo necesario.

A continuación, en los próximos dos apartados, se realizará la planificación del proyecto y se calculará el presupuesto. Este cálculo de presupuesto del proyecto software se realizará en función de los datos obtenidos en la estimación.

8.1. Planificación del proyecto software

Para realizar la tarea de planificación se utilizará la técnica de los diagramas de Gantt. Un diagrama de Gantt es una herramienta gráfica en la que se realiza una exposición del tiempo que se supone que se va a emplear para realizar tareas que pertenecen al proyecto. Asimismo, este tipo de diagramas están fijados en un marco temporal, que se deberá ir revisando según el avance del proyecto.

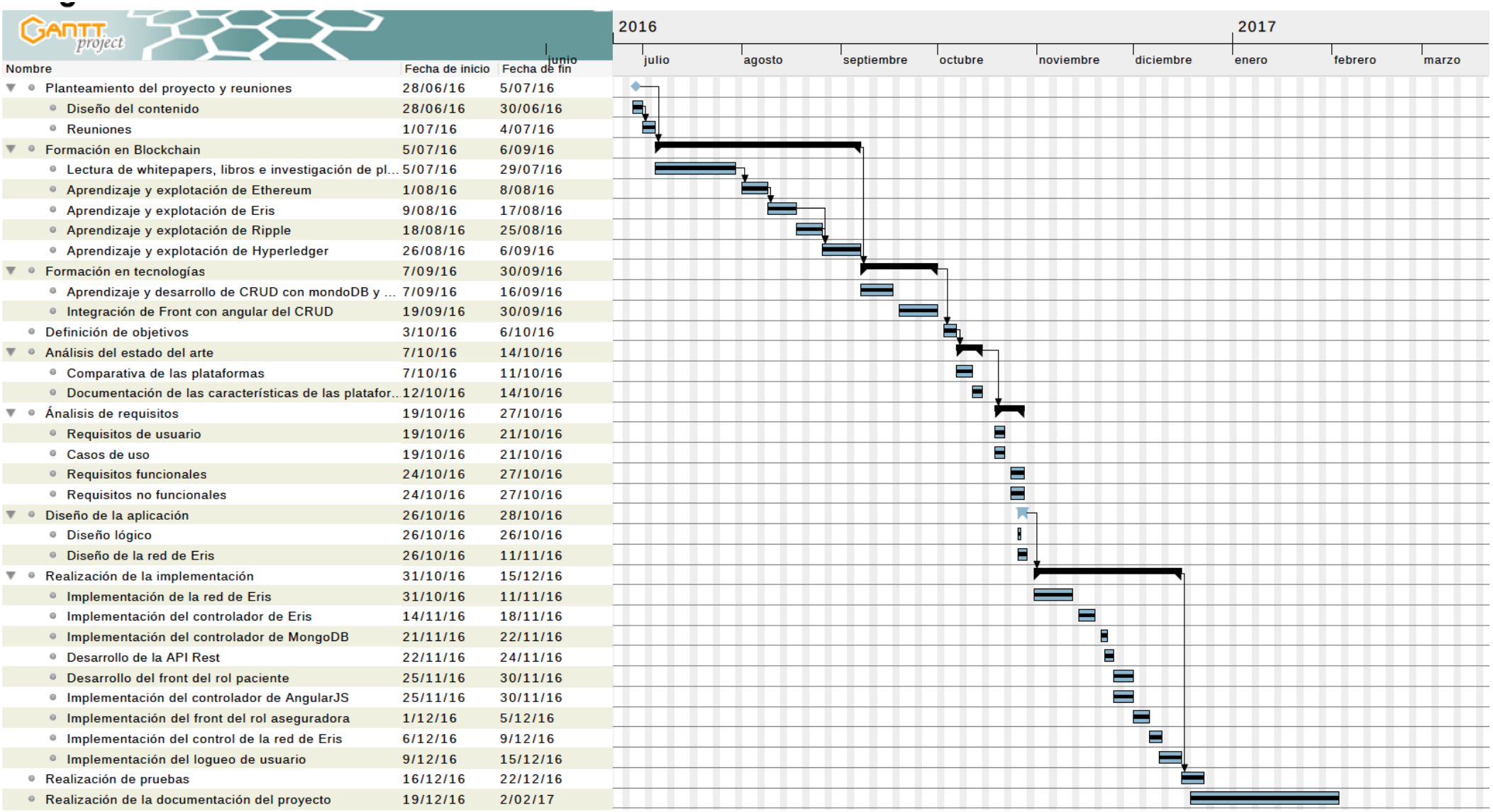
Antes de realizar el diagrama de Gantt, es de gran importancia definir las tareas que se irán realizando a lo largo del proyecto. Tras la definición de las tareas, se deben definir los tiempos realistas de ejecución de las mismas, la prioridad y su orden. En el campo de la gestión de proyectos, los diagramas de Gantt son considerados herramientas bastante útiles y eficaces. Esto es debido, por un lado, a que explica de forma visual algo que sería muy costoso de decir mediante palabras, y por otro, por su sencillez a la hora de analizar las actividades a realizar, las dependencias entre las mismas y su temporalización.

