



TRABAJO FIN DE GRADO
GRADO EN INGENIERÍA INFORMÁTICA
MENCIÓN EN TECNOLOGÍAS DE LA INFORMACIÓN



Análisis y prototipado de Identidad Digital Descentralizada basada en Blockchain

Estudiante: Pablo Santos-Cabaleiro
Dirección: José Manuel Vázquez-Naya
Martíño Rivera-Dourado

A Coruña, septiembre de 2022.

A mis padres y a mis abuelos, lo que soy es por vosotros

Agradecimientos

Quiero agradecer a mi familia y amigos el apoyo y cariño que me han dado siempre.

A mis padres por todo el esfuerzo que habéis hecho por mí, por vuestro apoyo incondicional y por creer en mi cuando ni yo mismo lo hacía. Os lo debo todo, sin vosotros nada sería posible.

A mis abuelos por transmitirme siempre vuestro orgullo y apoyo en todo lo que hago. Un beso enorme para los que no lo pudieron ver.

A mis amigos por hacer que todo sea más fácil y vivir esta aventura conmigo.

A José y Martiño por apoyarme con la idea de este proyecto y ayudarme a llevarla a cabo.

Resumen

En la actualidad, cada usuario visita innumerables servicios en Internet, los cuales le solicitan determinada información personal para permitir hacer uso de sus servicios. Con ello, la identidad digital del usuario en Internet se encuentra dispersa y replicada en las diferentes bases de datos de los servicios web. Esto supone las empresas que están detrás de los diferentes servicios web tengan la responsabilidad de garantizar la seguridad y la privacidad de los datos personales del usuario. Este trabajo de fin de grado se centra en la investigación y el análisis de la identidad digital descentralizada basada en la tecnología blockchain. El objetivo principal de esta tecnología es devolver al usuario la autoría y la capacidad de gestión auto soberana de su identidad digital en Internet, proponiendo un sistema descentralizado con una mayor privacidad y seguridad para los datos del usuario. Adicionalmente, se ha desarrollado una prueba de concepto en la que se implementa un sistema de compartición de la identidad digital de un usuario datos haciendo uso de la tecnología blockchain.

Abstract

Nowadays, each user visits many websites and web applications on the Internet, in which the owner companies stores certain information from the user to be able to use the services offered. As a result, the user's digital identity is scattered in many websites databases. This fact means that all these companies that are behind each website have the responsibility to protect and guarantee the user's personal data security and privacy. This degree thesis focuses on the research and analysis of decentralized blockchain-based digital identity. The main objective of this technology is to return to the user the authorship and self-sovereign management capacity of their digital identity on the Internet, proposing a more secure system focused on the privacy of their data. Additionally, a proof of concept has developed in which a system for sharing the digital identity of a data user is implemented using blockchain technology.

Palabras clave:

- Tecnología Blockchain
- Identidad Digital Descentralizada
- Cadena de bloques
- Contrato Inteligente

Keywords:

- Blockchain Technology
- Decentralized Digital Identity
- Blockchain
- Smart Contract

Índice general

1	Introducción	1
1.1	La identidad digital y su problemática actual	1
1.2	La descentralización de la identidad	2
1.3	Objetivos del proyecto	2
1.4	Introducción a la prueba de concepto	3
1.5	Organización de la memoria	4
1.5.1	Estado de la Cuestión	4
1.5.2	Metodología y Planificación	5
1.5.3	Prueba de Concepto	5
1.5.4	Resultados	5
1.5.5	Conclusiones y trabajo futuro	5
2	Estado de la cuestión	6
2.1	La Tecnología Blockchain	6
2.1.1	La cadena de bloques	6
2.1.2	Historia y trayectoria	7
2.1.3	Objetivos y propósitos de la tecnología blockchain	8
2.1.4	Elementos principales de la tecnología Blockchain	10
2.1.5	Smart Contracts	16
2.1.6	Tipos de Blockchains	17
2.2	Identidad Digital	20
2.2.1	El concepto de identidad digital	20
2.2.2	Características	21
2.2.3	Principios clave	21
2.2.4	Aspectos normativos	23
2.2.5	Ciclo de vida	24
2.2.6	Modelos	25

2.2.7	Problemas de la identidad digital en la actualidad	28
2.3	Identidad Digital Descentralizada (IDD)	30
2.3.1	La solución al sistema actual	30
2.3.2	Elementos principales	31
2.3.3	Funcionamiento	32
2.3.4	Ejemplo práctico	33
2.3.5	Principales ventajas de la identidad digital descentralizada	34
2.3.6	Análisis de las soluciones existentes en la actualidad	35
3	Metodología y Planificación	43
3.1	Metodología	43
3.2	Planificación	44
4	Prueba de concepto	46
4.1	Introducción	46
4.2	Diseño	48
4.2.1	Sistema de Gestión de Identidad Digital del Usuario (SGIDU)	48
4.2.2	Sistema Externo de Registro de Usuarios (SERU)	55
4.3	Funcionalidades	59
4.3.1	Sistema de Gestión de Identidad Digital del Usuario (SGIDU)	59
4.3.2	Sistema Externo de Registro de Usuario (SERU)	60
5	Resultados	62
5.1	Blockchain para la gestión auto-soberana de la identidad	62
5.2	Sistema de Gestión de Identidad Digital del Usuario	63
5.2.1	Registro de los datos del usuario en la blockchain	63
5.2.2	Visualización de los datos registrados en la vista de perfil	66
5.3	Sistema Externo de Registro de Usuarios	67
5.3.1	Autenticación del usuario e inicio de sesión en los SERUs	68
5.4	Autorización para la consulta de datos	69
5.4.1	Interfaz de perfil en los SERUs y registro del acceso	70
5.4.2	Visualización del registro de consentimientos aprobados en el SGIDU	73
5.5	Actualización de los datos personales del usuario	74
6	Conclusiones y trabajo futuro	77
6.1	Conclusión	77
6.2	Líneas de futuro del trabajo	78

A Material adicional	80
A.1 Código del contrato inteligente “RegisterData.sol”	80
Bibliografía	87

Índice de figuras

2.1	Ejemplo detallado de una cadena de bloques.	10
2.2	Ejemplo de una estructura de tipo Árbol de Merkle.	11
2.3	Esquema del funcionamiento de una función hash.	11
2.4	Fases del proceso de creación de un nuevo bloque en la blockchain detallado. .	12
2.5	Red Centralizada vs Red Peer to Peer.	13
2.6	Ejemplo de código de un <i>Smart Contract</i> para registrar usuarios.	16
2.7	Ejemplo de contenido de una ABI.	17
2.8	Tabla comparativa de las propiedades de cada tipo de blockchain.	19
2.9	Esquema general de los datos que componen la identidad digital.	20
2.10	Ciclo de vida de la Identidad Digital.	25
2.11	Esquema general de la interacción del proveedor de identidad con los usuarios y los servicios	26
2.12	Comparación de la estructura del modelo centralizado y el modelo descentra- lizado.	27
2.13	Comparación del proceso de emisión y compartición de credenciales entre el sistema Identidad digital convencional y el sistema de Identidad digital Des- centralizada.	30
2.14	Ejemplo propuesto para la emisión, validación y compartición de un título universitario mediante una identidad digital descentralizada.	34
2.15	Etapas del proceso de creación de una IDD con la solución DefinitiveID.	37
2.16	Componentes de la solución Hyperledger Sovereign Identity Blockchain.	38
2.17	Ejemplo de <i>AnonCreds</i> para la verificación de la mayoría de edad de una persona. .	40
2.18	Ejemplo de las diferentes interfaces de la app de Alastria ID.	41
2.19	Proceso de compartición de datos con Alastria ID.	42
3.1	Planificación final actualizada.	45

4.1	Arquitectura general de la prueba de concepto desarrollada. El SGIDU registra los datos personales en la blockchain del usuario y permite su gestión auto-soberana; y los SERUs son servicios en Internet compatibles con este sistema de identidad digital.	47
4.2	Arquitectura y componentes del SGIDU.	49
4.3	Ejemplo de anexión a la cadena y registro del Identificador Descentralizado. . .	51
4.4	Despliegue del <i>Smart Contract</i> y registro de datos del usuario.	52
4.5	Estructuras de almacenamiento de la identidad digital del usuario en el contrato. .	53
4.6	Despliegue del <i>Smart Contract</i> al través del SGIDU.	53
4.7	Ejemplo de interacción del <i>provider</i> para conectar el servicio web con la blockchain y registrar información en el <i>Smart Contract</i> . El objeto implementa las funciones necesarias para interactuar con la blockchain.	54
4.8	Arquitectura y componentes del SERU.	55
4.9	Ejemplo de la obtención de datos del usuario para construir la vista de perfil en el SERU “Pc Shop”.	56
4.10	Esquema de registro de los parámetros de datos necesarios en las bases de datos de los SERUs “Pc Shop” y “Biblioteca”.	57
4.11	Registro de consentimiento de un SERU en el <i>Smart Contract</i> del usuario. . . .	59
5.1	Interfaz de conexión de la blockchain del usuario.	63
5.2	Interfaz principal del SGIDU	63
5.3	Formulario de registro de datos personales del SGIDU.	64
5.4	Estructuras principales del contrato <i>RegisterData.sol</i> (ver apéndice A).	64
5.5	Registro del propietario del contrato y ejemplo de sentencia <i>require</i> . Cuando se crea el contrato se llama a la función <i>constructor()</i> del contrato y se define el valor <i>owner</i> con la <i>wallet</i> del usuario que ha ejecutado la transacción	65
5.6	Funciones de registro de información del usuario en el contrato.	65
5.7	Interfaz de perfil del SGIDU.	66
5.8	Función exclusiva del propietario del contrato para obtener la información personal registrada en él.	66
5.9	Formulario de registro de usuarios del SERU “Pc Shop”.	67
5.10	Formulario de registro de usuarios del SERU “Biblioteca”.	67
5.11	Código de los SERUs para la creación de usuarios.	68
5.12	Formulario de login del SERU “Pc Shop”.	68
5.13	Formulario de registro de usuarios del SERU “Biblioteca”.	68
5.14	Interfaz de solicitud de consentimiento de lectura de los datos personales del usuario en el SERU “Pc Shop”.	69

5.15	Interfaz de solicitud de consentimiento de lectura de los datos personales del usuario en el SERU “Biblioteca”	69
5.16	Función de registro de consentimiento de un SERU del contrato inteligente. . .	70
5.17	Interfaz de perfil del SERU “Pc Shop”.	71
5.18	Interfaz de perfil del SERU “Biblioteca”.	71
5.19	Función de registro de acceso del SERU de los datos del usuario permitidos. . .	72
5.20	Interfaz de solicitud de la clave privada para registrar el acceso del SERU. . . .	72
5.21	Formulario de actualización de contraseña del SERU “Pc Shop”.	73
5.22	Interfaz de perfil del SGIDU.	73
5.23	Función de obtención del histórico de registro de SERU con consentimiento. .	74
5.24	Formulario de actualización de datos del usuario en el SGIDU.	74
5.25	Interfaz de perfil del SGIDU completa.	75
5.26	Función de registro de los anteriores datos personales del usuario en la block-chain.	76
5.27	Interfaz de perfil del SERU “Pc Shop” actualizada.	76
5.28	Interfaz de perfil del SERU “Biblioteca” actualizada.	76

Introducción

El objetivo principal de este capítulo es presentar los diferentes aspectos generales del trabajo como los objetivos de su desarrollo, su estructura y organización o una pequeña introducción del tema a tratar. Los puntos comentados serán extendidos y tratados en detalle a lo largo de los diferentes capítulos en los que se estructura esta memoria (ver sección 1.5).

1.1 La identidad digital y su problemática actual

Actualmente, cada usuario hace uso de una gran cantidad de servicios en Internet, diferentes sitios web o aplicaciones, a las que proporciona una gran cantidad de los datos e información sobre su identidad digital. El Instituto Global McKinsey en su informe [1] identifica este problema: “Esto implica su custodia por una plataforma centralizada que tiene el control sobre los detalles del usuario y la identidad, lo que obliga a comprometer la privacidad del usuario”.

En un hipotético caso donde se produzca un ataque contra la empresa que aloja los datos del usuario y estos sean robados o filtrados, la privacidad del usuario se verá atacada y vulnerada sin éste poder hacer nada para evitarlo. Concretamente, entre los años 2013 y 2018, más de 14 mil millones de registros de datos fueron perdidos o robados, de los cuales, el 4% se encontraban encriptados (inútiles tras ser robados) mientras que el otro 96% se correspondía con datos en plano, con información sensible y personal de los usuarios [2].

1.2 La descentralización de la identidad

A raíz de la aparición de los problemas comentados en la sección 1.1, ha surgido un nuevo concepto de identidad digital descentralizada, que, con base la tecnología blockchain, permite la gestión soberana de la identidad digital del propio usuario por sí mismo. Este nuevo sistema, permite que el usuario pueda gestionar los datos que desee en un sistema personal, al que únicamente él tiene acceso, y desde el cual será posible compartir dichos datos aprobando o denegando las diferentes peticiones de consentimientos solicitadas por terceros. Todo esto de una manera transparente, consciente, privada y mucho más segura para él [3].

Debido a su descentralización, la identidad digital descentralizada propone una solución que trata de eliminar la figura de cualquier tipo de intermediario entre ambos usuarios finales. De esta manera, el generado de las credenciales, la gestión de los datos, la aprobación y revocación de consentimientos es tarea única y exclusivamente del usuario dueño de sus datos [4]. Esta nueva identidad digital propone una serie de beneficios ligados a una buena gestión de la identidad digital, como la posibilidad de crear valor económico con los datos personales del usuario, la mejora de la privacidad y la seguridad de los datos, o la transparencia a la hora la compartirlos.

1.3 Objetivos del proyecto

El objetivo principal de este trabajo ha sido analizar e investigar sobre la nueva solución de identidad digital descentralizada que permite la gestión auto-soberana de la identidad. Este análisis hace hincapié en aspectos como su funcionamiento, estructura, funcionalidades o los diferentes modelos existentes. Al mismo tiempo, también se han fijado los siguientes objetivos:

- Investigar y presentar los aspectos más relevantes de la tecnología blockchain, como su estructura, componentes, funcionamiento o modelos existentes.
- Introducir los diferentes conceptos básicos relacionados con la identidad digital, como su ciclo de vida, características, los diferentes modelos existentes y su estructura o los diferentes aspectos normativos vigentes en este ámbito.
- Analizar la aplicación e integración de la tecnología blockchain en el sistema de gestión de identidad digital descentralizada.
- Identificar y analizar las diferentes soluciones existentes de identidad digital descentralizada en la actualidad.

- Desarrollar de una prueba de concepto funcional que permita investigar en profundidad sobre los conceptos introducidos, como la estructura y el funcionamiento de las tecnologías y herramientas utilizadas en el desarrollo de un sistema de identidad digital descentralizado.
- Presentar y probar los detalles más técnicos a bajo nivel, así como las diferentes herramientas disponibles a través de la implementación de la prueba de concepto.
- Interactuar con la tecnología blockchain y los *Smart Contracts* en un ámbito práctico.
- Identificar aspectos mejorables en las funcionalidades existentes y proponer o crear alguna funcionalidad, método o procedimiento interesante en el ámbito práctico de la identidad digital descentralizada.

1.4 Introducción a la prueba de concepto

La prueba de concepto implementada consiste en un desarrollo simplificado de un sistema de identidad digital descentralizado y gestionado por el usuario. Para ello, se utiliza la tecnología blockchain como método de registro y compartición de los datos personales del usuario con diferentes servicios web externos compatibles, también implementados para la prueba de concepto. Este sistema está compuesto por dos subsistemas principales: el Sistema de Gestión de Identidad Digital del Usuario (SGIDU), a través del cual el usuario podrá conectar su blockchain y almacenar sus datos personales en la blockchain; y los Servicios Externos de Registro de Usuarios (SERU), que implementan un método de registro de usuarios adaptado al SGIDU, el cual implementa un sistema de registro de usuarios sin necesidad de aportar los datos personales pertinentes. En su lugar, el SERU podrán consultar diferentes atributos de la identidad digital del usuario registrado en la blockchain a través de un método de peticiones de consentimiento y registro de accesos en el SGIDU.

En el caso de uso propuesto, el usuario comienza con el despliegue de la blockchain, la que posteriormente registrará sus datos personales a través del SGIDU. Posteriormente, el usuario podrá registrarse en el SERU sin necesidad de aportar los datos personales pertinentes durante el proceso de registro y sin que dichos datos sean almacenados por el SERU en su base de datos propia. Una vez completado el proceso de registro, podrá permitir la petición de consentimiento de acceso del SERU y acceder a su nueva cuenta creada con los datos de acceso que ha permitido en la solicitud.

1.5 Organización de la memoria

La estructura de este documento se divide principalmente en cinco capítulos principales, listados a continuación:

1.5.1 Estado de la Cuestión

Este capítulo 2 presenta las temáticas principales del trabajo en el ámbito teórico. Se divide en tres secciones principales en las que se analiza cada uno de los tres temas principales:

La Tecnología Blockchain

Esta sección 2.1 pretende introducir los conceptos básicos necesarios para comprender como funciona la tecnología blockchain. En ella se trata el origen de la tecnología, sus características principales, los diferentes modelos existentes en la actualidad o los principales elementos de los que se compone, entre otros. Además de la presentación de los conceptos básicos, también se trata de identificar y presentar los aspectos clave de la tecnología sobre los que se fundamenta la identidad digital descentralizada.

La Identidad Digital

En esta sección 2.2 se tratan los aspectos más relevantes de la identidad digital en la actualidad, desde sus características principales, hasta los diferentes modelos existentes, pasando por su ciclo de vida, normativas y regulaciones vigentes. Al final del capítulo se añade un apartado en el cual se identifican los problemas de la identidad digital actual, sección muy importante para comprender el porqué de la siguiente sección.

La Identidad Digital Descentralizada

Tras analizar los problemas existentes de la identidad digital actual, esta sección 2.3 comienza presentando cómo la identidad digital descentralizada pone solución a los problemas identificados en la sección anterior. Posteriormente se tratan los aspectos principales de este innovador sistema como sus elementos principales, su funcionamiento o las diferentes ventajas que aporta. De cara al final de esta sección, se analizan algunas soluciones de identidad digital descentralizada existentes en la actualidad con el objetivo de comprender su funcionamiento, probar e identificar las funcionalidades más interesantes de cada una de ellas.

1.5.2 Metodología y Planificación

En el capítulo capítulo 3 se presentan los modelos de metodología y planificación utilizados a lo largo del trabajo, tanto para el ámbito de investigación como para el desarrollo de la prueba de concepto implementada. Concretamente se ha seguido un modelo de metodología ágil incremental, aplicando las iteraciones definidas por las diferentes fases identificadas en la planificación.

1.5.3 Prueba de Concepto

En este caso, el capítulo 4 se corresponde con un apartado más orientado al aspecto práctico, en el cual se documenta el proceso de desarrollo de una prueba de concepto de identidad digital descentralizada, así como un caso de uso propuesto para probarla. También se definen las diferentes fases de desarrollo como el diseño, la implementación, las funcionalidades identificadas.

1.5.4 Resultados

Tras presentar los detalles técnicos de la prueba de concepto implementada, en el capítulo 5 se documenta un caso de uso de prueba ejecutando la prueba implementada. Con ello se intenta mostrar el funcionamiento general de los sistemas implementados y añadir un mayor nivel de detalle en la explicación de las diferentes funciones implementadas en cada sistema.

1.5.5 Conclusiones y trabajo futuro

Finalmente, se concluye con el capítulo 6, en el que se tratan las conclusiones generales del proyecto, tanto del apartado de la investigación como del apartado práctico. Adicionalmente se incluye un apartado de trabajo futuro en el que se comentan los diferentes aspectos a mejorar de cara al futuro de la prueba de concepto implementada.

Estado de la cuestión

En este capítulo se presenta la investigación realizada de los conceptos principales del trabajo: la Tecnología Blockchain (ver sección 2.1), la Identidad Digital (ver sección 2.2) y la Identidad Digital Descentralizada (ver sección 2.3). El orden para tratar cada uno de los temas está planteado para comenzar introduciendo y presentando tanto la Tecnología Blockchain como la Identidad Digital con vistas a facilitar y mejorar la comprensión del análisis de la Identidad Digital Descentralizada presentado en la última sección de este capítulo.

2.1 La Tecnología Blockchain

En esta sección se tratan diversos puntos sobre la tecnología blockchain haciendo hincapié principalmente en aquellos aspectos más relevantes como son su estructura, los elementos principales de los que se compone, o su funcionamiento.

2.1.1 La cadena de bloques

La tecnología blockchain es un tipo de tecnología cuyo principal elemento es la cadena de bloques, un libro de contabilidad digital que cuenta con una estructura de datos compuesta por bloques interconectados entre sí. La cadena de bloques se encuentra replicada entre una comunidad de usuarios que forman la red, quienes pueden registrar información en ella y compartirla con el resto de los nodos. Cuenta con una estructura caracterizada por la ausencia de una autoridad central y por tener un carácter inmutable con respecto a la modificación de la información ya publicada [5].

La implementación y características de esta tecnología pueden variar en función del ámbito en el que se aplique. En la actualidad, es una tecnología en auge que sustenta una gran cantidad de nuevas plataformas anunciadas de un panorama muy variado. Existen algunos

ejemplos funcionales como métodos de pago como las criptomonedas, libros de contabilidad compartidos en negocios o contratos inteligentes automatizados para pólizas de seguros [5].