Las tareas que se han indicado para llevar acabo el presente proyecto son las siguientes:

1. **Planteamiento del proyecto y reuniones**
    - Diseño del contenido.
    - Reuniones.

2. **Formación en Blockchain**
    - Lectura de whitepapers, libros e investigación de plataformas.
    - Aprendizaje y explotación de Ethereum.
    - Aprendizaje y explotación de Eris.
    - Aprendizaje y explotación de Ripple.
    - Aprendizaje y explotación de Hyperledger.
3. **Formación en tecnologías**
    - Aprendizaje y desarrollo de CRUD con MongoDB y NodeJS.
    - Integración de Front con angular del CRUD.
4. **Definición de objetivos**
5. **Análisis del estado del arte**
    - Comparativa de las plataformas.
    - Documentación de las características de las plataformas.
6. **Análisis de requisitos**
    - Requisitos de usuario.
    - Casos de uso.
    - Requisitos funcionales.
    - Requisitos no funcionales.
7. **Diseño de la aplicación**
    - Diseño lógico.
    - Diseño de la red de Eris.
8. **Realización de la implementación**
    - Implementación de la red de Eris.
    - Implementación del controlador de Eris.
    - Implementación del controlador de MongoDB.
    - Desarrollo de la API Rest.
    - Desarrollo del front del rol paciente.
    - Implementación del controlador de AngularJS.
    - Desarrollo del front de la aseguradora.
    - Implementación del control de la red de Eris.
    - Implementación del logueo de usuario.
9. **Realización de pruebas**
10. **Realización de la documentación del proyecto**

En la siguiente página se muestra el diagrama de Gantt estimando las fechas de las tareas.



Nombre	Fecha de inicio	Fecha de fin
Planteamiento del proyecto y reuniones	28/06/16	5/07/16
Diseño del contenido	28/06/16	30/06/16
Reuniones	1/07/16	4/07/16
Formación en Blockchain	5/07/16	6/09/16
Lectura de whitepapers, libros e investigación de pl...	5/07/16	29/07/16
Aprendizaje y explotación de Ethereum	1/08/16	8/08/16
Aprendizaje y explotación de Eris	9/08/16	17/08/16
Aprendizaje y explotación de Ripple	18/08/16	25/08/16
Aprendizaje y explotación de Hyperledger	26/08/16	6/09/16
Formación en tecnologías	7/09/16	30/09/16
Aprendizaje y desarrollo de CRUD con mondoDB y ...	7/09/16	16/09/16
Integración de Front con angular del CRUD	19/09/16	30/09/16
Definición de objetivos	3/10/16	6/10/16
Análisis del estado del arte	7/10/16	14/10/16
Comparativa de las plataformas	7/10/16	11/10/16
Documentación de las características de las platafor...	12/10/16	14/10/16
Ánalisis de requisitos	19/10/16	27/10/16
Requisitos de usuario	19/10/16	21/10/16
Casos de uso	19/10/16	21/10/16
Requisitos funcionales	24/10/16	27/10/16
Requisitos no funcionales	24/10/16	27/10/16
Diseño de la aplicación	26/10/16	28/10/16
Diseño lógico	26/10/16	26/10/16
Diseño de la red de Eris	26/10/16	11/11/16
Realización de la implementación	31/10/16	15/12/16
Implementación de la red de Eris	31/10/16	11/11/16
Implementación del controlador de Eris	14/11/16	18/11/16
Implementación del controlador de MongoDB	21/11/16	22/11/16
Desarrollo de la API Rest	22/11/16	24/11/16
Desarrollo del front del rol paciente	25/11/16	30/11/16
Implementación del controlador de AngularJS	25/11/16	30/11/16
Implementación del front del rol aseguradora	1/12/16	5/12/16
Implementación del control de la red de Eris	6/12/16	9/12/16
Implementación del logueo de usuario	9/12/16	15/12/16
Realización de pruebas	16/12/16	22/12/16
Realización de la documentación del proyecto	19/12/16	2/02/17

Ilustración 48: Diagrama de Gantt

Si observamos las fechas del diagrama de Gantt las fechas de inicio y fin del proyecto son 28 de junio de 2016 y 3 de febrero de 2017 respectivamente. Haciendo un análisis total de los días empleados, se genera la suma de 185 (ciento ochenta y cinco) días, contando días laborables de lunes a viernes.

En la siguiente tabla se muestra un desglose de los hitos de las tareas, el número de días empleados, las horas estimadas y las horas empleadas.

TAREAS	DIAS ESTIMADOS	HORAS ESTIMADAS	HORAS REALES
PLANTEAMIENTO DEL PROYECTO Y REUNIONES	8	18	18
FORMACIÓN EN BLOCKCHAIN	49	147	172
FORMACIÓN EN TECNOLOGÍAS DE DESARROLLO	20	60	50
DEFINICIÓN DE OBJETIVOS	4	20	15
ANÁLISIS DEL ARTE	10	40	38
ÁNALISIS	9	30	25
IMPLEMENTACIÓN	48	240	264
REALIZACIÓN DE PRUEBAS	5	30	42
REALIZACIÓN DE LA DOCUMENTACIÓN DEL PROYECTO	34	102	85
TOTAL	185	687	709

Tabla 58: Planificación de horas y días

Los plazos establecidos en la planificación referidos a la realización de las tareas han sido cumplidos sin ningún tipo de retraso. Sin embargo, cabe mencionar que hay se realizó una estimación de las horas distinta a la que ha sido cumplida.

Si observamos la tabla, el número de horas estimadas fue de 687 (seiscientas ochenta y siete) horas, y las horas que han sido empleadas en el proyecto ha sido de 709 (setecientas nueve) horas. Esta desviación fue debido al aprendizaje de la nueva tecnología, debido a que la documentación que hay es bastante escasa y se tuvo que hacer un procedimiento de investigación.

Por ello, cabe destacar que la desviación ha supuesto un aumento del 3,2% del esfuerzo dedicado a la realización del presente proyecto.

Las tareas críticas que han tenido mayor impacto en el proyecto, tal y como se acaba de comentar, son todas las referidas a la tarea “Formación en Blockchain”. Actualmente, hay poca documentación a la hora de realizar pruebas de concepto sobre una determinada plataforma y el proceso de investigación está prácticamente pasado en la técnica ensayo y error.

Finalmente, no se ha tenido en cuenta en ningún momento la realización de la tarea de la presentación del proyecto. Esto es debido a que la misma se encuentra fuera del plazo de entrega de la presente documentación del proyecto y no se ha creído relevante a la hora de ser introducida en el diagrama de Gantt.

8.2. Presupuesto del proyecto

A continuación, se realizará la estimación del coste total del proyecto en función de los datos adquiridos en el apartado anterior. De este modo se van a tener en cuenta los siguientes tipos de gastos: de personal, de material empleado e indirectos.

8.2.1. Costes de personal

Los recursos implicados en el presente proyectos han sido los siguientes:

María Teresa Nieto Galán: Desarrolladora, arquitecta y analista de la aplicación web y autora del presente documento.

José María Álvarez Rodríguez: Tutor del proyecto

Cabe destacar, debido a que el tutor del proyecto tiene como función orientar el presente trabajo y realizar reuniones para solventar dudas, no se ha tenido cuenta a la hora en los costes de personal. Por lo tanto, solo se tendrá cuenta como recurso a María Teresa Nieto Galán.

En la siguiente tabla se muestra el coste de personal de dicho recurso. Para la realización de los cálculos se ha tenido en cuenta la tabla del apartado anterior.

PERSONAL	CARGO	COSTE POR HORA (€/H)	NUMERO TOTAL DE HORAS EMPLEADAS(H)	COSTE (€)
MARÍA TERESA NIETO GALÁN	Informático Junior	20	709	14180
			TOTAL	14.180

Tabla 59: Costes de personal

El coste se ha calculado estimando una media de 20€/h debido a que se ha considerado más apropiado realizar una estimación por hora empleada.

8.2.2. Costes de material implicado

A la hora de hacer un cálculo de los costes del material implicado en el proyecto hay que hacer dos diferenciaciones, costes hardware y costes software. Los primeros están relacionados con la máquina en la que se ha desarrollado el proyecto y los segundos se relacionan con las herramientas empleadas para llevarlo a cabo.