2.1.2 Historia y trayectoria

La tecnología blockchain es una tecnología relativamente moderna, con pocos años de trayectoria. En los años ochenta comenzó a sentar sus bases a través de diferentes proyectos que proponían ideas o implementaciones lo que hoy en día son los aspectos fundamentales de la tecnología.

Primeros pasos

En 1982, el criptógrafo David Chaum, propuso por primera vez un protocolo bastante similar a la cadena de bloques en su disertación “Computer Systems Established, Maintained, and Trusted by Mutually Suspicious Groups” [6]. Pero es años después, en 1991, cuando nace la tecnología Blockchain de la mano de Stuart Haber y W.Scott Stornetta. Estos dos científicos investigadores presentaron un innovador sistema basado en una cadena de bloques protegida mediante criptografía, utilizando marcas de tiempo inmutables. Su objetivo era desarrollar una solución computacional para sellar la fecha y hora de los documentos digitales, con el objetivo de hacerlos inmutables y proteger la integridad del documento. El siguiente año, actualizaron su trabajo incorporando los árboles de Merkle, con los que mejoraban la eficiencia del sistema al permitir la recopilación de más documentos en un solo bloque [7].

La aparición del Bitcoin

En 2008, Satoshi Nakamoto (pseudónimo), debido al hundimiento del sistema financiero global, propuso un nuevo protocolo de sistema de pagos electrónicos. Este sistema, se basaba en una red Peer to Peer (ver sección 2.1.4) que permitiría registrar las transacciones económicas digitales realizadas entre diferentes usuarios. El 31 de Octubre de 2008, Nakamoto publicó el informe [8] donde presentaba y explicaba en detalle el funcionamiento de este sistema. En este informe conceptualizó el concepto de cadena de bloques distribuida y modificó el modelo de *merkle tree*, creando un sistema más seguro. Este suceso fue muy relevante para la tecnología blockchain ya que los conceptos propuestos en su sistema se convertirían en la columna vertebral de la tecnología blockchain en el futuro.

Dos factores clave: Ethereum y Hyperledger

Desde la presentación del sistema de pagos propuesto por Nakamoto, han sido numerosas las aplicaciones propuestas que han surgido durante los años posteriores a su lanzamiento.

En 2013, Vitalik Buterin, uno de los primeros contribuyentes al código base de Bitcoin, comenzó a trabajar en un tipo de blockchain maleable, que tuviese la capacidad de realizar varias funciones diferentes en una red Peer to Peer (ver sección 2.1.4). Es así como tras algunos años de desarrollo, Ethereum es lanzado oficialmente en 2015. Como novedad, Ethereum implementaría una nueva función destinada a registrar otros activos como contratos o slogans, dando vida a los famosos *Smart Contracts*. Esto permitió a Ethereum convertirse en la base principal para desarrollar aplicaciones y proyectos totalmente descentralizadas [7].

Dos años después, en 2015, de la mano de Linux Foundation, surge el proyecto Hyperledger. Este proyecto sería de gran importancia para el desarrollo de la tecnología blockchain, ya que, además de mejorar el propio desarrollo de la tecnología, implementaría los ledgers, facilitando las transacciones empresariales en el mundo industrial. Este proyecto tuvo una gran repercusión llegando incluso hasta empresas punteras como IBM, Intel o SAP SE.

Actualidad y previsión futura

Durante los siguientes años hasta la actualidad, el número de proyectos que utilizan la tecnología blockchain ha ido creciendo a lo largo de los años. En un informe [9] realizado a 400 empresarios españoles revela que el 41% todavía no comprende la tecnología y sus beneficios. A nivel mundial, el Institute for Business Value, con su estudio [10], dictamina que actualmente un 33% de los 3000 altos ejecutivos encuestados afirman que sus empresas incorporan esta tecnología o se plantean el uso de esta.

Lo que es innegable es su proyección de futuro en los próximos años. Así lo avala el estudio [11] realizado por AMETIC e IDC, donde estiman un crecimiento sostenido de hasta un 53% en Europa occidental, llegando a una inversión total de 4.148,9 millones de dólares en 2023. En el caso de LATAM la cifra es del 49% para la misma fecha.

2.1.3 Objetivos y propósitos de la tecnología blockchain

La tecnología blockchain desde su origen, como hemos visto en la sección 2.1.2, ha tenido una serie de propósitos y objetivos determinados en cuanto a su funcionalidad en el mundo de Internet. A continuación, analizaremos cada uno de ellos.

Descentralización

La descentralización es el objetivo principal de la tecnología blockchain. Este propósito nace principalmente de la idea que propuso Nakamoto (ver sección 2.1.2) de quitarle el poder a las instituciones para que esté en manos de los usuarios. Con la descentralización, la tecnología blockchain trata de eliminar la figura de la autoridad central para sustituirla por un acuerdo común entre todos los usuarios que pertenezcan a la red. Esta característica se consigue a partir de la distribución de la información almacenada en la cadena a través de los diferentes nodos que la componen. A medida que se van anexionando nuevos nodos a la red, estos reciben una copia de la información registrada en la cadena, de manera que la información de la blockchain se encuentra descentralizada entre la totalidad de los nodos de la red.

Eliminar la figura de intermediario

Este objetivo está bastante relacionado con el anterior. La tecnología blockchain busca la independencia de los usuarios a la hora de comunicarse entre ellos o llevar a cabo cualquier tipo de intercambio de información. De esta manera, trata de eliminar aquellos elementos que impidan la relación directa entre los usuarios finales, eliminando cualquier organización o entidad que condicione este proceso. En su lugar, sustituye estas figuras dependientes por fundamentos matemáticos y tecnologías, como la criptografía asimétrica o la emisión de credenciales descentralizadas.

Trazabilidad de registros, transacciones y datos

Otro de los objetivos planteados por Nakamoto cuando lanzó su sistema de intercambio de pagos fue que permitir consultar de manera totalmente transparente el movimiento de los fondos intercambiados entre los usuarios. Con esto, el creador del Bitcoin presentaba a los usuarios un sistema seguro, pues cualquiera podría consultar los flujos de dinero intercambiados entre los usuarios.

Inmutabilidad

A través del uso de la criptografía asimétrica, esta tecnología busca crear una estructura inmutable, que no permita su modificación en el futuro. La integridad de los datos es una de las características más importantes de la tecnología blockchain, ya que aporta un valor único que otras tecnologías no disponen. Este aspecto es muy interesante para algunos ámbitos en los que se necesita la garantía de que un dato no pueda ser modificado una vez haya sido registrado.

2.1.4 Elementos principales de la tecnología Blockchain

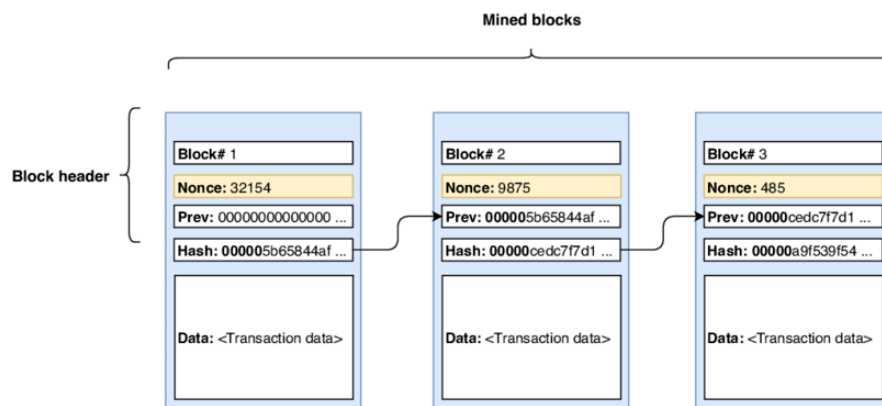
La tecnología blockchain, como se introduce en la sección 2.1.1, se compone de una serie de elementos principales interoperables entre sí. A continuación, analizamos en detalle cada uno de ellos.

Cadena de bloques

La cadena de bloques, conocida también como blockchain, es el elemento encargado del almacenamiento de los registros y transacciones realizadas por los usuarios. Estas transacciones son registradas en los diferentes bloques que conforman la cadena. Estos bloques, se encuentran interconectados entre sí a través del registro del código hash del bloque anterior, estableciendo una relación recursiva entre todos los bloques que componen de la blockchain. A continuación, analizaremos la estructura propia de la cadena de bloques, así como los elementos que componen cada bloque y los diferentes tipos de bloques existentes:

La estructura de un bloque

La estructura general de un bloque (ver figura 2.1) puede variar en función del tipo de blockchain y la funcionalidad sobre la que se aplica, pero generalmente se compone de 3 elementos: los datos de las transacciones almacenadas, el hash o función resumen del bloque y el hash del bloque anterior. En algunos casos, también puede incorporar una marca de tiempo (o *timestamp*), el *nonce*, el tamaño del bloque o parámetros relativos a la resolución del *nonce* o a la dificultad generación del hash [12].



Note: The fields of *Nonce*, *Prev*, and *Hash* contain arbitrary values

Figura 2.1: Ejemplo detallado de una cadena de bloques.

Elementos de un bloque

- En el campo de **datos** se encuentra la información de los datos almacenados en cada bloque. En este campo es posible almacenar valores simples desde un strings o números hasta un conjunto de datos, como por ejemplo transacciones recogidas en una estructura de árbol de Merkle (ver figura 2.2). Este tipo de estructura en forma de árbol permite simplificar un valor como resumen de todas las transacciones contenidas en el bloque. El tipo de dato almacenado depende del tipo de blockchain, del ámbito y de la funcionalidad al que es aplicada.

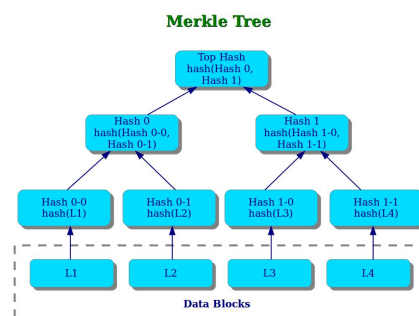


Figura 2.2: Ejemplo de una estructura de tipo Árbol de Merkle.

- El **timestamp** o marca de tiempo se encarga de realizar el registro de la fecha y la hora de generación del bloque, en formato UTC. Este valor se utiliza también para establecer los parámetros de proceso de minería, permitiendo a los nodos determinar el tiempo de extracción de los bloques para ajustar el parámetro de dificultad de creación de un nuevo nodo.
- El **hash** es un valor en formato hexadecimal de una longitud determinada calculado mediante una función aplicada a la información recogida en el bloque (ver figura 2.3). Una característica clave es que cada hash generado será distinto, ya que nunca se generan hashes idénticos debido al *timestamp*. Una vez los datos son almacenados en el bloque, este se cierra generando su hash, protegiendo la integridad de los datos del del bloque. Una modificación de los datos almacenados en el hash supondría obtener otro hash distinto. Además, el hash permite identificar a cada uno de los bloques de la cadena.



Figura 2.3: Esquema del funcionamiento de una función hash.

- El **hash del bloque anterior** es el componente que permite la interconexión entre los bloques. Esta conexión se produce a través el almacenamiento del hash del bloque anterior en cada bloque. Esto supone que además interconectar entre sí todos los bloques de la cadena, recursivamente, asegura que el bloque anterior no ha sido modificado, ya que, en ese caso no coincidiría su valor con el del hash almacenado en el bloque anterior.
- El **nonce** es un número aleatorio que añade un grado adicional de seguridad y dificultad a la hora de la obtención del código hash identifica un bloque. Durante el proceso de minería, los mineros tratan de averiguar este valor probando aleatoriamente números hasta obtener un código hash válido que cumpla las diferentes condiciones establecidas en cada caso.

Para registrar nuevos datos en la blockchain, debe registrarse la información que se desea almacenar, generar los datos necesarios para su creación y finalmente añadirlo a la cadena. Este proceso se basa principalmente en 4 etapas (ver figura 2.4) desde la generación del bloque hasta su adicción a la cadena.

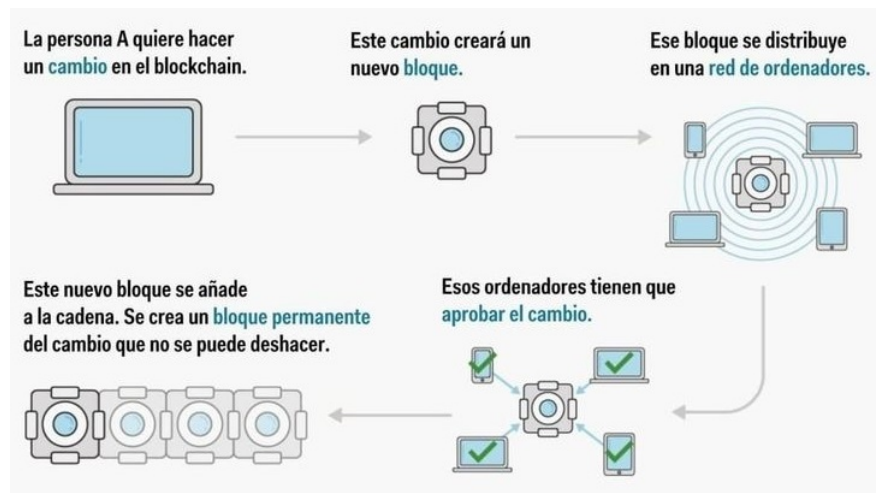


Figura 2.4: Fases del proceso de creación de un nuevo bloque en la blockchain detallado.

El proceso comienza con la creación de un nuevo bloque añadiendo la información correspondiente en el campo de datos y generando el código hash que identifica dicho bloque. Una vez generado, el bloque es enviado al resto de nodos de la red para que estos den su aprobación y el bloque pueda ser catalogado como válido. Para que esto se cumpla, deberá ser aprobado por al menos un 51% de los nodos que componen la totalidad de la red. En caso de ser válido, será añadido a la cadena de bloques y el nodo que haya generado ese bloque enviará al resto una copia de la cadena de bloques actualizada.

Criptografía

La criptografía es uno de los pilares principales sobre los que se sustenta la tecnología blockchain. Tiene la responsabilidad de proveer un mecanismo de seguridad para la codificación segura de las reglas del protocolo que rigen el sistema. Además, proporciona los hashes, firmas e identidades digitales encriptadas en los diferentes procesos de la cadena de bloques. Su función en la tecnología es imprescindible, tanto para permitir el funcionamiento general de la tecnología como para evitar la manipulación, hurto o modificación de la información registrada en la cadena de bloques.

Sistema Descentralizado

Un sistema descentralizado se caracteriza por la ausencia de una autoridad que encargada del control de la información de la red. En este tipo de sistemas, son los quienes controlan la red. Dependiendo del tipo de blockchain, puede existir cierta jerarquía o privilegios en cuanto a los derechos y funciones de los diferentes nodos que componen la red (ver sección 2.1.6).

Red Peer to Peer

Las redes *Peer to Peer* (o P2P) son redes en las que todos los nodos están interconectados entre ellos en una misma red (ver figura 2.5). Este tipo de redes son descentralizadas, es decir, no existe la figura de nodo “servidor” que proporcione la información al resto de nodos “clientes” para que puedan comunicarse. En algunos casos, la información se encuentra segmentada entre diferentes nodos, de manera que cuando un nodo solicita un dato, este es proporcionado directamente bien por uno o por varios nodos.

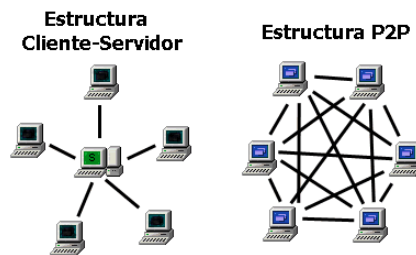


Figura 2.5: Red Centralizada vs Red Peer to Peer.

La tecnología blockchain cuenta con esta misma estructura. Se compone de una red *Peer to Peer* de nodos que intercambian información directamente entre ellos a través del protocolo de comunicación establecido (ver sección 2.1.4). Cada nodo cuenta con una copia local de la blockchain, la cual se va actualizando cada vez que un nuevo nodo añade un bloque válido a la cadena. Cuando esto sucede, el nodo encargado de añadir el bloque envía una copia de

la nueva cadena a todos los nodos para que estos actualicen su cadena de bloques local. De acuerdo con el siguiente artículo [13], existen diferentes tipos de nodos:

- **Nodos completos:** son aquellos nodos que almacenan una copia completa y actualizada de la cadena de bloques. Realizan funciones vitales para el funcionamiento de la red como la validación de los nuevos bloques generados y la transmisión de las actualizaciones de la cadena de bloques al resto de nodos.
- **Nodos Ligeros:** son aquellos nodos que dependen de un tercero para realizar las validaciones de las transacciones en la red. A diferencia de los nodos completos, no almacenan una copia de la Blockchain, sino que es el Supernodo quien le suministra esa información.
- **Nodos de minería:** al igual que los nodos completos, almacenan una copia completa de la blockchain, pero adicionalmente ejecutan el software propio de minería para calcular el hash del nuevo bloque.
- **Maestros o Nodos Maestros:** sus funcionalidades están ligadas al tipo de blockchain en la que son ejecutados, pueden tener participación en votaciones, ejecutar operaciones de protocolo, etc.
- **Supernodos:** son los nodos encargados de actuar como punto de comunicación e interconexión con el resto de los nodos de la red, proporcionando la información necesaria al resto de nodos que lo necesiten.

Protocolo de comunicación

El protocolo de comunicación es el software informático que permite la comunicación entre los distintos nodos de una red. Funciona de la misma forma que cualquier tipo de protocolo (como TCP/IP o SMTP). Otorga un estándar común en el que se definen las comunicaciones entre los ordenadores que componen en la red.

Algoritmos de consenso

El consenso es el elemento principal encargado de velar por la seguridad e integridad de la cadena, verificando que la información almacenada en la cadena de bloques es válida. Su función principal consiste en supervisar y controlar el crecimiento de la cadena de bloques, verificando y validando la creación de nuevos bloques. Para ello, se sustenta en un protocolo o mecanismo de consenso, a través del cual se logra la toma de decisiones con respecto a la validación de un nuevo bloque. Estos mecanismos cuentan con una serie de objetivos comunes definidos que velan por el interés grupal:

- **Búsqueda de acuerdo:** debe generar el mayor acuerdo posible del grupo.
- **Participativo:** debe ser tal que todos los nodos participen activamente en el proceso general.
- **Igualitario:** cada uno de los votos tiene el mismo peso.
- **Cooperativo:** los participantes no deben anteponer intereses personales y trabajar en equipo.
- **Inclusivo:** debe participar la mayor cantidad de votantes posible.

Principalmente existen dos tipos de mecanismos de consenso definidos actualmente:

Proof of Work

Proof of Work, o prueba de trabajo, es un protocolo inicialmente creado con el objetivo de reducir el spam en el correo electrónico. En 2009 fue adaptado al funcionamiento de la cadena de bloques de Bitcoin, siendo incorporado posteriormente a la gran mayoría de las blockchains. Este protocolo consiste en establecer un acertijo criptográfico, que requiere de una capacidad computacional muy alta para ser resuelto, para permitir la adición de un nuevo bloque a la cadena. A través de este método, se limita el crecimiento de la cadena de bloques a una cantidad de tiempo limitada, aumentando la seguridad de la cadena y evitando que cualquier hardware con una gran capacidad computacional pueda manipular y reescribir los bloques de una cadena desde cero.

Proof of Stake

Proof of Stake, o prueba de participación, es un protocolo creado como solución a los problemas de contaminación y gasto energético causados por el protocolo de prueba de trabajo. En este nuevo protocolo, se modifica el método de validación de los bloques, eliminando a los mineros y la prueba computacional que debían realizar para anexionar un nuevo bloque. En su lugar, serán los propietarios de la moneda de la blockchain quienes se encarguen de la supervisión de los nuevos bloques creados. Estos propietarios se podrán convertir en validadores a cambio de proporcionar una gran cantidad de sus monedas, mostrando así su compromiso con la red. Una vez convertidos en validadores, ellos serán los encargados de decidir si un nuevo bloque es válido y cumple los requisitos necesarios para añadirse a la cadena de bloques, o deba ser descartado.

2.1.5 Smart Contracts

Los *Smart Contracts* son programas informáticos que son almacenados en la cadena de bloques como una nueva transacción en la cadena. Se utilizan para automatizar y descentralizar cualquier tipo de función, acuerdo, o condición registrados, independientemente de su complejidad [14]. Los contratos inteligentes permiten programar transacciones y acuerdos confiables entre partes dispares y anónimas sin la necesidad de una autoridad central, un sistema legal o un mecanismo de cumplimiento externo [15]. Además, permiten interactuar con valores activos reales, de manera que, cuando se dispara una condición programada, el contrato ejecuta la cláusula contractual correspondiente.

Los términos y acuerdos entre las dos entidades recogidas en los contratos inteligentes se programan mediante código informático. Los contratos inteligentes tienen diferentes utilidades, pero se utilizan principalmente para definir condiciones de diferentes tipos de contratos en todo tipo de ámbitos como leyes, propiedad inteligente, cláusulas de compañías de seguros, préstamos, financiaciones, etc [16].

En el caso de Ethereum, blockchain fundadora de *Smart Contracts*, los contratos se programan en un nuevo lenguaje de programación orientado a objetos llamado *Solidity* [17]. Este lenguaje proporciona una serie de funciones, estructuras y sentencias que permiten programar diferentes estructuras complejas de datos, condiciones o requisitos. Otra funcionalidad interesante es la programación del costo monetario por el uso de funciones las diferentes funciones programadas, en este caso, en *Ether*, la moneda que utiliza Ethereum. En la figura 2.6 es posible ver un ejemplo básico de un de *Smart Contract* para registrar los usuarios de la red con su correspondiente *wallet*.

```
1  pragma solidity 0.7.0;
2
3  // Contrato que registra usuarios con su correspondiente wallet
4  contract Users {
5
6      mapping (address => bool) users;    // almacenamiento de wallets de usuarios
7
8      event UserRegistered(address indexed user);    // evento, ejecutado cada vez que un usuario es registrado
9
10     constructor() {
11         register(); // Registra el creador del contrato como el primer usuario
12     }
13
14     function register() public {
15         users[msg.sender] = true;
16         emit UserRegistered(msg.sender);
17     }
18 }
```

Figura 2.6: Ejemplo de código de un *Smart Contract* para registrar usuarios.

Una vez el contrato sea programado se compila, generando dos salidas: la ABI (Application Binary Interface) y la dirección hexadecimal que identifica el contrato. La ABI (ver figura 2.7) es un conjunto de binarios en formato JSON que contiene información de las funciones y las variables definidas en el contrato. Una vez compilado, se despliega en la blockchain registrando una nueva transacción en la cadena de bloques con la información necesaria. Una vez desplegado, mediante el ABI y la dirección del contrato, los programas pueden interactuar con el contrato realizando llamadas a las funciones definidas en él.

```
1 [{"inputs": [],
2   "stateMutability": "nonpayable",
3   "type": "constructor"},
4   {"anonymous": false,
5    "inputs": [{"indexed": true,
6               "internalType": "address",
7               "name": "user",
8               "type": "address"}],
9    "name": "UserRegistered",
10   "type": "event"},
11  {"inputs": [],
12   "name": "register",
13   "outputs": [],
14   "stateMutability": "nonpayable",
15   "type": "function"}]
```

Figura 2.7: Ejemplo de contenido de una ABI.

2.1.6 Tipos de Blockchains

La tecnología blockchain es una tecnología maleable que permite crear diferentes alternativas con una base común, pero con unas funcionalidades o características muy diferentes. En el mundo de las blockchain existen principalmente tres tipos: las blockchain públicas, privadas e híbridas.

Públicas o Permissionless

Las blockchain públicas, o sin permisos, son aquellas a las que cualquier usuario puede acceder a la red. En ellas cualquier usuario puede participar en ella e interactuar con la cadena de bloques. Para ello, el usuario debe descargarse la aplicación correspondiente, conectarse a la red y solicitar la versión más actualizada de la cadena de bloques. Una vez se haga con la copia actualizada de la cadena, el usuario cuenta con los mismos derechos que el resto de los participantes de la red a la hora de proponer y validar transacciones. La validación de los nuevos bloques se realiza a través de los mecanismos de consenso establecidos (ver sección 2.1.4).

Este tipo de blockchains tienen unas características muy definidas. No existen entidades centralizadas, ni ningún tipo de autoridad central que administre la red. Tampoco existen diferencias jerárquicas entre los nodos participantes en cuanto a funcionalidades y derechos, pues todos ellos son iguales en la red. Además, es frecuente que los participantes sean usuarios anónimos. En cuanto a su mantenimiento económico, se sostiene a través de la minería y el cobro de comisiones por cada transacción realizada en la red.

Las blockchains públicas son utilizadas especialmente útiles en soluciones que requieran transparencia o un consenso grupal. Algunos ejemplos podrían ser sistemas de votación transparentes o métodos de pago descentralizados.

Privadas o Permissioned

Las blockchains privadas, o permissionadas, generalmente cuentan con los mismos elementos que una blockchain pública, pero a diferencia de estas, se desarrollan en entornos restrictivos, bajo el control de uno o un grupo de usuarios. En este tipo de blockchains su acceso es restringido, de manera que cualquier usuario que quiera participar en la red, debe realizar una petición de acceso previamente.

Este tipo de blockchains ofrece una serie de características muy distintas a las blockchain públicas. En ella, los usuarios se encuentran clasificados en roles, diferenciados por los derechos y las funciones que puede ejecutar cada rol (administración de nodos, lectura, escritura, etc). No es posible el anonimato de los participantes, ya que estos deben identificarse en el proceso de solicitud de acceso a la red. Además, este tipo de redes cuentan con algoritmos de consenso más eficientes debido a que, por lo general, existe con un menor número de nodos en su red. En cuanto al mantenimiento económico, en este caso depende directamente de la organización o usuario encargado del proyecto.

Estas características hacen que este tipo de blockchain sea especialmente útil en determinados sectores que requieran gestionar el acceso de sus usuarios o la protección de la información. Frecuenta su uso en aplicaciones como la gestión de la privacidad en el sector médico, la realización de auditorías y automatización de registros, el seguimiento de artículos físicos o la gestión de la identidad digital en Internet.

Federadas o híbridas

Las blockchains federadas, o híbridas, constituyen una mezcla de los dos tipos anteriores. Coinciden con las blockchain públicas en el propósito de intentar eliminar la completa autonomía de una sola entidad sobre una blockchain, pero en este caso, no son abiertas a la participación de cualquier público. En su lugar operan bajo el liderazgo de un grupo y son

administradas en conjunto por un número determinado de individuos, organizaciones, entidades o compañías (consorcios). En ella los participantes de la red pueden tener diferentes roles de lectura, escritura o validación de la información. En cuanto al protocolo de consenso, es controlado por un grupo de nodos preseleccionados previamente. Con respecto al anonimato de los usuarios, es flexible, ya que dependerá de la funcionalidad para la cual esté pensado su uso [18].

Este tipo de blockchains son las más solicitadas a la hora de construir soluciones con grandes volúmenes de datos entre distintas entidades con una alta necesidad de confianza. Son una buena opción para soluciones compartidas entre entidades, empresas, gobiernos o asociaciones, principalmente en sectores en los que se dé esta estructura de consorcio, como el sanitario, la banca o el energético.

En la figura 2.8 podemos ver una comparación de las diferentes características de cada tipo blockchain.



	Públicos Bitcoin, Ethereum, Litecoin	Privados Hyperledger, Corda, Quorum	Federados Hyperledger, Corda, Quorum
Cualquiera puede participar	✓	✗	✗
Los participantes actúan, en general, como nodos	✓	✗	✗
Transparencia	✓	≈	≈
Hay un único administrador	✗	✓	✗
Hay más de un administrador	✗	✗	✓
No hay administradores	✓	✗	✗
Ningún participante tiene más derechos que los demás	✓	✗	✗
Se pueden implementar Smart Contracts	✓	✓	✓
Existe recompensa por minado de bloques	≈	✗	✗
Soluciona problema de falta de confianza	✓	✗	≈
Seguridad basada en protocolos de consenso	✓	✗	≈
Seguridad basada en funciones hash	✓	≈	≈

Figura 2.8: Tabla comparativa de las propiedades de cada tipo de blockchain.

2.2 Identidad Digital

A medida que los usuarios navegan por Internet y comparten información, van formando lo que es llamada su “Identidad Digital”. En esta sección, se presentan distintos aspectos relacionados con este nuevo concepto: su significado y características, ciclo de vida, elementos de los que se compone, diferentes modelos existentes y los problemas que presenta el sistema actual.

2.2.1 El concepto de identidad digital

El concepto de identidad digital se refiere específicamente al conjunto de datos, comportamientos, rasgos o características que identifican a un usuario concreto en un entorno virtual (ver figura 2.9 [19]).

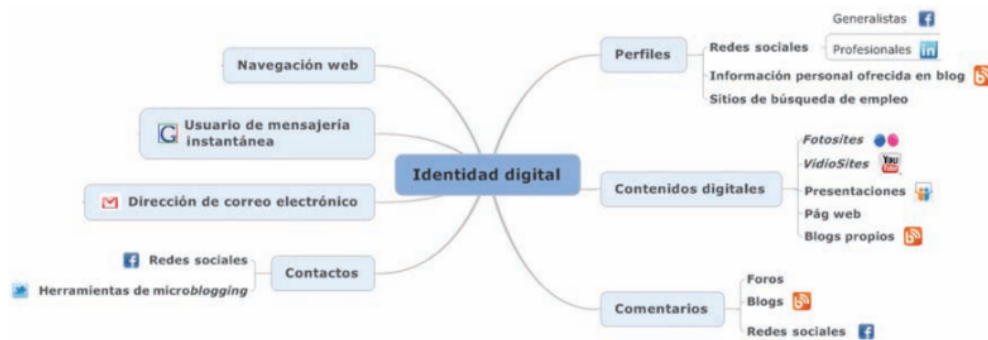


Figura 2.9: Esquema general de los datos que componen la identidad digital.

Según el modelo planteado por F.Georges [20], la identidad digital está conformada por tres tipos de “sub-identidades”:

- **Identidad Declarada:** la combinación de los atributos relevados propiamente por la persona, como su nombre y apellidos, correo electrónico, dirección, fotografías o elementos biométricos.
- **Identidad Actuada:** las acciones que realiza el usuario en el mundo virtual. Se definen por su interacción con la red o con otros usuarios. Algunos ejemplos de esta sub-identidad puede ser el historial web, intereses y gustos del usuario, opiniones, publicaciones, etc.
- **Identidad Inferida:** calculada según el análisis de las acciones realizadas por el usuario. A partir de estas acciones se realizan diferentes análisis como estudios de los perfiles de compradores en función de su historial de compras, estudios de gustos e intereses para anuncios personalizados, etc [19].

2.2.2 Características

La identidad digital tiene una serie de características intrínsecas. En los artículos [19], [21] se identifican algunas de las más interesantes:

- **Subjetiva:** La información depende de la percepción y subjetividad del resto.
- **Valiosa:** Tiene gran relevancia en el mundo digital. Puede definir comportamientos, acciones o intereses que diferentes entidades pueden utilizar para cualquier fin. Sin ella no podría ser posible la realización de diversas acciones como transacciones.
- **Indirecta:** No es posible conocer la persona directamente, únicamente si esta lo desea.
- **Compuesta:** Se crea a partir de la intervención de una o varias entidades o grupos independiente del consentimiento dado por el propietario de dicha identidad.
- **Real:** Repercute de manera directa en el mundo real, bien positiva o negativamente.
- **Contextual:** Según el perfil de la identidad digital tendrá una información u otra.
- **Crítica:** Su uso por terceros puede implicar riesgos para la persona.
- **Dinámica:** Está sometida a cambios constantes y nuevas incorporaciones.

2.2.3 Principios clave

La identidad digital cuenta con una serie de características y principios clave que fundamentan los pilares de esta identidad virtual. Identificados en el artículo [19], elaborado por la Fundación Telefónica, se analizan a continuación cada uno de ellos:

Visibilidad

La visibilidad de la identidad digital hace referencia a la fama o lo popularidad que puede tener un usuario en el mundo virtual. La identidad digital permite al usuario elegir el grado de visibilidad que quiere tener en Internet en diferentes ámbitos. El impacto de la visibilidad de una persona en el mundo digital puede ser medible, dependiendo del contexto en el que se encuentre: seguidores, contactos, artículos publicados, etc.

Reputación

La reputación de un usuario es un valor que no depende principalmente de la persona asociada a esa identidad digital. Este valor viene establecido por opiniones o valoraciones terceros sobre la identidad de una persona o entidad en Internet. Este principio tiene bastante

consideración en las personas, especialmente en ámbitos de interés o sobre los que desean llevar a cabo cualquier tipo de acción. Pongamos un ejemplo, la compra de un producto en una web, es habitual que el comprador consulte opiniones de otros compradores. En empresas como LinkedIn, Youtube o las redes sociales, la reputación de un usuario es un valor bastante relevante.

Portabilidad o Globalidad

Esta característica expresa la capacidad de estandarización y compartición de los datos que componen la identidad digital de un usuario. Un ejemplo son los estándares de autorización *single sign-on* que permiten la compartición de los datos entre entornos, entidades o ámbitos distintos. La empresa PayPal [22], implementa un modelo de funcionamiento que permite la portabilidad de datos de manera fiable entre terceros, facilitando la experiencia de uso al usuario y, al mismo tiempo, realizando una compartición de datos segura.

Privacidad

La privacidad de los datos hace referencia al derecho del usuario tanto de proteger sus datos en la red como de decidir qué información es visible. En la actualidad existe un gran flujo de compartición de datos entre usuarios y empresas, muchos de ellos sin el debido conocimiento de las condiciones de este proceso. Este es un tema muy debatido por diversas organizaciones e instituciones que intentan proteger la privacidad de los usuarios de las empresas que obtienen beneficios económicos con ellos. Actualmente el comercio de datos es uno de los sectores en auge, pues la compra/venta de datos personales entre empresas es uno de los sectores más cotizados. Así lo confirma el periódico The Economist en 2017, que informaba que los datos personales habían superado al petróleo como la fuente de valor más valiosa [23].

Transparencia

La transparencia se refiere principalmente al conocimiento de la información y las condiciones acordadas en la cesión de los datos entre el usuario y una entidad. La transparencia debe centrarse en ofrecer valor a cambio del servicio prestado de una manera lícita por parte del solicitante, facilitando la comprensión de las políticas del intercambio al usuario.

Seguridad

La seguridad de los datos del usuario viene dada, principalmente, por la gestión que realice de ellos. En la identidad digital es clave una buena gestión de los datos personales, con la que escoger quien tiene acceso a ellos, durante cuánto tiempo u otros parámetros referidos con la exposición de los datos personales en Internet.

2.2.4 Aspectos normativos

En el ámbito normativo existen diferentes reglas y leyes por las que se rige la identidad digital en la actualidad. Al igual que en otros ámbitos, para la identidad digital no existe un estándar global, por lo que su uso depende principalmente de las leyes establecidas en ese territorio.

España

Actualmente, el gobierno España trabaja sobre un modelo centralizado (ver sección 2.2.6) a través del DNI electrónico (actualmente 4.0), los certificados digitales y la firma electrónica. En ese ámbito la legislación vigente viene dada por el Real decreto 1553/2005, recogido en el artículo 307 del BOE [24]. En este documento se recoge la regulación del documento nacional de identidad y sus certificados de firma electrónica divididos en distintos artículos. En cada uno de ellos se indican los diferentes requisitos para su expedición, validez, renovación o su contenido. Además de las comentadas, las siguientes leyes recogen información sobre conceptos relacionados con la identidad digital en el territorio español:

- Identificación remota por vídeo para expedición de CE de confianza [25].
- Ley 6/2020 [26].
- Ley 11/2007 de acceso electrónico de los ciudadanos a los servicios públicos [27].

Unión Europea

A nivel europeo, se aplica de manera vigente el Reglamento (UE) 910/2014 [28], también conocido como el Reglamento eIDAS, aprobado el 23 de Julio de 2014. Este reglamento está destinado a la identificación electrónica y los servicios de confianza para las transacciones electrónicas del mercado interior (implementado en España por la Ley 6/2020 [26]). Este documento trata de establecer un marco legal único y centralizado para el reconocimiento de firmas electrónicas e identidades en la UE.

Al mismo tiempo se encuentra vigente el Reglamento General de Protección de Datos (RGPD), en el que se define la normativa relacionada con la gestión de la identidad digital y el refuerzo del control de los usuarios sobre sus datos en Internet. Cabe nombrar también la segunda directiva de servicios de pago PSD2 [29], creada para garantizar un mayor nivel de seguridad en los datos durante los pagos realizados por los usuarios en Internet. Recientemente se ha elaborado una propuesta que actualiza diversos aspectos del vigente Reglamento eIDAS, denominado eIDAS 2 [30]. La UE propone también una serie de recomendaciones relacionadas con el marco digital [31].

2.2.5 Ciclo de vida

La gestión de la identidad digital de un usuario está conformada por un conjunto de procesos de negocio y tecnologías que conforman su ciclo de vida (ver figura 2.10). Principalmente, consta de cinco fases principales:

Creación

La creación se corresponde con la fase que inicia el proceso, en la cual se dan de alta los datos correspondientes en cada caso. Este proceso se puede producir de diferentes modos en función del ámbito en el que nos encontremos. En el caso de la identidad digital de un usuario, se produce cuando realiza sus primeras búsquedas o se registra en sus primeras webs aportando diferentes datos personales.

Propagación

La propagación consiste en el proceso llevado a cabo entre dos entidades, usuarios u organizaciones en donde se produzca un intercambio de datos personales entre ellas. Este intercambio de información es crucial en la gestión de la información, debe realizarse de manera segura, lícita y transparente para ambas partes.

Uso

Junto con la anterior etapa, la etapa del uso de los datos conforma prácticamente la totalidad de la gestión de la identidad digital de un usuario. En esta etapa, los sistemas, entidades o agentes cuyo consentimiento de datos fue dado anteriormente por el usuario, utilizan los datos del usuario para diferentes servicios propios como consultas, autenticaciones, estudios de intereses o incluso la realización de transacciones.

Mantenimiento

La fase de mantenimiento se mantiene presente durante la totalidad del ciclo de vida comprendido entre las fases de propagación y uso. Posteriormente a su creación y antes de su eliminación, se realizan las distintas modificaciones necesarias sobre los datos, propiedades y permisos creados por el usuario previamente a su transmisión. Muchos de estos cambios implican una redistribución de los datos o incluso la anulación de permisos a poseedores de datos cuyo consentimiento fue aceptado previamente.

Eliminación

Por último, el ciclo termina con el borrado de la identidad digital del usuario, cerrando todas las anteriores fases. En esta etapa, el usuario puede ejecutar la orden de borrar todos los datos de todos los sistemas siempre y cuando no exista otra ley o política que lo impida.

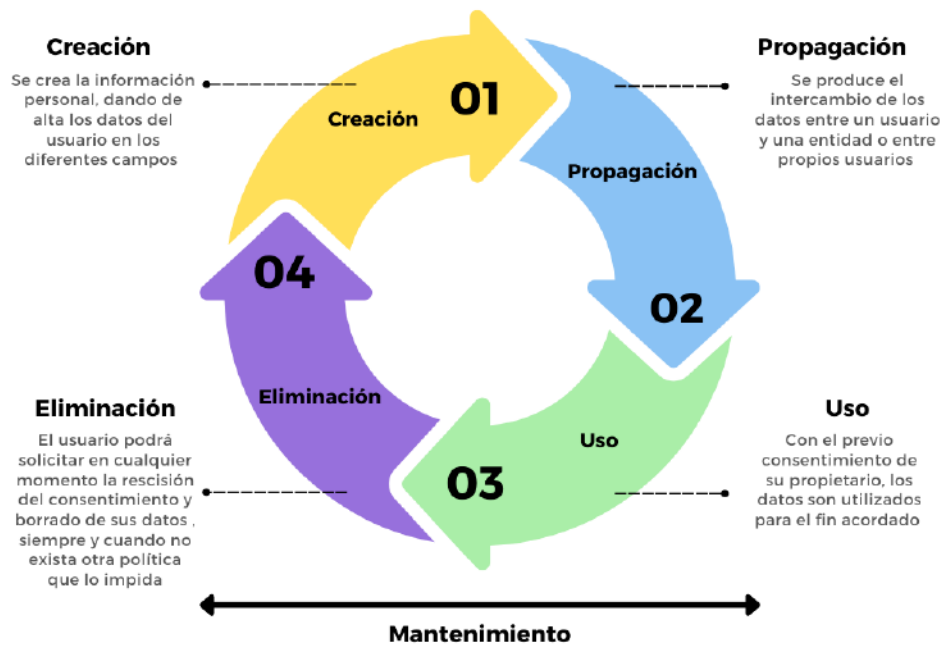


Figura 2.10: Ciclo de vida de la Identidad Digital.

2.2.6 Modelos

La estructura de la identidad digital de un usuario puede ser distinta en función del modelo a seguir. Principalmente, podemos encontrar tres tipos de modelos distintos:

Modelo Centralizado

El modelo centralizado se compone de un ente central como una institución o entidad quien se encarga de emitir las correspondientes credenciales de sus usuarios. Este es el modelo más frecuente en la actualidad. En general, el ente central tiene poder absoluto sobre los usuarios y puede realizar cualquier acción sobre el sistema. Por otro lado, las funcionalidades o derechos de los usuarios vienen establecidas directamente por la entidad administradora.

De acuerdo con los artículos [32] y [33], este modelo puede clasificarse en dos subtipos, diferenciados principalmente por el territorio geográfico y por la entidad encargada de emitir las credenciales de identificación del usuario:

- **Modelo Escandinavo:** son las empresas privadas (bancos y telecomunicaciones) son las encargadas de proveer los servicios de identidad digital al gobierno y empresas (TUPAS en Finlandia, BankID en Suecia ...).
- **Modelo Occidental:** son los gobiernos quienes proveen a bancos y empresas de las diferentes soluciones de Identidad Digital.

Modelo Federado

El modelo de identidad federado está bastante relacionado con el modelo centralizado, aunque presenta algunas diferencias significativas. Recordemos que en anterior modelo existía una entidad central que proporcionaba las credenciales al usuario con las cuales acceder a su servicio. El modelo federado presenta una solución al usuario para reducir el número de credenciales del usuario en los diferentes servicios utilizados. En este caso, el centro de este modelo de es el proveedor de identidad (ver figura 2.11 [34]), que actúa como puente entre el usuario y el servicio al que accede. Los diferentes servicios permiten el acceso a sus servicios utilizando credenciales de otra web, entorno o empresa, haciendo uso de métodos de autenticación como Single Sign-On [34].