Coste Hardware

El proyecto ha sido realizado con la máquina propia de la autora y desarrolladora del mismo, el cual tiene las siguientes características.

- Fabricante: Apple
- Modelo: MacBook Pro (Retina 13 pulgadas, principios de 2015)
- Procesador: 2,7 GHz Intel Core i5
- Memoria: 8 GB 1867 MHz DDR3
- Gráficos: Intel Iris Graphics 6100 1536 MB
- Coste: 1449 (mil cuatrocientos cuarenta y nueve) euros.

Teniendo en cuenta de que la máquina ha sido utilizada durante 8 meses aproximadamente, se tendrá en cuenta la amortización de la misma durante dicho periodo. Suponiendo que la amortización de un ordenador es de 48 meses.

$$Coste\ Hardware = 1449 * \frac{8}{48} = 241{,}5\ euros$$

Coste Software

Las herramientas utilizadas para el desarrollo de la aplicación han sido de tipo open source. Debido a ello, los costes software relacionados a las mismas hacen un total de 0 (cero) euros.

Sin embargo, a la hora de realizar la documentación del proyecto la herramienta utilizada ha sido Microsoft Office 365, con un coste de 69 (sesenta y nueve) euros al año. Por lo tanto, teniendo en cuenta el uso de la herramienta durante el periodo del desarrollo del proyecto, se calcula lo siguiente:

$$Coste\ Software = \frac{69}{36} * 2 = 3{,}83\ euros$$

Para el cálculo se ha tenido en cuenta que la herramienta sólo ha sido usada en total durante dos meses y que el periodo de amortización de la misma es de 36 meses.

En la siguiente tabla se muestran los costes de material:

MATERIAL	TIPO	PRECIO (€)	Nº DE MESES EMPLEADO	PERIODO DE AMORTIZACIÓN (MESES)	COSTE (€)
MACBOOK PRO 2015	Hardware	1449	8	48	241,5
MICROSOFT OFFICE 365 PERSONAL	Software	69	2	36	3,83
				TOTAL	245,33

Tabla 60: Costes de material

Costes indirectos

En cuanto a los costes indirectos se ha tenido en cuenta lo siguiente:

COSTE INDIRECTO	COSTE MENSUAL (€)	Nº DE MESES EMPLEADO	COSTE (€)
ADSL Y LUZ	45	8	360
		TOTAL	360

Tabla 61: Costes indirectos

Coste total del proyecto

Finalmente, la tabla de costes totales teniendo en cuenta los costes anteriores es la siguiente:

TIPO DE COSTE	COSTE (€)
COSTES DE PERSONAL	14.180
COSTES DE MATERIAL	245,33
COSTES INDIRECTOS	360
IMPORTE TOTAL SIN BENEFICIO	**14.785,33**
BENEFICIO ESPERADO (20%)	2.957,07
IMPORTE TOTAL CON BENEFICIO	**17.742,4**
IVA (21%)	3.725,91
TOTAL	**21.468,31**

Tabla 62: Coste total

Como podemos observar en la tabla el coste del proyecto es de **21.468,31€ (veintiún mil cuatrocientos sesenta y ocho euros con treinta y uno céntimos).**

9. Conclusiones y trabajo futuro

La aplicación diseñada puede marcar un antes y un después en la gestión de los historiales médicos. El sector de la administración de la sanidad ha tenido una gran evolución en los últimos años, desde la gestión física hasta la gestión digital.

El problema hasta el momento es que el paciente no tiene acceso sobre su propio historial, únicamente es gestionado por los médicos en España y por los mismos y las aseguradoras en otros países como en Estados Unidos.

Con el diseño y uso de esta aplicación el historial médico pertenecería al paciente y todos los cambios que se realizarán sobre el mismo se realizarán de forma transparente e íntegra.

El desarrollo de este proyecto nació como idea de realizar una aplicación con un caso de uso real utilizando una tecnología nueva que tiene previsto un gran futuro, Blockchain.

Tal y como se ha ido viendo a lo largo del proyecto, Blockchain es una tecnología muy joven que se encuentra en un periodo de desarrollo e investigación. Hasta la fecha ya hay plataformas Blockchain que ya se encuentran en entornos de producción de forma estable, sin embargo, hay otras que están en periodo de nacimiento y sólo se encuentran en modo de desarrollo, por lo que la información y lo que hay desarrollado hasta el momento puede ser bastante cambiante.