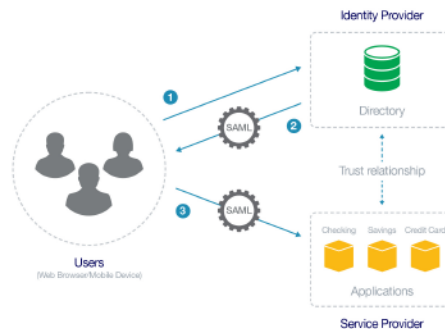


Figura 2.11: Esquema general de la interacción del proveedor de identidad con los usuarios y los servicios

Un ejemplo de este modelo son las credenciales emitidas por Google o Apple, las cuales permiten a un usuario autenticarse en otros servicios web diferentes. Este modelo ofrece una gran comodidad a los usuarios, pero al mismo supone que un robo de este tipo de credenciales permita acceder a todos los servicios que permitan este tipo de autenticación.

Al igual que en el caso anterior, en este modelo, un usuario no es el verdadero propietario de sus datos digitales. La autoría de estos datos corresponde en este caso al proveedor de identidad, el cual tiene un papel muy importante en los procesos de acceso del usuario a los servicios. Esto puede deducirse como un riesgo para la privacidad del usuario, ya que el proveedor de identidad, al igual que en el caso de la entidad central, puede controlar los servicios utilizados con la identidad digital del usuario [33].

Modelo Descentralizado o Self-Sovereign Identity

El modelo descentralizado presenta un nuevo sistema para la gestión la identidad digital centrado en el usuario. En este modelo, el usuario es el único propietario de su identidad digital y de todos los datos relacionados con ella. En el modelo descentralizado no existe ningún tipo de autoridad central ni proveedora, sino que el usuario crea y gestiona su identidad digital de manera independiente. Para ello sustituye el sistema de nombres centralizado de los anteriores modelos por la tecnología Blockchain, a través de la que se crea un nuevo sistema de nombres descentralizado junto con los Identificadores Descentralizados (DID) y las Credenciales Verificables [33] (ver sección 2.3.2).

Las relaciones entre el usuario y los servicios se realizan de manera directa a través de una conexión directa entre pares que elimina al intermediario y la validación de una entidad central (ver figura 2.12). En esta conexión, cada extremo decide las credenciales que va a compartir y la cantidad de tiempo de dicha conexión. Para establecer esta conexión, utiliza la criptografía asimétrica, donde cada dueño de sus claves privadas es dueño de la información asociada a ella y donde se es posible la compartición de las claves públicas sin comprometer la privacidad ni la seguridad de los usuarios [35].



Figura 2.12: Comparación de la estructura del modelo centralizado y el modelo descentralizado.

2.2.7 Problemas de la identidad digital en la actualidad

Los sistemas de identidad digital actuales constan de una serie de desventajas, problemas y dificultades que suponen un gran impedimento para su adopción por el público general. Alex Preukschat, en su artículo “*Self Sovereign Identity: a guide to privacy for your digital identity with Blockchain*” [36], identifica y numera un conjunto de problemas relacionados con la identidad digital y sus métodos actuales:

- **Proximidad:** los usuarios no interactúan físicamente, sino a distancia, digitalmente.
- **Escala:** la identidad digital depende de los grandes centros de información e identidad quienes, en numerosos casos, son dueños de los diferentes datos personales del usuario.
- **Flexibilidad:** las soluciones de identidad digital actuales se limitan a esquemas o conjuntos de atributos fijos, sujetos a pocas modificaciones o cambios que mejoren sus características, metodología e implementación.
- **Privacidad:** las soluciones de identidad digital se basan en una básica colección de datos que identifican al usuario y que, a menudo, se recopilan sin su conocimiento.
- **Consentimiento:** los datos contenidos en miles bases de datos de identidad a menudo son expuestos careciendo de ningún tipo de consentimiento, poniendo gravemente en peligro la seguridad y privacidad del usuario.

El doctor en informática y matemáticas aplicadas Carlos García Moreno, apoyado en el estudio [37] sobre la IA aplicada a la identidad digital creado por Randy Bean, afirma que los problemas más notables y frecuentes del uso de datos personales provienen del desconocimiento del usuario en diferentes cuestiones. En general, los usuarios desconocen donde se almacenan sus datos personales, quien puede acceder a ellos o como se trata la información una vez accedida.

Además de estos problemas generales existen otros inconvenientes que dificultan el uso de la identidad digital en la actualidad. Algunos de los más destacados se analizan a continuación:

Centralización y dependencia de terceros

La centralización de los proveedores de identidades digitales, tanto de los modelos centralizados como modelos federados (ver secciones 2.2.6 y 2.2.6 respectivamente), supone una serie de inconvenientes e impedimentos para el usuario. Como hemos visto, en estos modelos, el usuario no es verdaderamente el propietario de sus datos, sino que su auditoría realmente pertenece a las entidades en la que han sido registrados. Esto supone que los proveedores

pasen a ser un elemento dependiente en cuanto al uso de la identidad digital del usuario, así como su protección y seguridad. Esta dependencia supone que los servicios utilizados con estas identidades proporcionadas sean seguidos y controlados por los emisores de la identidad digital, teniendo capacidad hasta para ser eliminados, pues realmente son los dueños de las credenciales.

Brechas de seguridad y robos de información

El registro de los datos del usuario en cada uno de los servicios o sitios web que accede supone una gran cantidad de puntos vulnerables que ponen en peligro la seguridad y privacidad de sus datos. Frecuentemente surgen nuevas brechas de seguridad en diferentes entidades u organizaciones que almacenan información sensible de los usuarios que se han registrado en ella. Estos datos, a menudo, son puestos en venta en el mercado negro o publicados en Internet, donde cualquier usuario puede consultar esta información. En este punto, la privacidad y seguridad de los datos de los usuarios se ha visto corrompida, sin estos ser responsables de ello.

Procesos de registro repetitivos

Si analizamos la cantidad credenciales que un usuario dispone a día de hoy o la cantidad de sitios web en los que se ha registrado a lo largo de su vida en Internet, es prácticamente imposible tener constancia de todos ellos. La mayoría de las veces un usuario debe registrarse e ingresar sus datos en cada sitio web en el que desea disfrutar del servicio que le proporciona. Esto hace que el usuario deba repetir el mismo proceso para cada web, introduciendo una y otra vez los mismos datos para cada una de ellas.

Replicación de los datos del usuario en Internet

El almacenamiento de los datos de un usuario en cada web en la que se registra supone que estos datos se encuentren replicados en cada base de datos de las diferentes webs. Este sistema es muy ineficiente pues supone almacenar la misma información tantas veces como sitios web utilice el usuario. Añadido a esto, imaginemos que el caso en el que un usuario actualice su email modifique su número de teléfono o la dirección de su vivienda. Para aplicar este cambio, el usuario debería acceder a cada una de las webs en las que se almacena ese atributo y modificarlo. Este proceso abrumador e ineficiente, a pesar de que los modelos federados ponen solución en cierta parte a este problema.

2.3 Identidad Digital Descentralizada (IDD)

Como hemos visto, existen una gran cantidad de inconvenientes en los métodos de identidad digital implementados en la actualidad. Los usuarios demandan un protocolo seguro, de sencillo uso y que le permita tener la capacidad exclusiva de administrar sus datos digitales en Internet. En esta sección, se presenta la identidad digital descentralizada, un innovador sistema de identidad digital que propone solución a gran parte de los inconvenientes de la identidad digital actual.

2.3.1 La solución al sistema actual

La Identidad Digital Descentralizada es una solución de identidad digital creada como alternativa al sistema de identidad digital tradicional. Este nuevo tipo de identidad propone un sistema basado en el modelo descentralizado (ver sección 2.2.6), en el que el usuario es el único dueño de sus datos y el único responsable de la gestión de su identidad digital en Internet.

El objetivo principal de esta solución es unificar las credenciales y los datos personales del usuario en un sistema legal y totalmente válido. En este sistema, el usuario podrá registrar credenciales y datos personales de diferentes ámbitos y servicios. De esta manera, una vez creada y registrada la diferente información del usuario, los proveedores de servicios registrados dentro del ecosistema podrán consultar esta información y verificarla para otorgar acceso sus webs o realizar transacciones (ver figura 2.13).

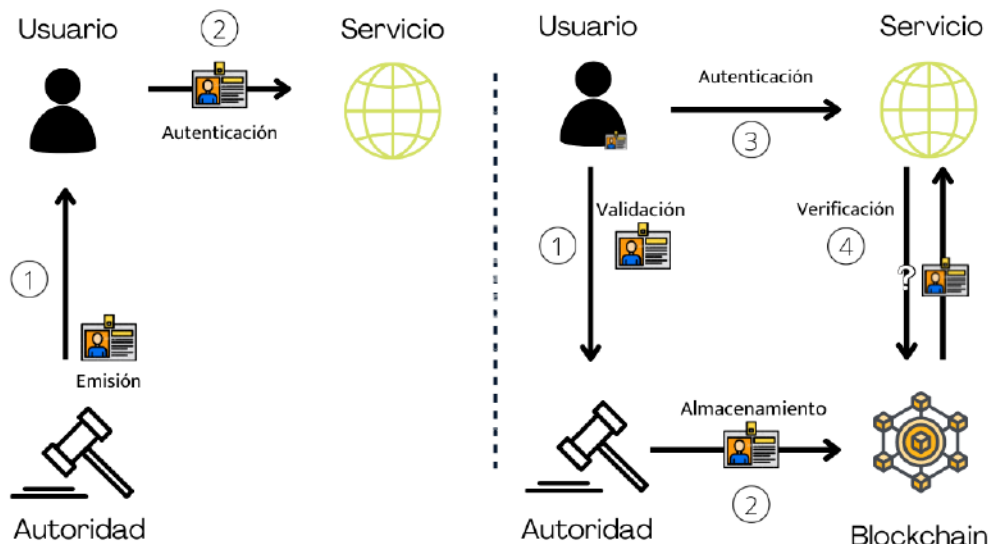


Figura 2.13: Comparación del proceso de emisión y compartición de credenciales entre el sistema Identidad digital convencional y el sistema de Identidad digital Descentralizada.

La IDD, utiliza la tecnología blockchain como base del sistema para el almacenamiento de credenciales y datos registrados por el usuario. Por otro lado, se fundamenta también en la criptografía asimétrica y el uso de las de claves públicas/privadas para acreditar a los usuarios como responsables y poseedores únicos de la información asociada a su clave. Además, se compone de una serie de elementos interoperables para hacer posible su funcionamiento, a continuación, los trataremos en detalle.

2.3.2 Elementos principales

El sistema de identidad digital descentralizada está compuesto de varios elementos, que, en conjunto, conforman este sistema de identidad digital [38]:

Identificadores Descentralizados W3C

Los identificadores como nombres de usuarios o documentos identificativos son reemplazados por otros identificadores descentralizados (DIDs) [39] que identifican las identidades digitales descentralizadas emitidas por los propios usuarios. Estos identificadores se vinculan a sus metadatos de Infraestructura de Clave Pública Descentralizada (DPKI) [40] controlada por ellos mismos. Con ello le permitirá asociar sus datos con los agentes de usuario para realizar intercambios con la cadena de bloques o los libros de contabilidad distribuidos para proteger la privacidad y la seguridad de las transacciones.

Sistemas Descentralizados

Cada identidad descentralizada se encuentra establecida sobre una plataforma blockchain que ofrecerá soporte, mecanismos y todas aquellas herramientas y características necesarias para desarrollar la identidad digital descentralizada en cada caso. Al mismo tiempo, gracias a la integridad, transparencia y adaptabilidad de la tecnología blockchain, el sistema adoptará unas características intrínsecas de la propia tecnología, así como diferentes funcionalidades según el contexto o ámbito para la que se use en cada caso.

Agentes de usuario

Los agentes de usuario son los encargados principales de proporcionar al usuario funcionalidades como la creación, administración, verificación o compartición de su identidad digital. Su función principal consiste en conectar al usuario con el servicio creado en el sistema, permitiendo el uso de las funcionalidades implementadas en este.

Credenciales verificables

Las credenciales verificables engloban el conjunto de información y credenciales del usuario. Estas credenciales tratan de sustituir las credenciales físicas centralizadas, como tarjetas de crédito o documentos de identificación, por credenciales digitales verificables con la misma validez.

Wallets

La función principal de las *wallets* es el almacenamiento y gestión de las credenciales del usuario. Se encarga de proteger la información que el usuario registra en ellas, generalmente a través de agentes digitales. Además, permite realizar copias de seguridad de los datos, transferir claves y credenciales entre diferentes *wallets* digitales o usuarios. Una de sus características más importante es la portabilidad y la estandarización.

Solucionador universal

El solucionador universal actúa como de motor de búsqueda, permitiendo a los usuarios obtener información sobre otros usuarios o transacciones realizadas en la blockchain de manera rápida, transparente y sencilla. Con ello facilita la intercomunicación entre usuarios, así como la creación de transacciones. Al mismo tiempo es posible que implemente servicios de *cloud*, actuando como almacenamiento para dispositivos inteligentes.

2.3.3 Funcionamiento

En el uso de esta tecnología podemos identificar tres subprocesos: la creación de credenciales, su verificación y su gestión. Para ello, existen diferentes elementos que intervienen en cada proceso interactuando entre ellos a través de la red blockchain:

Emisor

El emisor es el componente encargado de emitir las credenciales firmadas para el usuario. Para ello valida y verifica en la blockchain la identificación del usuario dueño de dicha certificación. Una vez esta verificación es exitosa, emite el certificado firmado y se lo asigna al usuario relacionándolo con su *wallet*. Los datos necesarios para la emisión de las credenciales emitidas son solicitados al usuario correspondiente, pues él es el único poseedor de dichos datos y estos no podrán ser transmitidos libremente por la red.

Titular

El titular se corresponde con el usuario final, el usuario que desea compartir sus datos con terceros de una manera privada y totalmente controlada. Es el encargado de registrar sus identificadores personales en la blockchain. La información certificada de cada titular es controlada, administrada y encriptada por el propio usuario mediante su clave privada.

Verificador

Este componente es el encargado de verificar las credenciales del titular. Comprueba que dichas credenciales cumplen unos requisitos establecidos y que al mismo tiempo están certificadas correctamente por un emisor válido. Para ello utiliza la blockchain como método de validación y verificación. Debido a su interoperabilidad, pueden existir varios verificadores que contrasten la información y credenciales necesarios.

2.3.4 Ejemplo práctico

Para facilitar la comprensión de los elementos y su funcionamiento, se analiza el caso de uso de identidad digital descentralizada propuesto por Microsoft [41] (ver figura 2.14).

Imaginemos el caso de un usuario que ha logrado conseguir un título universitario al haber superado todas las asignaturas. Al usuario le interesa obtener una copia digital de dicho certificado, por lo que pide a la escuela de idiomas que le entregue su correspondiente certificado firmado. El usuario, utilizando los agentes de usuario conecta dicho certificado con su identidad digital descentralizada. De este modo, cualquier servicio o entidad podrá comprobar, a través de las claves públicas, que el usuario es poseedor de dicho certificado de una manera legítima y totalmente válida.

- **Paso 1** - La universidad firma la copia digital del certificado y utiliza el Agente de usuario para conectarlo con la identidad respectiva del estudiante.
- **Paso 2** - El agente de usuario almacenará los datos en la bóveda de datos personales del estudiante. Además, el estudiante puede acceder a la información, centros de identidad y el Solucionador Universal utilizando también el agente.
- **Paso 3** - En el caso de que el estudiante deba presentar el certificado a otras entidades, a través de sus claves públicas, comprobarán la validez de la certificación emitida por la universidad y la autoría del usuario.

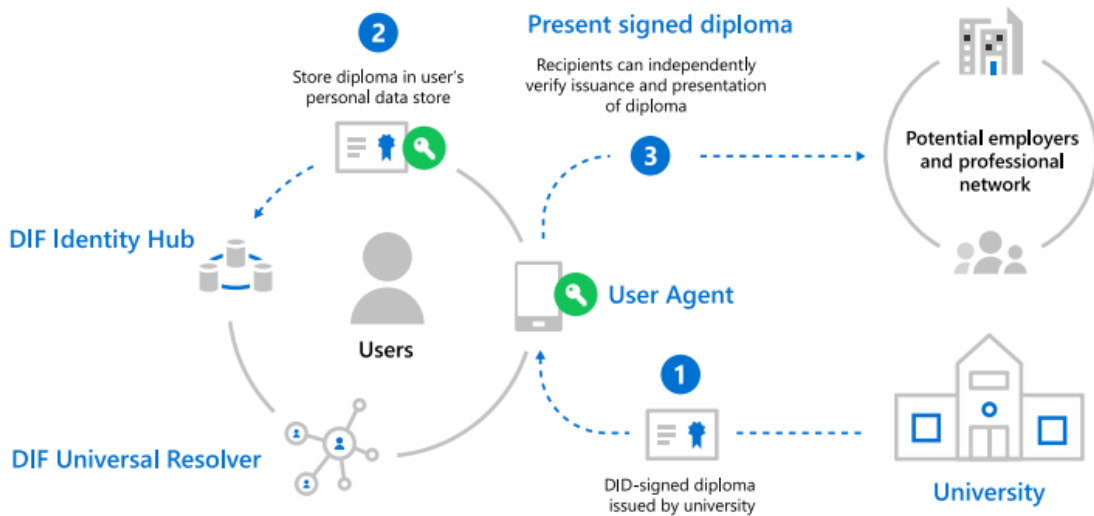


Figura 2.14: Ejemplo propuesto para la emisión, validación y compartición de un título universitario mediante una identidad digital descentralizada.

2.3.5 Principales ventajas de la identidad digital descentralizada

En esta sección se identifican las diferentes ventajas de la identidad digital descentralizada y cómo se benefician de ellas los usuarios.

Descentralización y eliminación de intermediarios

Elimina cualquier tipo de intermediario irrelevante en el proceso de intercambio de datos entre un usuario y una empresa u organización. El intercambio se realiza directamente entre las dos entidades finales sin necesidad de ningún tercero que apruebe, ratifique o permita dicho intercambio.

Inmutabilidad basada en la criptografía

A través de la criptografía asimétrica en la que se basa la tecnología blockchain, es posible relacionar la información y los datos del titular con su clave privada. Esto implica que un cambio en la propiedad física del titular suponga también un cambio en la identidad digital y en su información dentro del espacio digital.

Mayor seguridad y privacidad

La estructura y componentes de la tecnología blockchain proporcionan una gran seguridad y privacidad de los datos del usuario. Debido a que en este sistema de identidad digital, el usuario es tanto el poseedor único de sus datos como el responsable de su gestión, se reducen todos aquellos puntos vulnerables que suponían las empresas que almacenaban la información del usuario. En su lugar, la información únicamente reside bajo la gestión del usuario teniendo un mayor control sobre los datos y una mayor privacidad de estos.

Simplificación de los procesos de registro

La IDD permite eliminar el repetitivo proceso de registro necesario en diferentes entidades para poder utilizar los servicios ofrecidos. En su lugar, aquellas webs o programas de tercero que tengan el consentimiento del usuario podrán solicitar la información requerida del usuario sin necesidad de almacenarla en su sistema.

Interoperabilidad

Este sistema descentralizado propone un gran aumento de la interoperabilidad y escalabilidad entre instituciones, entidades, organizaciones o empresas, así como software o metodologías de autenticación de usuarios. Con un único método globalizado y seguro, en lo referido a seguridad informática y para el propio usuario, facilitaría las distintas relaciones entre las diferentes organizaciones encargadas de las certificaciones digitales y optimizaría también las propias relaciones entre diferentes países logrando así un mayor acercamiento digital a nivel mundial.

2.3.6 Análisis de las soluciones existentes en la actualidad

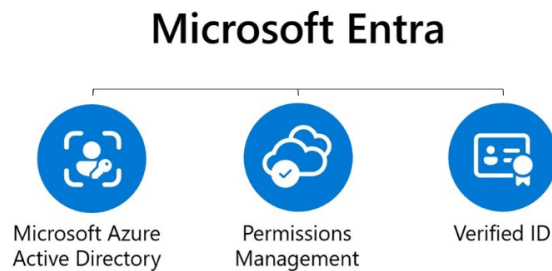
En esta sección, se tratan las diferentes soluciones actuales en el ámbito de la identidad digital basadas en la tecnología blockchain. Para cada una de ellas, se tratan su estructura, funcionalidades y funcionamiento.

Para ello, se organizan las diferentes soluciones en función del tipo de usuario al que están destinadas. Comenzando con Microsoft Entra y DigitalID, dos soluciones destinadas a usuarios u organizaciones que actúen como cliente final, permitiendo crear y gestionar identidades digitales descentralizadas. Por otro lado, soluciones destinadas a desarrolladores como Hyperledger, que ofrecen a los programadores un servicio de identidad digital descentralizada basada en blockchain para implementarlo en sus apps descentralizadas. Por último, el ámbito de la investigación e innovación del campo de la identidad digital basada en blockchain, como Alastria con su herramienta Alastria ID.

Microsoft Entra

Microsoft Entra es la nueva solución que propone Microsoft al ámbito de la identidad digital descentralizada. Se compone de una familia de 3 productos interoperables diseñados específicamente para que los usuarios aseguren su identidad digital:

- **Azure Active Directory:** encargado de proteger la organización y conectar a las personas con sus apps, dispositivos y datos.
- **Microsoft Entra Permissions Management:** herramienta desde la que se administran y supervisan los permisos en la infraestructura del usuario.
- **Microsoft Entra Verified ID:** crea, emite y verifica credenciales descentralizadas que respeten la privacidad y permitan interacciones más seguras con cualquier usuario.