Sin embargo, Blockchain va a ser toda una revolución como fue Internet en el momento de su nacimiento. Todas las características que hemos ido analizando a lo largo de este proyecto son características muy necesarias en el mundo de la informática, como la seguridad, la integridad, la transparencia y el bajo coste.

Por otro lado, a la hora de la realización de este proyecto se han tenido que invertir muchas horas de aprendizaje. Principalmente, estos conocimientos son bastante nuevos y van acompañados de tecnologías bastante nuevas y que son muy utilizadas en la actualidad.

Asimismo, cabe destacar la escasa información que puede haber en ocasiones a la hora de realizar este tipo de labor de investigación. Son muchas las plataformas que, al encontrarse en estado de desarrollo, como por ejemplo Hyperledger, tienen implementación realizada pero muy poca documentación, por lo que es bastante laborioso intentar comprender el funcionamiento de las mismas.

Finalmente, en cuanto a trabajo futuro pueden realizarse múltiples laboras de mejora de la aplicación. Estas mejoras son las siguientes:

Hasta ahora el historial médico es almacenado en un smart contract, lo cual, si el historial tiene un formato muy complejo, a la larga podría haber problemas de rendimiento. Actualmente hay una plataforma llamada IPFS, Inter Planetary File System. Es una plataforma *peer to peer* que permite almacenar ficheros de forma íntegra. Sin embargo, aunque esta plataforma no es de tipo Blockchain se podría realizar una implementación que fusionara estas dos tecnologías.

Por otro lado, en cuanto a posibles mejoras de la aplicación diseñada, se podría hacer un sistema automático de permitir la compartición y revocación del historial médico entre distintas entidades. De esta forma, se le daría más control al paciente sobre sus propios datos.

Asimismo, se podrían añadir más actores sobre la aplicación, como los médicos. Los médicos también gestionan el historial de los pacientes, de modo que podría estudiarse la forma de añadirse este rol y ver cuáles serían sus ventajas y desventajas.

Finalmente, como trabajo futuro, en el caso de que se quisiera realizar el traspaso de esta aplicación a entornos de producción se debería realizar una nueva red Blockchain que permitiera la carga que estos entornos requieren. Por ello se debería realizar una nueva red con mayor número de nodos que permitiera altos procesamientos de datos.

10. Glosario: definiciones y abreviaturas

Aplicación web: Es una herramienta compuesta por una interfaz visual que utilizan los usuarios para comunicarse con el servidor de la aplicación.

Backend: Representa la capa de negocios de la aplicación web, donde se realizan todas las operaciones lógicas necesarias para que la aplicación posea un determinado comportamiento.

Core: Núcleo central de la aplicación donde principalmente se recoge toda la lógica de negocio.

CP: Sigla utilizada para referirse a Caso de Prueba.

Criptomoneda: moneda digital no regulada de intercambio utilizada en Internet y que está controlada por sus desarrolladores.

CU: Sigla utilizada para referirse a Caso de Uso.

Fallo bizantino: Fallo de tipo arbitrario o malicioso.

Frontend: Representa la capa de presentación de una aplicación web mediante la cual los usuarios se conectan con el backend de la misma.

Hash: Función criptográfica computable mediante un algoritmo que cumple:

$$H: U \rightarrow M$$

$$x \rightarrow h(x)$$

Ledger: Anglicismo que en español se refiere a "libro contable". Es el registro donde se almacenan todas las transacciones que se realizan en una plataforma Blockchain.

Minero: máquinas o chips que realizan operaciones computacionales para verificar las transacciones que se llevan a cabo en una red Blockchain.

PBFT: Siglas para referirse al algoritmo utilizado en Hyperledger llamado Practical Byzantine Fault Tolerance.

Plataforma: Conjunto de nodos que conforman una red Blockchain.

PoS: Sigla utilizada para referirse al algoritmo de consenso *Proof-of-Stake*.

PoW: Sigla utiliza para referirse al algoritmo de consenso *Proof-of-Work*.

RF: Sigla utilizada para referirse a un Requisito Funcional.

RNF: Sigla utilizada para referirse a un Requisito No Funcional.

RPCA: Sigla utilizada para referirse al algoritmo de consenso *Ripple Protocol Consensus Algortihm*.

Shareholder: Anglicismo utilizado para referirse a las personas interesadas en el desarrollo de una aplicación.

Whitepaper: Anglicismo utilizado para referirse a un documento de investigación sobre un tema en concreto.

11. Referencias

[1] “Health Level Seven International - Homepage.” [Online]. Available: http://www.hl7.org/. [Accessed: 19-Feb-2017].

[2] “Unified Medical Language System (UMLS).” [Online]. Available: https://www.nlm.nih.gov/research/umls/. [Accessed: 19-Feb-2017].

[3] “SNOMED International.” [Online]. Available: http://www.snomed.org/snomed-ct. [Accessed: 19-Feb-2017].

[4] “WHO | ICD-11 Revision,” *WHO*. [Online]. Available: http://www.who.int/classifications/icd/revision/en/. [Accessed: 19-Feb-2017].

[5] “Docker,” *Docker*. [Online]. Available: https://www.docker.com/. [Accessed: 05-Jan-2017].

[6] “AngularJS — Superheroic JavaScript MVW Framework.” [Online]. Available: https://angularjs.org/. [Accessed: 05-Jan-2017].

[7] “Node.js.” [Online]. Available: https://nodejs.org/es/. [Accessed: 16-Jan-2017].

[8] “Git.” [Online]. Available: https://git-scm.com/. [Accessed: 05-Jan-2017].

[9] “Reinventando la gestión de datos,” *MongoDB*. [Online]. Available: https://www.mongodb.com/es. [Accessed: 05-Jan-2017].

[10] “What is REST (representational state transfer)? - Definition from WhatIs.com,” *SearchSOA*. [Online]. Available: http://searchsoa.techtarget.com/definition/REST. [Accessed: 16-Jan-2017].

[11] “Hype Cycle Research Methodology | Gartner Inc.” [Online]. Available: http://www.gartner.com/technology/research/methodologies/hype-cycle.jsp. [Accessed: 16-Jan-2017].

[12] F. F. Ozair, N. Jamshed, A. Sharma, and P. Aggarwal, “Ethical issues in electronic health records: A general overview,” *Perspect. Clin. Res.*, vol. 6, no. 2, pp. 73–76, 2015.

[13] M. Crosby, Nachiappan, P. Pattanayk, S. Verma, and V. Kalynamaraman, “Blockchain Technology Beyond Bitcoin.” .

[14] S. Underwood, “Blockchain Beyond Bitcoin.” .

[15] M. Swan, *Blockchain, blueprint for a new economy*. .

[16] “Ethereum Project.” [Online]. Available: https://www.ethereum.org/. [Accessed: 05-Jan-2017].

[17] “Monax,” *Monax*. [Online]. Available: https://monax.io/. [Accessed: 05-Jan-2017].

[18] “Welcome to Ripple,” *Ripple*. [Online]. Available: https://ripple.com/. [Accessed: 05-Jan-2017].

[19] “Hyperledger Fabric.” [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/latest/. [Accessed: 05-Jan-2017].

[20] C. Kanaracus, “Don’t believe the blockchain hype: Examining the weaknesses and risks,” *ZDNet*. [Online]. Available: http://www.zdnet.com/article/dont-believe-the-blockchain-hype-examining-its-weaknesses-and-risks/. [Accessed: 16-Jan-2017].

[21] “The Benefits of Blockchain Across Industries.” [Online]. Available: http://www.oracle.com/us/corporate/profit/big-ideas/041316-siyer-2982371.html. [Accessed: 16-Jan-2017].

[22] BitFury Group, “Proof of Stake versus Proof of Work.” .

[23] D. Shwartz, N. Youngs, and A. Britto, “The Ripple Protocol Consensus.” .

[24] C. Perez-Sola and J. Herrera-Joancomarti, “Bitcoin y el problema de los generales bizantinos.” .

[25] M. Castro and B. Liskov, “Practical Byzantine Fault Tolerance and Proactive Recovery.” .

[26] “gnu.org.” [Online]. Available: https://www.gnu.org/licenses/gpl-3.0.en.html. [Accessed: 16-Jan-2017].

[27] “Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal.,” *Noticias JurÃ-dicas*. [Online]. Available: http://noticias.juridicas.com/base_datos/Admin/lo15-1999.html. [Accessed: 16-Jan-2017].

[28] “Blockchain Will Become ‘Beating Heart’ of the Global Financial System,” *World Economic Forum*. [Online]. Available: http://weforum.org/press/2016/08/blockchain-will-become-beating-heart-of-the-global-financial-system/. [Accessed: 05-Jan-2017].