En este caso, nos centraremos en Microsoft Entra Verified ID. Esta herramienta permite la creación de una identidad digital descentralizada para el usuario y consta de otras interesantes funcionalidades, como la verificación y administración de credenciales o diferentes soluciones personalizadas de compartición de datos. Una de las más interesantes es la capacidad de gestión del usuario de administrar el acceso o revocación de los diferentes terceros a sus credenciales.

Además, trata de automatizar la verificación de las credenciales de identidad y permite a los usuarios tener interacciones protegidas y privadas entre ellos. Para ello utiliza claves criptográficas intercambiadas durante la emisión y la verificación, eliminando la necesidad de establecer una federación uno a uno entre el verificador y el emisor.

Aunque estaba previsto su lanzamiento a partir de Julio de 2022, todavía no es posible disfrutar de sus servicios finales. En su lugar, facilita una serie de tutoriales en los que se muestran los pasos para llevar a cabo proceso de configuración [42], la generación de credenciales verificables [43] o su posterior comprobación [44] de esta versión en desarrollo. También podemos encontrar el *datasheet* de la solución, así como los enlaces a la documentación general proporcionada por Microsoft [45].

DefinitiveID - Wise Security

DefinitiveID es un sistema de identidad digital global desarrollado por la empresa española Wise Security [46] y basado en el modelo AlastriaID (ver sección 2.3.6). Ofrece una solución de identidad digital descentralizada basada en blockchain y el estándar internacional DID [39], el cual identifica personas, negocios y dispositivos en una arquitectura privada y segura para el usuario. Este sistema permite a los usuarios identificarse en una o varias plataformas con un único ID descentralizado. Su funcionamiento consiste en una app desde la que es posible administrar la identidad digital descentralizada del usuario. El proceso de creación se divide en 4 etapas, como podemos observar en la figura 2.15:

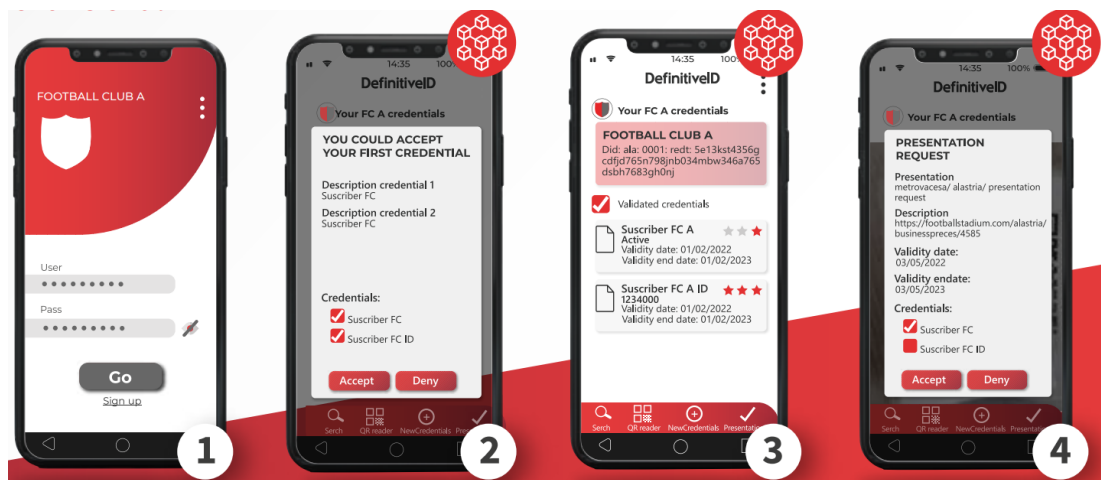


Figura 2.15: Etapas del proceso de creación de una IDD con la solución DefinitiveID.

- 1 - El usuario inicia sesión con sus credenciales en la aplicación la organización.
- 2 - Se inicia el proceso de generación de credenciales con DefinitiveID, en el que registra los datos del usuario y genera un DID universal que será registrado en la blockchain.
- 3 - Una vez creadas, el usuario podrá una lista de las diferentes credenciales generadas y registrados en la interfaz principal de DefinitiveID.
- 4 - Una vez generadas las credenciales, los terceros podrán realizar peticiones de datos al usuario para solicitar sus credenciales.

DefinitiveID hace principal hincapié en la seguridad y la privacidad del usuario, por lo que ningún dato de carácter personal es almacenado en la blockchain, ni siquiera cifrado. En su lugar, se almacena el identificador generado único para cada usuario y los datos del usuario se almacenan en la app de cada usuario.

Hyperledger Sovereign Identity Blockchain Solutions

Hyperledger Sovereign Identity Blockchain Solutions es una solución propuesta por Hyperledger destinada a la creación y gestión de apps para la gestión de identidad digital descentralizada basada en blockchain. Se compone de 3 proyectos interoperables (ver figura 2.16) como Hyperledger Aries, encargado del nivel de aplicación, la gestión y creación de agentes; Hyperledger Indy, el cual proporciona la infraestructura de blockchain necesaria sobre la que actúa Aries; y Hyperledger Ursa, un conjunto de bibliotecas criptográficas que permiten abstraer su implementación en los niveles superiores.



Figura 2.16: Componentes de la solución Hyperledger Sovereign Identity Blockchain.

Hyperledger Aries

Aries es un proyecto aceptado en Hyperledger en marzo de 2019 que nace con el objetivo de implementar un sistema que permita el despliegue de agentes y credenciales verificables en múltiples ecosistemas. Es una infraestructura para interacciones *Peer to Peer* basadas en blockchain que se define en *Trust over IP Technical Stack, Layer 2* (comunicaciones seguras peer to peer) y *Layer 3* (protocolos de intercambio de datos).

Esta herramienta define los diferentes protocolos de mensajería necesarios implementados en un *toolkit* reutilizable e interoperable, diseñado para iniciativas y soluciones centradas en crear, transmitir, almacenar y utilizar credenciales verificables. En su núcleo se encuentran los protocolos que permiten la conectividad entre agentes que utilizan mensajería segura para intercambiar información.

La arquitectura Aries define cómo conectar las diferentes implementaciones en un agente para admitir Indy y otras implementaciones de Identidades Digitales Descentralizadas y credenciales verificables al mismo tiempo. Una parte de la definición de Aries también engloba el almacenamiento de datos y la importación y exportación de estos datos desde entre sistemas.

Hyperledger Indy

Hyperledger Indy es un libro de contabilidad distribuido (DLT) que ofrece de herramientas, bibliotecas y componentes reutilizables diseñados específicamente para la identidad descentralizada concienzuda con el cumplimiento de los estándares exigidos por GDPR [47]. Este sistema se compone de diferentes herramientas, servicios y funcionalidades reutilizables para proporcionar identidades digitales enraizadas en blockchains u otros registros distribuidos. Uno de los principios clave de Hyperledger Indy es su enfoque de “privacidad por el diseño”. Dada la naturaleza inmutable de la tecnología blockchain, es muy importante que las identidades digitales sean administradas con sumo cuidado, manteniendo los valores humanos en el centro del foco. Hyperledger Indy ofrece una serie de características principales:

- Creado única y exclusivamente para una identidad digital descentralizada.
- Identificadores Descentralizados (DIDs) v1.0 únicos a nivel mundial [39].
- Correlación resistente basada en el diseño.
- Credenciales verificables en un formato interoperable para el intercambio de atributos y relaciones digitales (en proceso de estandarización en el W3C).

El proyecto consta principalmente de dos repositorios que contienen herramientas diferentes:

- **Indy Client:** Contiene el software que permite crear clientes de Indy para interactuar con las tecnologías distribuidas y realizar un seguimiento de las claves y otros datos relacionados con la identidad.
- **Indy Node:** Conformar la sección del componente blockchain de Indy. Escrito en Python, contiene el código base que incorpora todas las funcionalidades necesarias para ejecutar nodos que conformen una red blockchain. Se divide en dos repositorios principales:
 - **Indy-Plenum:** contiene la mayor parte del código del libro distribuido para Indy, incluida la implementación de consenso.
 - **Indy-Node:** administra las transacciones que permiten leer y escribir en el libro mayor.

Una de las funcionalidades más interesantes que propone Hyperledger Indy es la compartición selectiva. Implementa el mecanismo *Zero-Knowledge proof* a través de las llamadas *Anoncreds*. Este mecanismo permite el intercambio de información selectiva fiable y verificada sin necesidad de compartir el resto de los datos de la identidad digital. Un ejemplo útil,

(ver figura 2.17) es la prueba de que una persona es mayor de una edad determinada según la fecha de nacimiento en una licencia de conducir con credencial verificable. De esta manera cualquier servicio podría consultar esta condición si sin divulgar ninguna otra información del usuario, como la fecha de nacimiento.

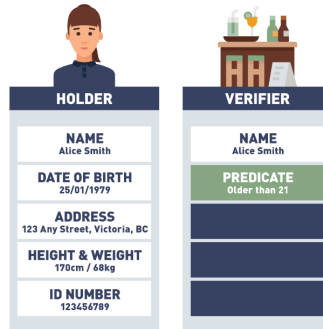


Figura 2.17: Ejemplo de *AnonCreds* para la verificación de la mayoría de edad de una persona.

Hyperledger Ursa

El proyecto Hyperledger Ursa se encarga de proporcionar las herramientas criptográficas necesarias para utilizar diferentes funciones en aplicaciones de nivel superior. Consiste en una serie de bibliotecas modulares que permiten la abstracción del aspecto criptográfico utilizado por Indy y Aries, delegando las funciones relacionadas con la criptografía a este proyecto.

Ursa toma los algoritmos o funciones sin procesar y los empaqueta de acuerdo con la implementación necesaria para que puedan integrarse fácilmente y de manera segura en proyectos superiores. Con ello, consigue abstraer la implementación a bajo nivel de las diferentes funciones, de manera que al crear aplicaciones sobre Aries o al utilizar el libro mayor proporcionado por Indy, parte de las diferentes utilizadas se delegarán a las diferentes bibliotecas de Ursa. Algunos de estos paquetes incluyen:

- Generado de pares de claves pública/privada, firma y verificación.
- Encriptación y desencriptación de datos.
- Verificación y generado de hash de datos.
- Tecnología de prueba de conocimiento cero (ZKP), incluyendo la emisión y revocación de credenciales ZKP y la generación y verificación de ZKP.

Alastria

Alastria [48] es una asociación sin ánimo de lucro compuesta por administraciones públicas como el Gobierno de España, empresas privadas como Telefónica, Iberdrola, Deloitte o múltiples universidades españolas. Estas entidades cooperan entre sí para fomentar la economía digital a través de la creación de un nuevo ecosistema que facilite el desarrollo de sistemas descentralizados basados en la tecnología blockchain. Esta asociación promueve la colaboración entre los miembros para acelerar el desarrollo de una infraestructura común que sirva como base para las aplicaciones del futuro. Para ello propone un nuevo ecosistema compuesto por 4 redes distintas: la red T, la red B, la red H y la reciente red H+.

Alastria ID es un modelo de identidad digital inspirado en el concepto de *Self Sovereign identity (SSI)*. Se fundamenta en 3 pilares fundamentales: seguridad, la información debe guardarse de forma segura; controlabilidad, el usuario debe tener el control total de sus datos y debe poder ver quien puede y tiene acceso a ellos; y portabilidad, el uso de los datos no debe estar sujeto a un solo proveedor, sino debe poder utilizarse donde quiera. Este nuevo modelo permitirá que los ciudadanos dispongan del total control sobre su información personal, permitiendo la realización de transacciones dotadas de validez legal, siguiendo las directrices que marca la Unión Europea y la regulación española.

Este modelo opera en la anteriormente comentada Red T, basada en *Quorum* [49]. Su funcionamiento consiste en la implementación de contratos inteligentes y componentes software que permitirán su integración con los *Back-Ends* de diferentes servicios desde su aplicación móvil (ver figura 2.3.6).

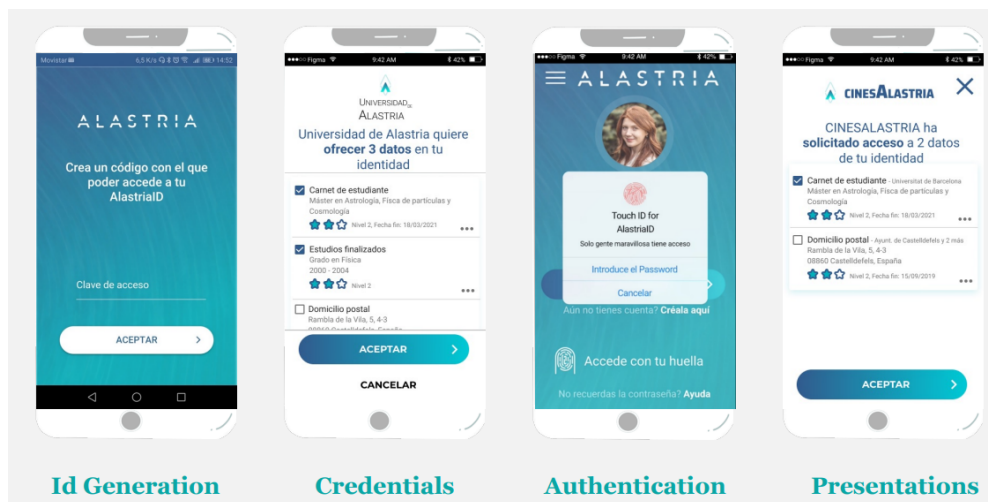


Figura 2.18: Ejemplo de las diferentes interfaces de la app de Alastria ID.

A través de esta app, los usuarios tendrán el control sobre las transacciones asociadas a su identidad y podrán compartir sus datos con total conocimiento cuando estos sean solicitados por los diferentes servicios. Además, un usuario también podrá crear su identidad digital, generar DIDs y compartir información con diferentes proveedores, haciendo uso de la criptografía asimétrica y funciones hash para proteger la privacidad de sus datos.

En la figura 2.19 podemos observar un ejemplo detallado de los diferentes pasos que componen el proceso en el que se realiza una transacción de datos donde un usuario utiliza Alastria ID para un fin concreto, como alquilar un vehículo:

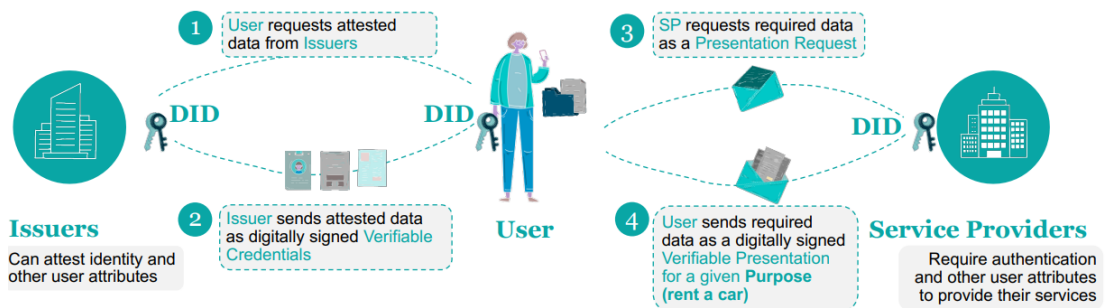


Figura 2.19: Proceso de compartición de datos con Alastria ID.

Podemos encontrar información más detallada sobre la explicación de este caso de uso implementado en la fuente proporcionada por Alastria [50].

En su página de github [51], podemos encontrar información más detallada la implementación MVP que hemos comentado. También permite consultar la documentación detallada de los diferentes componentes y su implementación. Además, incluye un tutorial donde muestra el proceso de despliegue de sus componentes acompañado de una explicación y una documentación del caso de uso implementado.

Metodología y Planificación

Este trabajo se ha realizado siguiendo una planificación determinada y haciendo uso de metodologías de ingeniería existentes. En esta sección se presentan las diferentes fases en las que se ha estructurado el trabajo y las metodologías utilizadas para su realización.

3.1 Metodología

Para la realización de este trabajo se ha optado por una metodología ágil basada en el desarrollo incremental e iterativo. Este tipo de metodología se caracteriza por la división de un proyecto en varias fases, identificando las fases principales y desglosando cada una de ellas en otras fases secundarias más concretas. Una vez identificadas y planificadas todas estas fases, se pasa con la implementación de cada una de ellas, comenzando por los conceptos más básicos y añadiendo mejoras en cada interacción implementada. Cada interacción se centra en desarrollar una etapa en concreto, y una vez finalizada, se continuará con la siguiente.

Este tipo de metodología aporta una serie de ventajas clave para el desarrollo de cualquier proyecto. La flexibilidad y adaptabilidad a cambios, gracias a la división del proyecto en varias fases independientes; la reducción de riesgos, debido a la constante evaluación durante el desarrollo individual de cada fase; o la mejora de la calidad del proyecto, en vista del incremento de las mejoras implementadas en cada una de las iteraciones llevadas a cabo.

Para aplicar este tipo de metodología, primeramente, se ha realizado una planificación inicial general, en la que se han identificado los puntos principales del proyecto, así como los sub apartados de los que se componen. Principalmente se identifican dos ámbitos, el teórico y el práctico, representados por la investigación y el análisis de la identidad digital descentralizada y la implementación de la prueba de concepto respectivamente.

En el caso de la investigación del tema principal, se han dividido en tres secciones distintas, con sus correspondientes conceptos en cada una de ellas. Primeramente, introduciendo los temas de blockchain e identidad digital para posteriormente tratar las cuestiones relativas a la identidad digital descentralizada.

Por otro lado, en cuanto al desarrollo de la prueba de trabajo, se divide en las diferentes fases de planificación del desarrollo software, como el diseño, el análisis de funcionalidades, la implementación y la etapa de test. Concretamente, para la realizar la implementación se ha seguido una metodología basada en el desarrollo incremental iterativo. Siguiendo este modelo se ha comenzado con la implementación de las estructuras y funcionalidades básicas y se han ido añadiendo nuevas funcionalidades en cada nueva iteración realizada.

3.2 Planificación

Como hemos comentado en la anterior sección para llevar a cabo la metodología escogida en el desarrollo de este proyecto, es necesario realizar una planificación inicial de las diferentes fases del proyecto. En este caso, ha sido realizada a través de Microsoft Project [52], creando diferentes tareas y subtareas con cada fase a ejecutar. Como podemos ver en la imagen 3.1, se identifican diferentes fases clave, como la recopilación de información, la documentación de la parte de investigación u otras relacionadas con el desarrollo de la prueba de concepto implementada.

En cuanto a la estimación temporal para la ejecución de las diferentes fases, se ha estimado un margen bastante grande debido a que disponía de bastante tiempo. En algunos casos se ha sobrestimado el tiempo necesario para realizar algunas tareas, como en el caso de la implementación del *Front-End*. En cambio, otras tareas como la implementación de nuevas funcionalidades han supuesto una cantidad de tiempo mayor que la estimada inicialmente.

El seguimiento de la planificación se ha actualizado frecuentemente. Al finalizar cada fase o tarea, se ha añadido o reducido el tiempo estimado empleado en ella mediante las dependencias entre las tareas. De este modo se puede comparar el tiempo estimado y empleado finalmente para cada tarea. Al mismo tiempo se han añadido nuevas tareas identificadas posteriormente a la planificación. En la figura 3.1 es posible observar la planificación final actualizada del proyecto.

CAPÍTULO 3. METODOLOGÍA Y PLANIFICACIÓN

Modo de	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
★	<i>Researching de propuestas</i>	3 días	vie 04/02/22	mar 08/02/22	
→	➤ Investigación y ámbito teórico	102 días?	vie 11/02/22	lun 30/05/22	
→	➤ Recopilación de información	49,33 días	vie 11/02/22	lun 04/04/22	
→	Estudio de la tecnología blockchain	12 días	vie 11/02/22	mié 23/02/22	1FC+2 días
→	Estudio de la identidad digital	17 días	jue 24/02/22	lun 14/03/22	4
→	Análisis de soluciones SSI existentes	20 días	mar 15/03/22	lun 04/04/22	5
→	➤ Investigación y documentación teórica	14 días?	lun 16/05/22	lun 30/05/22	
→	Introducción Blockchain	4 días?	lun 16/05/22	mié 18/05/22	
→	Identidad digital	6 días	jue 19/05/22	jue 26/05/22	8FC+1 día
→	Soluciones identidad digital descentralizada con blockchain	3 días?	jue 26/05/22	lun 30/05/22	9
→	➤ Implementación de la prueba de concepto	100,67 días?	lun 30/05/22	vie 09/09/22	
→	➤ Diseño	14 días?	lun 30/05/22	lun 13/06/22	
→	Aspectos generales (lenguajes, tecnologías, modelos...)	4 días?	lun 30/05/22	jue 02/06/22	10
→	Blockchain	3 días?	lun 06/06/22	mié 08/06/22	13FC+2 días
→	Smart Contracts	5 días?	mié 08/06/22	lun 13/06/22	14
→	➤ Desarrollo e implementación	60,67 días?	mié 15/06/22	mar 16/08/22	
→	➤ Implementacion Backend	58,67 días?	mié 15/06/22	lun 15/08/22	
→	Servicio Web App	2 días?	mié 15/06/22	jue 16/06/22	15FC+2 días
→	Servicio de autenticación	3 días?	jue 16/06/22	lun 20/06/22	18FC-1 día
→	➤ Blockchain	22 días?	lun 25/07/22	lun 15/08/22	
→	Integración de blockchain con el backend	2 días?	lun 25/07/22	mar 26/07/22	19FC+2 días
→	Programacion y despliegue de Smart Contracts	7 días?	jue 28/07/22	mié 03/08/22	21FC+2 días
→	Modificación de la implementación del sistema	7 días?	lun 08/08/22	lun 15/08/22	32FC+2 días
→	➤ Implementacion frontend	4 días	vie 12/08/22	mar 16/08/22	
→	Interfaz de usuario y CSS	4 días	vie 12/08/22	mar 16/08/22	23FC-2 días
→	Testing y corrección de errores	3 días	mié 31/08/22	vie 02/09/22	34
→	➤ Documentación Prueba de Concepto	5 días?	mar 06/09/22	vie 09/09/22	
→	Diseño	2 días?	mar 06/09/22	mié 07/09/22	26FC+2 días
→	Funcionalidades	2 días?	mié 07/09/22	vie 09/09/22	28
→	Resultados	3 días?	mié 07/09/22	vie 09/09/22	29FC-2 días
→	➤ Tareas no programadas inicialmente	38 días	mié 03/08/22	lun 12/09/22	
→	Identificación de funcionalidades y coreccion de la memoria	2 días	mié 03/08/22	jue 04/08/22	22
→	Implementación de nuevas funcionalidades	7 días	lun 22/08/22	lun 29/08/22	23FC+7 días
→	Implementación del sistema externo de registro de usuarios adicional	2 días	mar 30/08/22	mié 31/08/22	33
→	Modificaciones de la documentación	25 días	mié 17/08/22	lun 12/09/22	25

Figura 3.1: Planificación final actualizada.

Prueba de concepto

En este capítulo se detalla toda la información relacionada con la implementación de la prueba de concepto. A lo largo de él, se tratan los diferentes aspectos principales de la prueba de concepto implementada, como la arquitectura y el diseño (ver sección 4.2) o las diferentes funcionalidades de cada sistema (ver sección 4.3).

4.1 Introducción

La prueba de concepto implementada consta principalmente de un prototipo de un sistema de gestión de identidad digital descentralizado y auto gestionable, desde el cual el usuario puede registrar, gestionar y compartir su información personal con otros sistemas web de terceros que la soliciten. Para ello utiliza la tecnología blockchain y los *Smart Contract* como método de almacenamiento y compartición de los datos personales del usuario. En este documento, nos referimos a este sistema como Sistema de Gestión de Identidad Digital del Usuario (SGIDU).

El uso de la tecnología blockchain en este caso viene fundamentado por las diferentes ventajas que aporta con respecto a otros sistemas de almacenamiento y compartición de datos, como las bases de datos tradicionales. La ventaja principal es la capacidad de descentralización de la información y auto gestión de los datos por parte del usuario, eliminando cualquier tipo de intermediario entre el usuario y los servicios finales. Esto otorga al usuario la auditoría única de sus datos y el control total sobre ellos. Por otro lado, también permite la integridad de la información registrada gracias a la garantía de la inmutabilidad que ofrece la estructura de la blockchain blockchain. Y finalmente, el despliegue de los *Smart Contracts*, que permiten el almacenamiento de información, la automatización de procesos y el cumplimiento de los términos y condiciones registrados en él.

El objetivo de esta prueba de concepto es implementar un prototipo en un entorno de pruebas que nos permita interactuar con la blockchain, consultar y registrar datos, crear nuevas transacciones, programar y desplegar *Smart Contracts* y hacer uso de algunas de las herramientas y *frameworks* disponibles para esta tecnología. Con ello no se busca implementar una solución funcional real, ya que excede la complejidad y el objetivo de esta prueba de concepto. Aunque se ha simplificado el sistema para su desarrollo, a lo largo de este capítulo se comenta cómo se plantearía este sistema en un entorno real de despliegue.

Para hacer uso de este Sistema de Gestión de Identidad Digital del Usuario (SGIDU), se implementa adicionalmente un Sistema Externo de Registro de Usuarios (SERU) que permite hacer uso de las diferentes funcionalidades del SGIDU. De esta forma, la arquitectura propuesta para esta prueba de concepto (ver figura 4.1) cuenta con dos tipos de sistemas: el SGIDU, desde donde el usuario registra y gestiona sus datos; y los SERUs, servicios en Internet compatibles con esta tecnología. Es importante remarcar que cualquier SERU que funcione con la arquitectura propuesta debe ser compatible con el SGIDU.

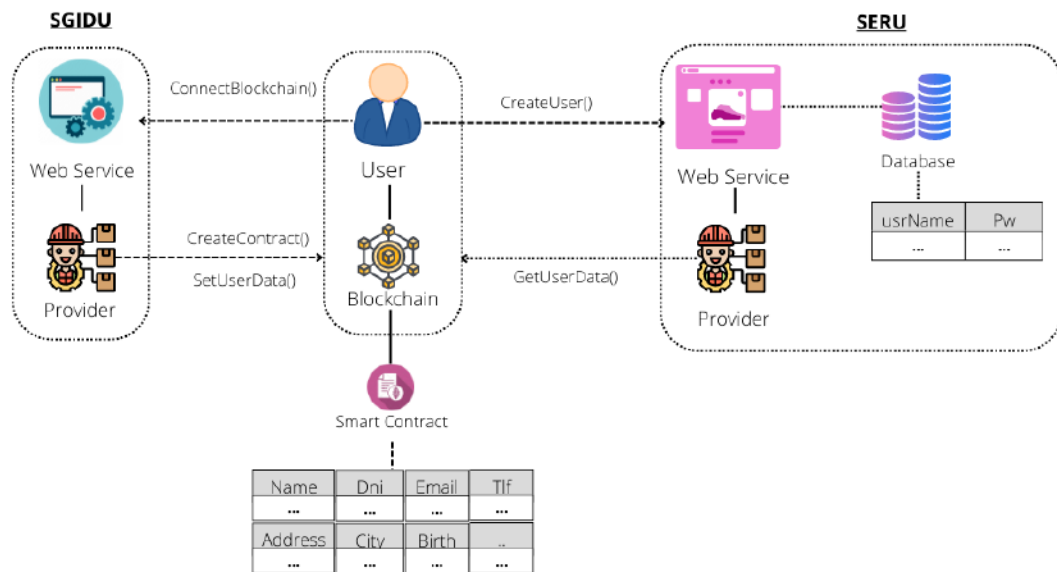


Figura 4.1: Arquitectura general de la prueba de concepto desarrollada. El SGIDU registra los datos personales en la blockchain del usuario y permite su gestión auto-soberana; y los SERUs son servicios en Internet compatibles con este sistema de identidad digital.

Cada SERU implementa un innovador método de registro y consulta de datos que permite al usuario crear una nueva cuenta de usuario sin necesidad de indicar sus datos personales. Para poder consultar la información personal registrada en la blockchain del usuario, los SERUs realizan peticiones de consentimiento para poder acceder a diferentes atributos del usuario. Cuando el usuario acepta esta solicitud, registra este consentimiento para que posteriormente el SERU pueda acceder a los diferentes atributos del usuario en cada caso. Los SERUs utilizan bases de datos para registrar la información de los diferentes usuarios creados. En ellas, únicamente se almacenan los datos necesarios para autenticar al usuario (usuario y contraseña) y los parámetros necesarios para realizar consultas a su blockchain correspondiente (URL, puerto y *wallet*).

En esta prueba de concepto se proponen dos ejemplos de SERU: “Pc Shop” y “Biblioteca”. Ambos interactúan con la blockchain del usuario para obtener los datos a los que el usuario autoriza el acceso. De esta forma, cada SERU tiene acceso a los atributos autorizados para ese SERU, previamente registrados por el usuario en la blockchain a través del SGIDU. El motivo de esta replicación es disponer de dos ejemplos de SERU que accedan a atributos diferentes de la identidad digital del usuario para poder probar todas las funcionalidades del SGIDU.

4.2 Diseño

Como hemos visto en la figura 4.1, la prueba de concepto consta de dos tipos de sistemas diferentes: el Sistema de Gestión de Identidad Digital del Usuario (SGIDU), que utiliza la blockchain para almacenar y compartir la información personal del usuario; y (2) los Sistemas Externos de Registro de Usuarios (SERU), que consultan esta blockchain para los datos personales y gestionan un servicio web independiente. Cada tipo de sistema se compone de una serie de elementos que permiten realizar las diferentes funcionalidades programadas en ellos. A continuación, se presenta el diseño, arquitectura y componentes de cada uno.

4.2.1 Sistema de Gestión de Identidad Digital del Usuario (SGIDU)

El SGIDU permite al usuario almacenar sus datos personales en su blockchain. Para ello el usuario debe desplegar previamente y conectar su blockchain al sistema, introducir la información que desea almacenar y el SGIDU desplegará un contrato inteligente en su blockchain con la información registrada. También permite visualizar la información almacenada en el contrato o actualizar los datos personales almacenados. El SGIDU cuenta con cuatro componentes principales como podemos ver en la figura 4.2: el Servicio Web, la Blockchain del usuario, el *Provider* y el *Smart Contract*.

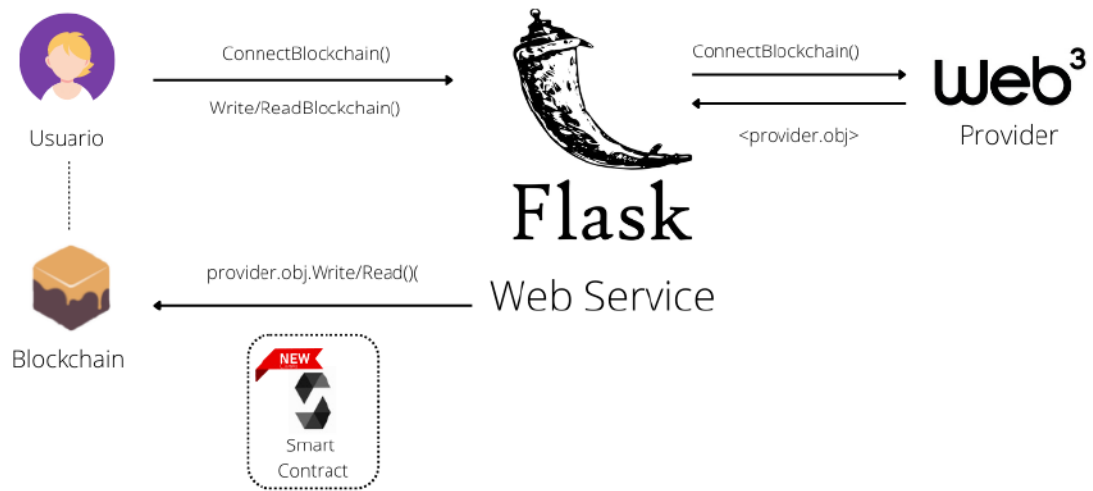


Figura 4.2: Arquitectura y componentes del SGIDU.

Servicio Web

El servicio web es el componente encargado del *Back-End* y *Front-End* del sistema. Es un *framework* diseñado para programar servicios web dinámicos con los que el usuario puede interactuar mediante las diferentes interfaces programadas en él. Este componente actúa como nexo entre los otros componentes del sistema como la blockchain o el *provider* y permite al usuario interactuar con ellos a través de las distintas vistas. Su importancia en el sistema es fundamental, pues es el punto de entrada que recibe las acciones por pantalla del usuario, las interpreta y construye las correspondientes respuestas y llamadas desencadenantes en los otros componentes.

En este caso, el *framework* elegido para implementar el sistema ha sido Flask. Para la implementación de este componente se plantearon dos opciones principalmente: Flask [53] o Django [54]. El motivo de elección de Flask se debe a las características y necesidades del proyecto. Principalmente porque está destinado a ser usado en aplicaciones pequeñas, sencillas o con un grado de complejidad bajo, a diferencia de Django. Por otro lado, ofrece una mayor flexibilidad de codificación, la incorporación de extensiones y elementos personalizables, además de un método sencillo e intuitivo para implementar las diferentes vistas e interfaces. Esto ha permitido dedicarle una mayor cantidad de tiempo a los elementos de interés del sistema, como la blockchain o los contratos inteligentes.

Blockchain

Como hemos introducido anteriormente, la tecnología blockchain tiene ciertas ventajas respecto a otros sistemas tradicionales de almacenamiento como la trazabilidad, la inmutabilidad o el uso de los contratos inteligentes automatizados. Pero, la ventaja principal de los sistemas blockchain frente al resto es la descentralización. La descentralización es un valor clave para el usuario, pues le permite interactuar con los usuarios finales de manera directa, sin un intermediario que condicione la compartición o la gestión de su identidad digital. Además, garantiza al usuario ser el poseedor único de sus datos, sin estar condicionado a cumplir las diferentes políticas impuestas por una organización centralizada como un gobierno o una empresa. Con ello, al mismo tiempo, supone que el usuario sea el único responsable de la gestión de sus datos, así como la compartición con los diferentes servicios de terceros.

Realizando una comparación de un caso de uso similar en el que se utilizase una base de datos, no tendríamos de la descentralización de la información del usuario. En su lugar, los datos serían almacenados en bases de datos centralizadas de un proveedor de servicio. Esto supone que, en el caso de una pérdida de servicio de este proveedor o un borrado de la información, el usuario no podría interactuar con el sistema ni acceder a los datos.

En el caso de la blockchain, distribuye la información registrada en la cadena de bloques entre los diferentes nodos de la red, de manera que cada nodo obtiene una copia exacta de la cadena. En caso de que un nodo se caiga o pierda la información, cualquier otro nodo puede enviarle una copia sin producir pérdidas de servicio. En el caso de las bases de datos, tampoco contarían con la integridad de la información almacenada, pues en caso de acceso de una intrusión en el sistema, el atacante podría modificar la información.

En un entorno real, el tipo de blockchain óptimo para este sistema sería el privado. Este tipo de blockchains, como hemos visto en la sección 2.1.6, se caracteriza por capacidad de gestionar los nodos que acceden a la cadena y atribuirles diferentes roles y permisos para realizar determinadas funcionalidades en la red. En este caso, existirían nodos *permisionadores* que gestionarían el acceso a la cadena, nodos con roles de escritura sobre la cadena de bloques u otros nodos con únicamente permisos de lectura.

Un escenario de producción real similar es el de DefinitiveID (ver sección 2.3.6). Este sistema se basa en el modelo de Alastria ID, utilizando una blockchain privada con un gran número de nodos en la que la información registrada en la cadena de bloques se encuentra distribuida entre los diferentes nodos que componen la red.

El registro de la información personal del usuario se realiza en la propia de aplicación, de manera que ningún dato personal del usuario se almacena en la cadena, ni siquiera cifrado. En su lugar, al registrarse o iniciar sesión en un servicio de terceros, se generan unas Credenciales Verificables y un Identificador Descentralizado (DID). Este ID permite identificar las credenciales del usuario, y tras ser generado, se registra en la cadena de bloques. Al registrar este valor, se replica la nueva cadena actualizada con este valor al resto de nodos de la red para proteger la integridad del indicador y hacerlo inmutable en el tiempo (ver figura 4.3). Para compartir los datos del usuario, los diferentes servicios pueden solicitar las Credenciales Verificables mediante solicitudes de consentimiento de acceso que el usuario puede aceptar, denegar o incluso revocar una vez aceptadas.

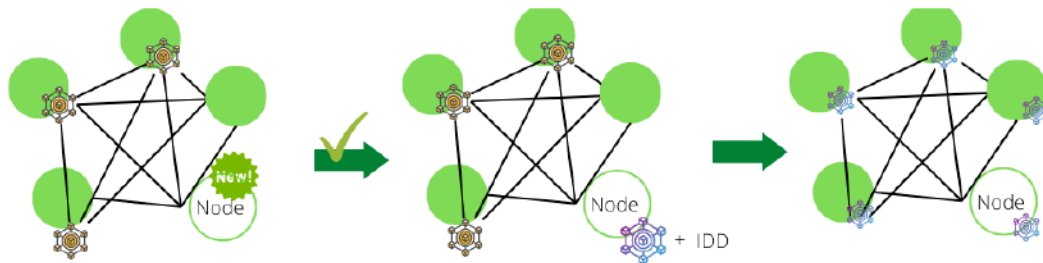


Figura 4.3: Ejemplo de anexión a la cadena y registro del Identificador Descentralizado.

El sistema de DefinitiveID es una solución en producción, por lo que es muy compleja en cuanto a estructura e implementación y resulta demasiado extenso y fuera del objetivo de esta prueba de concepto. Como hemos comentado anteriormente, el objetivo de esta prueba de concepto es interactuar con la tecnología blockchain y probar las diferentes funcionalidades y herramientas de la tecnología.

Para la prueba de concepto implementada, se ha simplificado tanto la arquitectura como el componente de la blockchain. En este caso, se ha escogido la blockchain de Ethereum, principalmente por el despliegue de los *Smart Contract*. Así mismo se ha utilizado la solución de Ganache para realizar el proceso de creación y despliegue de la red y la cadena de bloques.

Ganache [55] es una herramienta desarrollada por *Truffle Suite* que permite realizar un despliegue de una cadena de bloques de Ethereum en un entorno local de pruebas, para interactuar con ella y probar el despliegue de *Smart Contract*. En este caso, la red se desplegará con únicamente un nodo, es importante remarcar que esta red blockchain es una simplificación, adaptada a la prueba de concepto.

Para esta prueba de concepto, el usuario debe desplegar una blockchain con Ganache y conectarla con el SGIDU para poder almacenar su información personal en la cadena de bloques. Para almacenar la información del usuario se utiliza un *Smart Contract* (ver sección 4.2.1), en el que se definen las estructuras de datos en las que se almacenan los diferentes atributos de la información que el usuario registra en la cadena de bloques. Este contrato es creado y desplegado la primera vez que el usuario conecta su blockchain al SGIDU y registra sus datos en la cadena de bloques (ver figura 4.4).

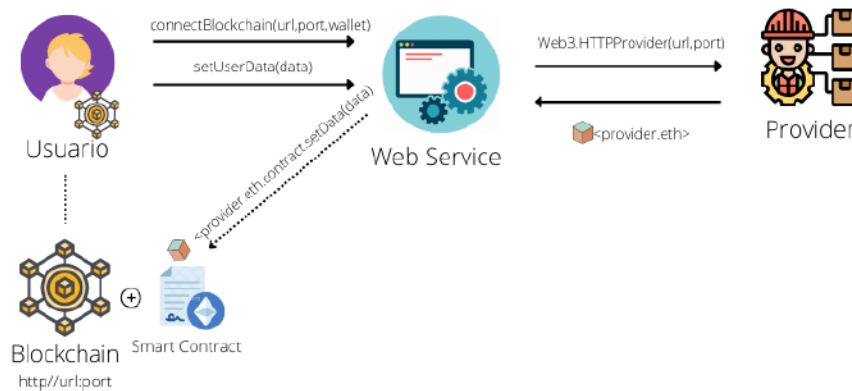


Figura 4.4: Despliegue del *Smart Contract* y registro de datos del usuario.

La blockchain, además de registrar la información del usuario, también permite compartirla. En este caso con los diferentes SERU que han sido implementados en esta prueba de concepto. Un SERU es un sistema externo que interactúa con el sistema desarrollado. Para esto, los SERUs deben ser compatibles con el SGIDU. En esta prueba de concepto, se han desarrollado dos sistemas SERU. Cuando el usuario crea una nueva cuenta en uno de los SERUs debe indicar, además de sus credenciales, los parámetros necesarios para que puedan consultar la información registrada en su blockchain. Para ello, los SERUs deberán solicitar el consentimiento del usuario para acceder a los diferentes parámetros de su información, este consentimiento será registrado también en el contrato inteligente desplegado anteriormente. Comentaremos este sistema en la correspondiente sección dedicada a los SERUs (ver sección 4.2.2).

Esta es una solución funcional para implementar prueba de concepto simplificada y probar la compartición de datos con la blockchain escogida. A pesar de no estar descentralizada porque únicamente se crea un nodo en la cadena, es funcional y suficiente para implementar un prototipo que permita realizar esta función de registro y compartición de datos. Un posible despliegue, pero más complejo, se puede hacer con alternativas como Quorum [49] que permiten realizar el despliegue de un nodo privado de Ethereum.

Smart Contract

El contrato inteligente es el encargado de almacenar los datos de la identidad digital del usuario, así como otros tipos de información relacionados para esta prueba de concepto. En el contrato se definen tres estructuras de datos principales (ver figura 4.5): *d*, los datos personales del usuario; *webs* donde se almacena información sobre los SERUs con consentimiento para consultar la información; y *dHistory* la estructura en la que se almacenan las actualizaciones de datos personales realizadas por el usuario.

```

struct Data {
    address wallet; string email;
    string dni; string name;
    string surname; string gender;
    uint birthday; string addr;
    string city; string postalCode;
    string country; string phoneNumber;
    string creditCard;
}

Data private d;

struct WebsInfo{
    string name;
    bool permission;
    uint256[] access;
    string[] dataAccessed;
    uint256 acceptanceDate;
}

WebsInfo[] internal webs;

struct DataHistory {
    Data data;
    uint256 lastUpdate;
}

DataHistory[] internal dHistory;
```

Figura 4.5: Estructuras de almacenamiento de la identidad digital del usuario en el contrato.

El contrato es desplegado en la blockchain del usuario cuando este registra por primera vez sus datos personales a través del SGIDU (ver figura 4.6). Durante este proceso, se compila el código del contrato, se genera la dirección que identifica el contrato y la Interfaz Binaria de Aplicación (ABI), que contiene la información para realizar las llamadas a las funciones programadas en el contrato. Finalmente se crea un nuevo bloque con la transacción de la creación del contrato y se realiza la llamada a las funciones para almacenar la información del usuario.

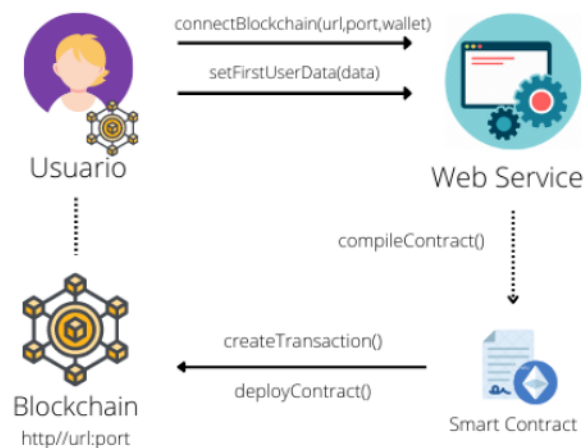


Figura 4.6: Despliegue del *Smart Contract* al través del SGIDU.

Para realizar llamadas a las funciones programadas en el contrato es suficiente con aportar el ABI generado, la dirección del contrato y la *wallet* del nodo que realiza esta llamada. Para aquellas funciones que registren información en el contrato, es necesario indicar también la clave privada del nodo. Es importante aclarar que la clave privada no es almacenada en ningún momento, simplemente que es solicitada al usuario para firmar las transacciones creadas al ejecutar las operaciones de escritura del contrato. En el capítulo donde se muestran los resultados (ver 5) de la prueba de concepto se puede observar el código de las diferentes funciones del contrato utilizadas por el SGIDU.

Provider

El *provider* es el elemento que actúa como conector entre el servicio web del SGIDU y la blockchain de usuario. A través de él, el servicio web le entrega las diferentes acciones que el usuario ha elegido en su interfaz y éste las ejecuta sobre la cadena de bloques y el contrato inteligente. Permite consultar todo tipo de información de la cadena o registrar nuevos elementos como bloques o transacciones. También es el encargado de desplegar los contratos en la cadena de bloques y de ejecutar las llamadas de las diferentes funciones que residen en él. La conexión se realiza mediante un nodo, de manera que recibe parámetros como la URL y el puerto en que se encuentra desplegada la blockchain y devuelve un objeto (ver figura 4.7) a través del cual el servicio web ejecuta las diferentes funciones implementadas en la librería para interactuar con la blockchain.

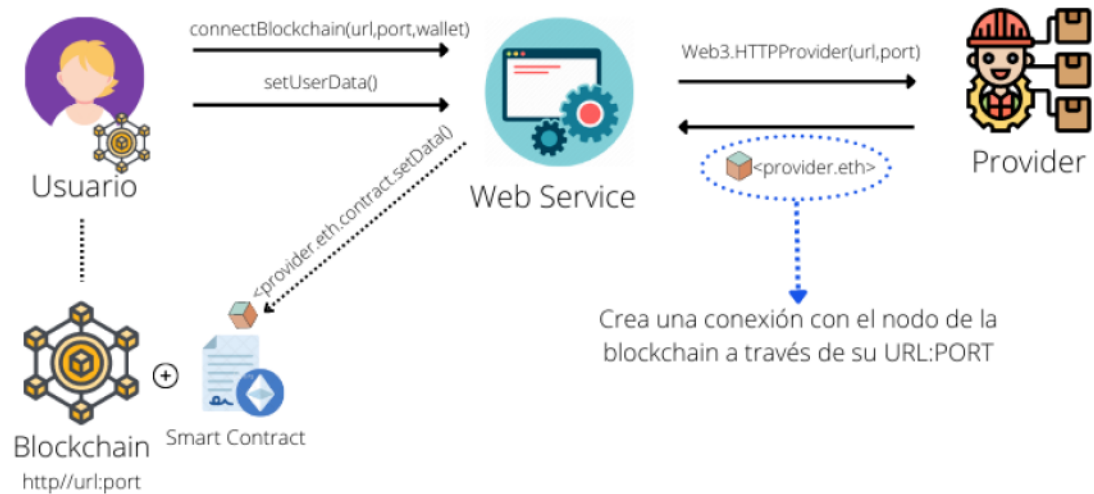


Figura 4.7: Ejemplo de interacción del *provider* para conectar el servicio web con la blockchain y registrar información en el *Smart Contract*. El objeto implementa las funciones necesarias para interactuar con la blockchain.

La selección del *provider* va bastante relacionada con la blockchain que se haya elegido, para interactuar con cada blockchain es necesario utilizar un *provider* determinado. En este caso, el *provider* necesario para interactuar con la cadena de bloques ha sido *Web3.py* [56]. Esta librería implementa, en el lenguaje de programación Python, las funciones necesarias para realizar cualquier interacción con la cadena de bloques de Ethereum.

4.2.2 Sistema Externo de Registro de Usuarios (SERU)

El SERU, como hemos introducido anteriormente, permite registrar clientes o usuarios en su sistema. Para ello, utilizan una base de datos tradicional en la que se almacenan dos tipos de parámetros: los parámetros de autenticación del usuario, usuario y contraseña; y los parámetros de conexión a la blockchain, URL, puerto y *wallet*. El resto de los datos del usuario serán obtenidos a través de consultas a la información almacenada en su blockchain. Para realizar estas consultas, los SERUs deben solicitar al usuario el correspondiente consentimiento previo para le permita acceder a su información personal almacenada en la blockchain. Para aprobar este consentimiento el usuario debe indicar su clave privada para aprobar la transacción y el registro del consentimiento en el *Smart Contract*. La clave privada es solicitada en cada registro, en ningún caso es almacenada en la base de datos del SERU.

El SERU se compone principalmente de cuatro elementos (ver figura 4.8): (1) el servicio web implementado mediante Flask, al igual que la interfaz del usuario; (2) una base de datos en la que se almacena la información necesaria para el registro y autenticación de los usuarios; (3) el *Smart Contract* del usuario para realizar consultas de datos personales; y (4) el *provider* para interactuar con la blockchain del usuario para consultar información.

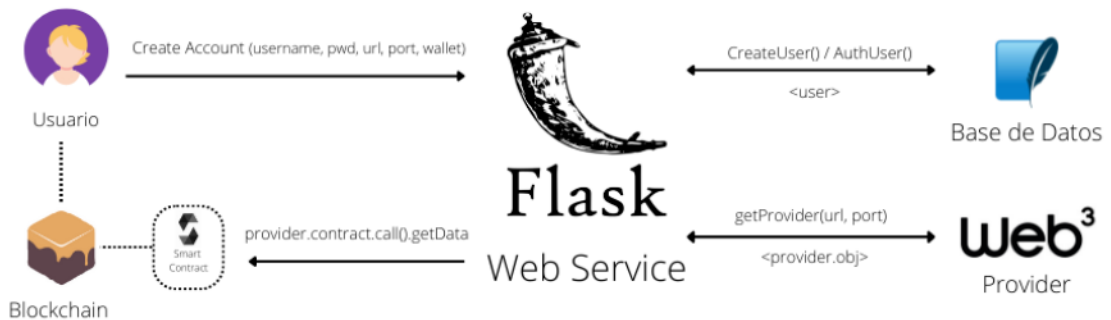


Figura 4.8: Arquitectura y componentes del SERU.

Como hemos comentado en la introducción (ver sección 4.1), en este caso, se implementan dos SERU diferentes para solicitar parámetros distintos de la identidad digital del usuario: “Pc Shop” y “Biblioteca”. Debido a que su estructura y componentes de ambos son idénticos, trataremos el SERU en general para evitar duplicar la información.

Servicio Web

El servicio web implementado en el SERU, al igual que el servicio web del SGIDU, se encarga de proporcionar la estructura web del *Back-End* y *Front-End* del sistema. Proporciona al usuario diferentes interfaces web desde las que podrá registrarse, autenticarse, o visualizar la información del perfil de su nueva cuenta creada. Para ello interactúa con los diferentes componentes del sistema para obtener la información necesaria y presentársela al usuario por pantalla. En la figura 4.9 podemos ver como interactúa con el resto de los componentes para generar la interfaz de perfil del usuario, una vez haya sido registrado en el SERU y aceptado el consentimiento de acceso de sus datos.

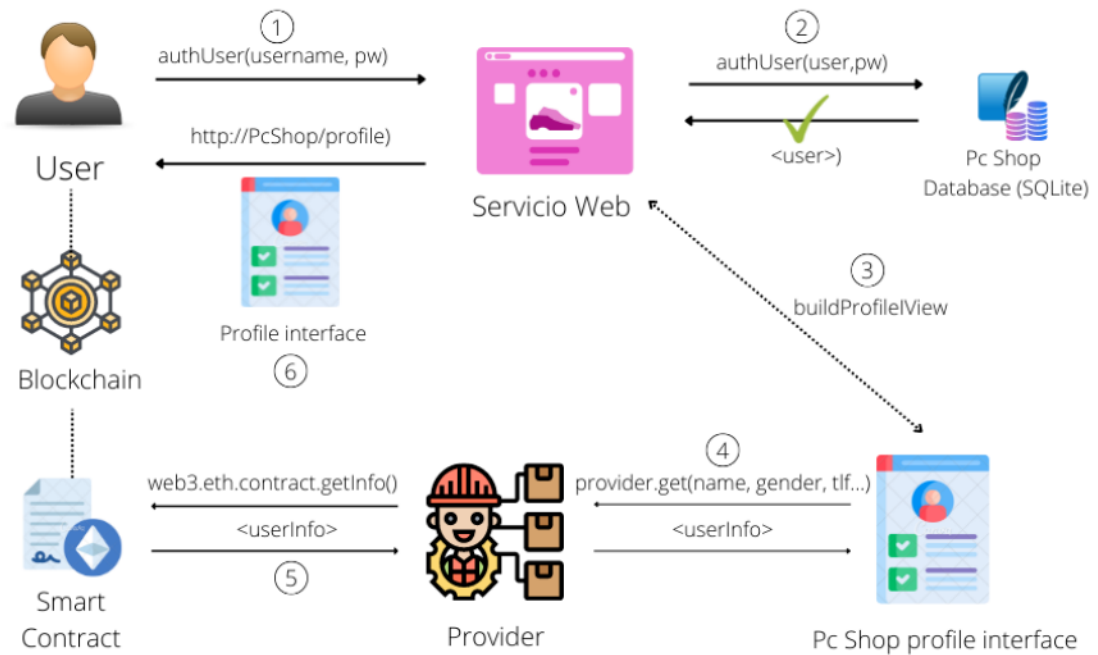


Figura 4.9: Ejemplo de la obtención de datos del usuario para construir la vista de perfil en el SERU “Pc Shop”.

Para la implementación del servicio web también se ha elegido Flask. Los motivos de su elección son los mismos comentados en el caso del SGIDU (ver sección 4.2.1).

Base de datos

La base de datos es el elemento encargado de almacenar las credenciales de los usuarios creados. En ella, como podemos ver en la figura 4.10, se registran los parámetros de autenticación y conexión a la blockchain comentados anteriormente: su nombre de usuario y contraseña, URL, puerto y *wallet*. Tras haber registrado esta información, el servicio web solicita la información de los diferentes usuarios registrados a la base de datos, bien para cotejar las credenciales en el proceso de autenticación o para obtener la información necesaria para consultar la blockchain del usuario. Es importante recordar que en esta base de datos no se registran los datos personales del usuario, sino que se consultan en la blockchain a través de las funciones programadas en el *Smart Contract*. Además, la clave privada que autoriza el registro del consentimiento y los diferentes accesos del SERU no es almacenada.

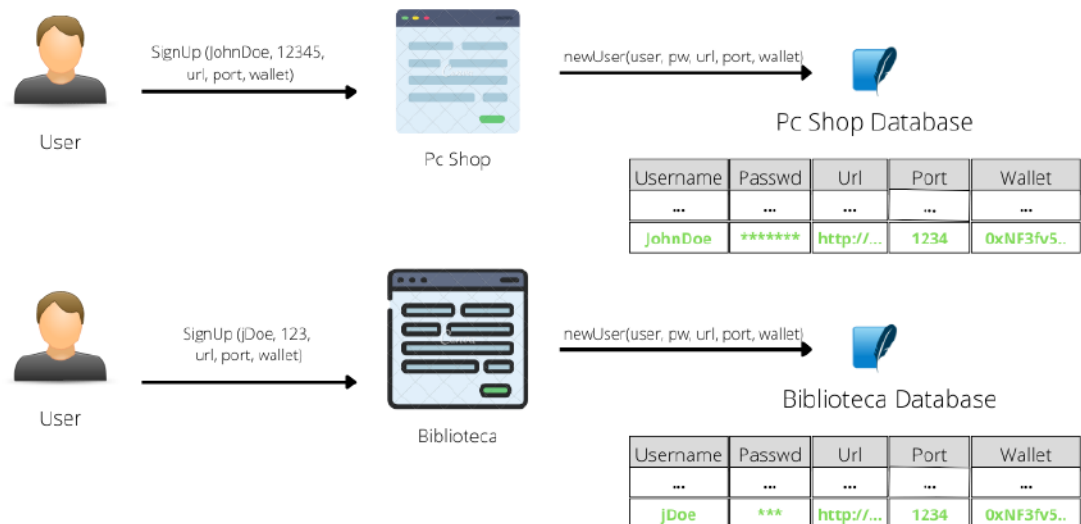


Figura 4.10: Esquema de registro de los parámetros de datos necesarios en las bases de datos de los SERUs “Pc Shop” y “Biblioteca”.

Como herramienta de base de datos se ha utilizado SQLite [57]. Al igual que en el caso del servicio web, se ha tratado de escoger una solución funcional y óptima para los requerimientos necesarios, que permita simplificar los procesos más básicos para centrarse en aquellos más complejos. SQLite se ajusta perfectamente a esto, pues ofrece un servicio de base de datos simple, con las funcionalidades básicas. Abstrae el sistema de base de datos a un fichero local (db.sqlite), auto generado en la compilación del sistema, sobre el que se registra la información sin necesidad de conexión a Internet.

Provider

El *provider* del SERU, al igual que el *provider* del SGIDU, actúa como nexo entre el servicio web del SERU y la blockchain del usuario. En este caso se encarga principalmente de realizar las llamadas a las funciones almacenadas en el contrato inteligente desplegado en la blockchain del usuario. Indicando la *wallet* del usuario, la dirección del contrato y el ABI (Application Binary interface) generado en despliegue, podrá hacer uso de las funciones de consulta de datos del usuario, registro del consentimiento del SERU y de cada acceso a los datos del usuario que realice. Estas dos últimas funciones deben ser aprobadas por el usuario solicitando su clave privada.

En este caso también se utiliza la librería “web3.py” como *provider*, pues los SERUs también están programados en Python y acceden a la misma blockchain que el SGIDU.

Smart Contract

A pesar de que el contrato inteligente del usuario es desplegado por el SGIDU, los SERUs hacen uso de las diferentes funciones programadas en él. Principalmente utilizan tres tipos de funciones: las funciones que permiten obtener los diferentes atributos de información personal del usuario, las que permiten registrar el consentimiento del SERU en el contrato y por último las funciones de registro del acceso a los datos del usuario.

En la figura 4.11 podemos ver el proceso donde el usuario acepta la solicitud de consentimiento que permite a la web para acceder a los diferentes atributos solicitados y como este consentimiento es registrado en el contrato del usuario. Para el caso de las funciones de escritura del contrato el usuario debe indicar su clave privada para crear la nueva transacción en la blockchain. Recordemos que la clave privada es solicitada por pantalla en cada caso, no se almacena en ningún momento.

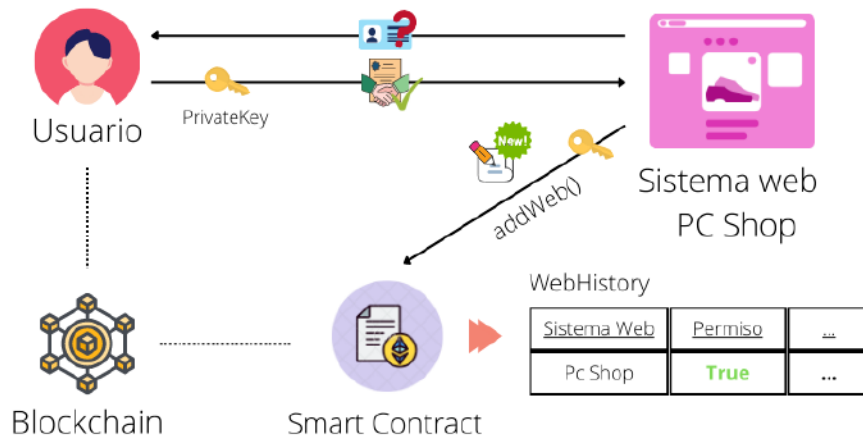


Figura 4.11: Registro de consentimiento de un SERU en el *Smart Contract* del usuario.

Al igual que en el caso del SGIDU, podemos ver las diferentes funciones del contrato ejecutadas en cada caso en el capítulo de Resultados (ver capítulo 5).

4.3 Funcionalidades

Como hemos en el capítulo relativo a la metodología (ver sección 3) las diferentes funcionalidades han sido implementadas mediante nuevas iteraciones secuenciales, añadiendo nuevas funciones y acciones al sistema. Al igual que en el caso de la arquitectura, dividimos las funcionalidades según los sistemas en los que han sido implementadas: SGIDU y SERU.

4.3.1 Sistema de Gestión de Identidad Digital del Usuario (SGIDU)

En el caso del SGIDU, las funcionalidades aplicadas en este caso se corresponden principalmente con la conexión de la blockchain al sistema, la creación, compilación y despliegue del contrato en la blockchain y las funcionalidades para almacenar los datos del usuario en el contrato. Además, se han añadido otras funcionalidades adicionales como la actualización de los datos personales del usuario.

Conexión y registro de datos personales en la blockchain del usuario

Con esta funcionalidad se implementan las funciones necesarias en el *Back-End* del sistema para el que el usuario conecte su blockchain al SGIDU. También se añaden las funciones para desplegar el contrato inteligente en la blockchain del usuario y las funciones propias del contrato para registrar la información del usuario.

Visualización de los datos registrados en la blockchain

En esta interacción se añaden al contrato las funciones necesarias para consultar los datos registrados anteriormente en la blockchain del usuario. Realizando las llamadas a estas funciones, el usuario podrá completar su vista de perfil en el SGIDU, visualizando la información personal que ha registrado. En ella podrá consultar los datos personales que ha registrado previamente, las diferentes webs a las que ha permitido el acceso a sus datos personales y las diferentes actualizaciones que ha realizado de sus datos.

Actualización de los datos personales del usuario y registro del histórico de modificaciones de datos

Algunos datos personales del usuario pueden variar con el tiempo, teniendo que ser modificados. Un ejemplo puede ser el número de teléfono o los datos referidos a su residencia. Para ello se ha añadido una nueva funcionalidad en la que el usuario podrá actualizar los datos que ha registrado previamente. En esta nueva interacción se implementa en el contrato la estructura que permite almacenar el histórico de datos actualizados del usuario. Realizando peticiones a esta estructura, el usuario podrá consultar las diferentes actualizaciones de sus datos almacenadas en el contrato a lo largo del tiempo, así como información relacionada con cada modificación y la fecha y hora en la que se ha realizado.

4.3.2 Sistema Externo de Registro de Usuario (SERU)

En el caso del SERU, se han implementado las funcionalidades necesarias tanto para el registro como la autenticación de usuarios. En las correspondientes iteraciones, se ha completado la prueba de concepto añadiendo las funcionalidades principales de solicitud y registro tanto del consentimiento como del acceso a los datos.

Registro de usuarios

Se implementa en el SERU el proceso y las interfaces necesarias al servicio web para la creación y registro de nuevos usuarios. Para ello se valida la información introducida por el usuario durante su registro en el SERU y se crea una tupla en su base de datos interna almacenando los parámetros de autenticación y los parámetros de conexión de a la blockchain.

Autenticación de usuarios

Adicionalmente al registro de usuarios, se implementa también el proceso de autenticación en el SERU. Para ello se implementa la interfaz de login, así como las diferentes funciones de validación necesarias para que un usuario pueda iniciar sesión en el SERU. El usuario deberá introducir sus credenciales de registro, el SERU las valida y procede con el inicio de sesión del usuario a través de la librería *flask-login*.

Registro de consentimiento del SERU la blockchain del usuario

Tras las funcionalidades de registro y autenticación, se añade la funcionalidad de solicitud y registro del consentimiento de consulta de los datos del usuario. Este proceso se realiza mediante una nueva interfaz de solicitud de consentimiento en los SERUs, en la que el usuario podrá indicar si permitir el acceso a los diferentes atributos de su identidad digital indicados en cada solicitud. También se implementan las funciones necesarias en el contrato para registrar el consentimiento y los diferentes accesos que el SERU realizará posteriormente a los datos del usuario.

Consulta de datos personales en blockchain e interfaz de perfil

Esta funcionalidad permite obtener al usuario tener una vista de perfil en los SERUs en la que puede consultar la información del perfil creado en el SERU correspondiente. Para hacer uso de esta funcionalidad, se implementan las funciones necesarias en el contrato inteligente para que el SERU consulte la información personal a la que tiene permitido el acceso. Cada vez que el usuario acceda a su perfil y el SERU acceda a sus datos, registrará el acceso en su contrato. El usuario deberá confirmar este registro a través de su clave privada.

Modificación de la contraseña del usuario

Finalmente, se ha añadido la funcionalidad que permite modificar la contraseña establecida en la cuenta del usuario creada en el SERU.

Capítulo 5

Resultados

Este capítulo muestra los resultados del trabajo expuesto a lo largo de este documento. En él se presentan los diferentes sistemas implementados en la prueba de concepto expuesta en el capítulo 4. En el apéndice A podemos encontrar el código completo del *Smart Contract* programado al que se hace referencia en las diferentes secciones comentadas.

El desarrollo de esta prueba de concepto ha permitido analizar la aplicación e integración de la tecnología blockchain en el sistema de identidad descentralizada, probar los detalles técnicos y las herramientas disponibles para interactuar con la tecnología blockchain y los *Smart Contracts*. De esta forma, se han cumplido todos los objetivos del proyecto.

5.1 Blockchain para la gestión auto-soberana de la identidad

El primer elemento de la prueba desarrollada es la blockchain. Dependiendo del tipo de blockchain elegida el proceso es diferente, pero como hemos comentado en el diseño (ver sección 4.2), en este caso se ha utilizado la herramienta Ganache. El despliegue de la blockchain con Ganache es muy sencillo. Únicamente es necesario instalar la interfaz de usuario de Ganache UI [58], crear un nuevo *workspace*, indicar la dirección en la que se desplegará y el número de nodos de la red, uno en este caso, y confirmar el despliegue. Al confirmar estos datos, se despliega una blockchain de Ethereum funcional en la dirección indicada.

Una vez desplegada podemos ver el panel general de la blockchain en donde podremos consultar la información de la red desplegada y de la cadena de bloques. Como podemos observar en la figura 5.1, se muestran los datos de URL, puerto y la *wallet* del nodo que se ha generado. Presionando el icono de la llave, podremos consultar la clave privada del nodo generado.

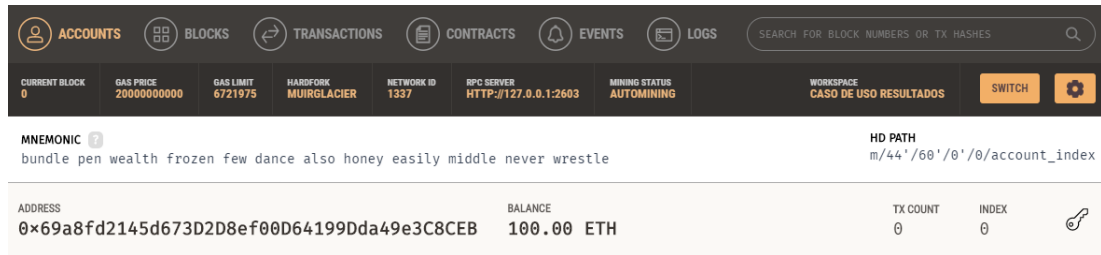


Figura 5.1: Interfaz de conexión de la blockchain del usuario.

5.2 Sistema de Gestión de Identidad Digital del Usuario

El siguiente elemento principal de la prueba de concepto implementada es el Sistema de Gestión de Identidad Digital del Usuario (SGIDU). La interfaz principal de este sistema muestra un sencillo formulario (ver figura 5.2) a través del cual el usuario puede conectar la blockchain que ha desplegado. Para ello, el usuario debe indicar la dirección URL y el puerto en la que ha desplegado su blockchain y la *wallet* del nodo generado.

 A screenshot of a web form titled "Connect your blockchain". The form has three input fields: "Url" with the value "HTTP://127.0.0.1", "Port" with the value "2603", and "Wallet" with the value "0x69a8fd2145d673D2D8ef00D64199Dda49e3C8CEB". Below the input fields is a "Connect" button.

Figura 5.2: Interfaz principal del SGIDU

5.2.1 Registro de los datos del usuario en la blockchain

Además de conectarla, el SGIDU permite al usuario registrar sus datos en la blockchain. Dispone de una interfaz de registro típica en la que proporcionar al usuario un formulario (ver figura 5.3) en el que introducir los datos que sea registrar.

Figura 5.3: Formulario de registro de datos personales del SGIDU.

Para registrar esta información, el SGDIU compila el contrato inteligente (ver apéndice A) y lo despliega sobre la blockchain del usuario. En este contrato se definen diferentes estructuras en las que se almacena la información del usuario. En la siguiente figura 5.4, podemos identificar cada una de ellas: *Data*, la estructura que almacena los atributos actuales del usuario; *webs*, una lista en donde se registra la información relacionada con los SERUs que el usuario ha permitido acceder a sus datos; y *dHistory*, en la que se almacenan las diferentes actualizaciones que el usuario ha hecho de su información personal.

```

struct Data {
    address wallet; string email;
    string dni; string name;
    string surname; string gender;
    uint birthday; string addr;
    string city; string postalCode;
    string country; string phoneNumber;
    string creditCard;
}

Data private d;

struct WebsInfo{
    string name;
    bool permission;
    uint256[] access;
    string[] dataAccessed;
    uint256 acceptanceDate;
}

WebsInfo[] internal webs;

struct DataHistory {
    Data data;
    uint256 lastUpdate;
}

DataHistory[] internal dHistory;

```

Figura 5.4: Estructuras principales del contrato *RegisterData.sol* (ver apéndice A).

Cabe destacar que la clave privada introducida por el usuario no es almacenada por temas de seguridad y privacidad. La clave privada es necesaria en este punto para que el SGIDU pueda desplegar el contrato y crear una nueva transacción para registrarlo en la cadena de bloques. Otro aspecto importante que comentar es que durante el despliegue del contrato se define el parámetro `owner` del contrato. En la figura 5.5 podemos ver cómo se establece este valor al ejecutar el constructor del contrato. En esta instrucción se obtiene la *wallet* del usuario que ha realizado la transacción y lo establece como propietario del contrato.

```
// address who deploys contract will set as contract owner
address owner;

constructor() {
    owner = msg.sender;
}

function saveData () public {
    require(msg.sender == owner);
    dHistory.push(DataHistory(d, block.timestamp));
}
```

Figura 5.5: Registro del propietario del contrato y ejemplo de sentencia `require`. Cuando se crea el contrato se llama a la función `constructor()` del contrato y se define el valor `owner` con la *wallet* del usuario que ha ejecutado la transacción

Esta característica es muy importante, pues es una manera de definir cuáles funcionalidades pueden ser ejecutadas por el creador del contrato y cuáles por cualquier nodo de la red. En las diferentes funciones de escritura definidas en el contrato se comprobará, a través de la sentencia `require`, que la *wallet* del usuario que realiza la llamada a la función en cuestión coincida con el atributo de `owner` registrado. En este caso solo existe un nodo, pero en un entorno real, este sería el método de establecer los roles de los diferentes nodos de la cadena y las funcionalidades que podría llevar a cabo cada uno.

Tras compilar y desplegar el contrato, el SGIDU define los valores que el usuario ha introducido previamente en el formulario en el contrato. Por motivos de número de parámetros, la función debe dividirse en dos funciones más pequeñas, pues el compilador de *Solidity* no permite indicar tantos parámetros en una sola función. Estas funciones son: `setData1()` y `setData2()` (ver figura 5.6).

```
function setData1(address _wallet, string memory _email, string memory _dni, string memory _name,
    string memory _surname, string memory _gender, uint _birthday, string memory _addr) public {
    require(msg.sender == owner);
    d.wallet = _wallet; d.email = _email; d.dni = _dni; d.name = _name;
    d.surname = _surname; d.gender = _gender; d.birthday = _birthday; d.addr = _addr; }

function setData2 (string memory _city, string memory _postalCode, string memory _country,
    string memory _phoneNumber, string memory _creditCard) public {
    require(msg.sender == owner);
    d.city = _city; d.postalCode = _postalCode; d.country = _country;
    d.phoneNumber = _phoneNumber; d.creditCard = _creditCard;}
```

Figura 5.6: Funciones de registro de información del usuario en el contrato.

5.2.2 Visualización de los datos registrados en la vista de perfil

El SGIDU consta de una interfaz de perfil (ver figura 5.7) en la que permite al usuario visualizar la información almacenada en el contrato desplegado en su blockchain: los datos principales de su identidad digital que ha registrado previamente, los datos de los diferentes SERUs con consentimiento de acceso a la blockchain y el historial de las diferentes actualizaciones que el usuario ha realizado.

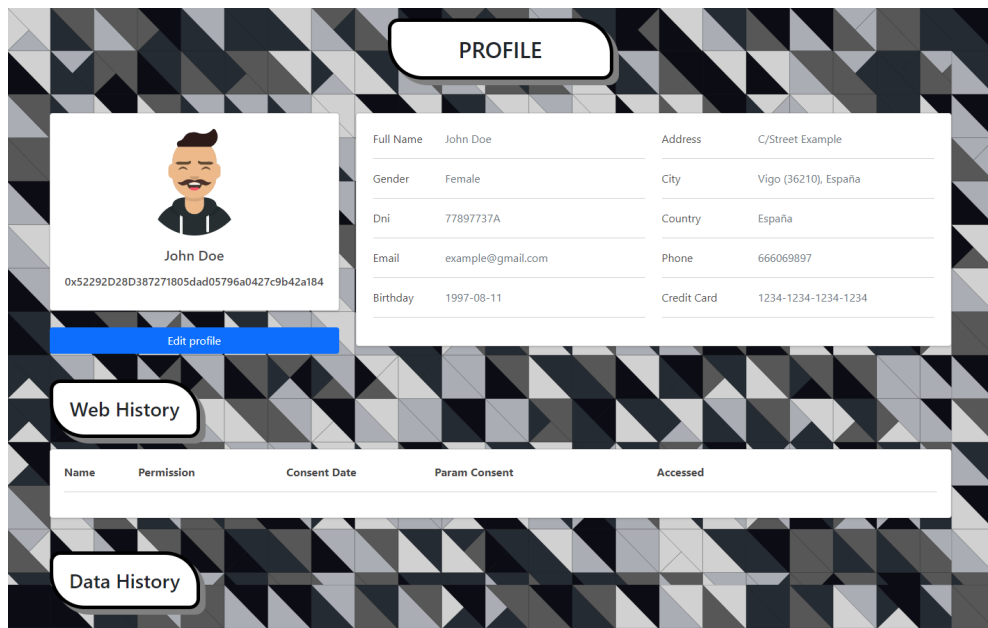


Figura 5.7: Interfaz de perfil del SGIDU.

Para obtener los datos registrados en el contrato, el SGIDU realiza una llamada a la función `getData()` (ver figura 5.8) del contrato mediante la *wallet* del usuario. Esta función únicamente puede ser ejecutada por el dueño del contrato (mediante el código `require` que vemos en la figura 5.8) y le permite obtener la estructura completa con todos los atributos de su identidad digital almacenados en el *Smart Contract*.

```
// Owner function to get his personal data
function getData() public view returns (Data memory){
    require(msg.sender == owner);
    return d; //retrieve last user data registered
}
```

Figura 5.8: Función exclusiva del propietario del contrato para obtener la información personal registrada en él.

5.3 Sistema Externo de Registro de Usuarios

El otro elemento principal de la prueba de concepto es el Sistema Externo de Registro de Usuarios (SGIDU). Como hemos comentado anteriormente, para este caso de uso se han implementado dos SERUs: “Pc Shop” y “Biblioteca”. Un aspecto a tener en cuenta es que, a pesar de ser idénticos en cuanto a estructura, componentes y arquitectura, cada SERU accede a diferentes parámetros de la identidad digital del usuario registrada en la blockchain.

Ambos SERU permiten al usuario crear una nueva cuenta en cada uno de ellos. Es importante comentar que, para poder crear esta nueva cuenta, el usuario que realice el registro debe haber registrado su información personal en la blockchain previamente mediante el SGIDU. En el caso de que cumpla esta condición, el SERU mostrará al usuario una interfaz de registro en la que proporciona un formulario (ver figuras 5.9 y 5.10) para introducir los parámetros de autenticación (usuario y contraseña) y conectividad (URL, puerto y *wallet*) con su blockchain.

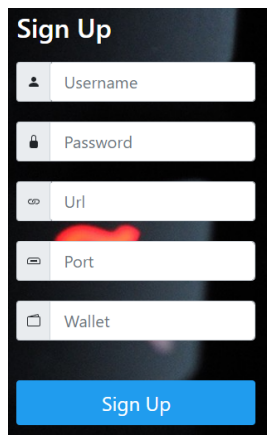
El formulario de registro de usuarios del SERU "Pc Shop" se muestra sobre un fondo oscuro. Tiene el título "Sign Up" en la parte superior. A continuación, hay cinco campos de entrada con iconos a la izquierda: "Username" (persona), "Password" (candado), "Url" (cadena infinita), "Port" (puerto) y "Wallet" (monedero). Al final del formulario hay un botón azul con el texto "Sign Up".

Figura 5.9: Formulario de registro de usuarios del SERU “Pc Shop”.

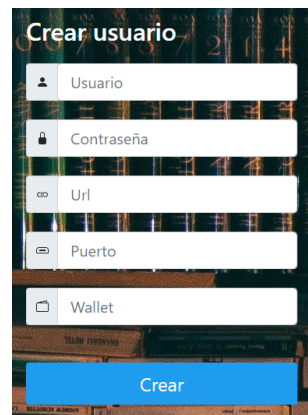
El formulario de registro de usuarios del SERU "Biblioteca" se muestra sobre un fondo con una imagen de estanterías de libros. Tiene el título "Crear usuario" en la parte superior. A continuación, hay cinco campos de entrada con iconos a la izquierda: "Usuario" (persona), "Contraseña" (candado), "Url" (cadena infinita), "Puerto" (puerto) y "Wallet" (monedero). Al final del formulario hay un botón azul con el texto "Crear".

Figura 5.10: Formulario de registro de usuarios del SERU “Biblioteca”.

Para el registro del usuario en las bases de datos de cada SERU, primero se realizan una serie de comprobaciones previas antes de crear el usuario. El SERU valida que no exista el nombre de usuario introducido, la *wallet* o los parámetros de conexión de la blockchain en su base de datos. Adicionalmente, comprueba que el usuario ha registrado anteriormente su información personal en su blockchain. Una vez validados estos parámetros, ejecuta la siguiente sección de código (ver figura 5.11) que permite crear el nuevo usuario en su base de datos correspondiente.

```

# create a new user with the form data. Hash the password so the plaintext version isn't saved.
try:
    #Creates new user object
    new_user = User(username=username, password=generate_password_hash(password, method='sha256'), url=url, port=port, wallet=wallet)
    # add the new user to the database
    db.session.add(new_user)

    #Commit the changes
    db.session.commit()

except Exception as e:
    error = "An unexpected error occurred while creating a new user"
    return render_template("error.html", error=error, details=e)

```

Figura 5.11: Código de los SERUs para la creación de usuarios.

5.3.1 Autenticación del usuario e inicio de sesión en los SERUs

Además del registro, el SERU permite al usuario autenticarse y acceder a su vista de perfil. Para ello presenta una interfaz básica de inicio de sesión en la que muestra un formulario (ver figuras 5.12 y 5.13) en el que el usuario debe introducir sus credenciales para acceder a su cuenta.

here'."/>

Figura 5.12: Formulario de login del SERU “Pc Shop”.

aquí'."/>

Figura 5.13: Formulario de registro de usuarios del SERU “Biblioteca”.

El SERU obtiene la información introducida en el formulario y la valida, comprobando si el usuario existe, si la contraseña es válida y si es posible establecer conexión con su blockchain. A continuación, autentica al usuario y comprueba si dispone de consentimiento para consultar los datos de su blockchain comprobándolo en el *Smart Contract*. Dependiendo de este valor, redirige al usuario a dos interfaces diferentes, para presentar la solicitud de consentimiento (ver sección 5.4) o, en caso de que ya disponga de consentimiento, registrar el acceso a sus datos (ver sección 5.4.1).

5.4 Autorización para la consulta de datos

Esta es una de las funcionalidades más interesantes de la prueba de concepto. Permite implementar un sistema de solicitudes de consentimiento para que el usuario permita a cada SERU acceder a los atributos de los datos del usuario solicitados. De esta manera, el usuario puede tener un control total sobre los diferentes SERUs que acceden a su información, así como los parámetros a los que accede cada uno o el instante de tiempo en el que han consultado esa información.

Si es la primera vez que el usuario accede a su cuenta en el SERU, éste muestra una interfaz en la que se indican los diferentes atributos del usuario a los que solicita acceder. Como podemos ver en cada caso (ver figuras 5.14 y 5.15) cada SERU solicita el acceso a diferentes atributos, para posteriormente realizar las correspondientes llamadas a las funciones de consulta de datos del contrato. Estos atributos se definen en la implementación interna de cada SERU en cada caso.

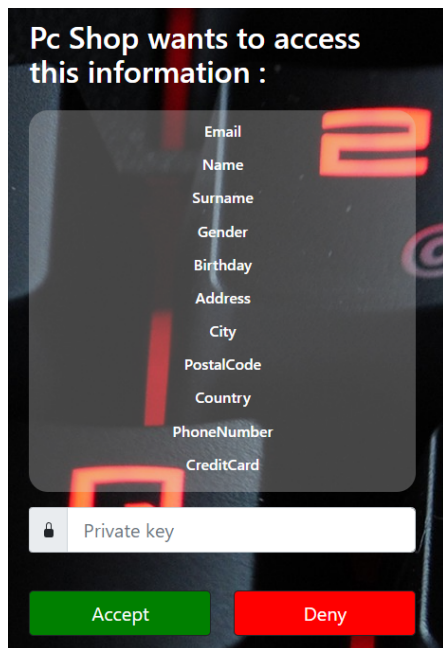


Figura 5.14: Interfaz de solicitud de consentimiento de lectura de los datos personales del usuario en el SERU “Pc Shop”.

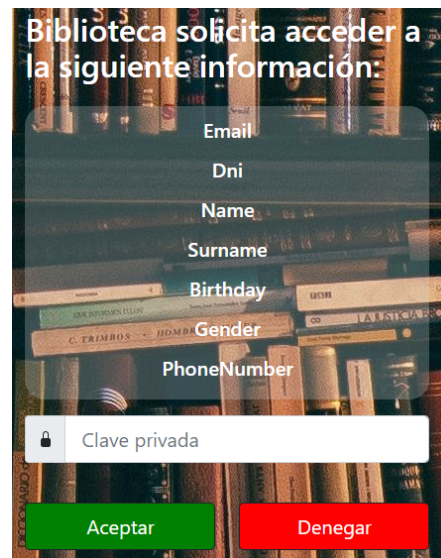


Figura 5.15: Interfaz de solicitud de consentimiento de lectura de los datos personales del usuario en el SERU “Biblioteca”.

En caso de que el usuario acepte el consentimiento solicitado, debe indicar su clave privada al SERU para registrar el consentimiento en su blockchain y añadir la información correspondiente al contrato. Recordemos que el SERU no almacena la clave privada del usuario, sino que la solicita en cada situación en la que el usuario deba registrar tanto el consentimiento como el acceso del SERU a sus datos. Una vez que el usuario indique su clave privada y acepte el consentimiento, el SERU realiza la llamada a la función `addWeb()` del contrato (ver figura 5.16) que registra en la lista `webs` del *Smart Contract* los datos del consentimiento solicitados. Durante este proceso también se registran los atributos a los que ha permitido acceder en cada solicitud y la fecha y hora del consentimiento. En caso de que el usuario deniegue la solicitud, el SERU cierra la sesión del usuario y le redirige a la interfaz principal de login.

```
function addWeb(string memory _Webname, string[] memory _dataAccessed) public {
    require (!checkWebRegistered(_Webname));
    uint[] memory access;
    if (checkEmptyWebs()){
        // Add genesis webInfo without info (index 0)
        string[] memory dataAux;
        webs.push(WebInfo ("None", false, access, dataAux, 0));
    }
    stringToWebIndex[_Webname] = webs.length;
    webs.push(WebInfo (_Webname, true, access, _dataAccessed, block.timestamp));
}
```

Figura 5.16: Función de registro de consentimiento de un SERU del contrato inteligente.

5.4.1 Interfaz de perfil en los SERUs y registro del acceso

Si el usuario ya ha permitido anteriormente el consentimiento de acceso a sus datos al SERU, será redirigido a su vista de perfil correspondiente. En esta interfaz, el SERU permite visualizar la información de los diferentes datos a los que el usuario le ha permitido acceder. En las siguientes figuras 5.17 y 5.18 podemos ver un ejemplo de los perfiles del usuario en los SERUs “Pc Shop” y “Biblioteca” respectivamente.

Para obtener esta información, en cada SERU se realizan las llamadas al contrato desplegado en la blockchain del usuario que permiten obtener la información solicitada. Cada una de estas funciones comprueban en la estructura de registro de webs del usuario que el SERU correspondiente dispone de los permisos necesarios para obtener cada uno de los atributos del usuario que ha solicitado. Al mismo tiempo, cada vez que el SERU consulta los datos del usuario, se registra dicho acceso a través de la función `addAccess()` programada en el contrato (ver figura 5.19). Esta función se encarga de registrar la fecha y hora en la que el SERU ha accedido a los datos del usuario.

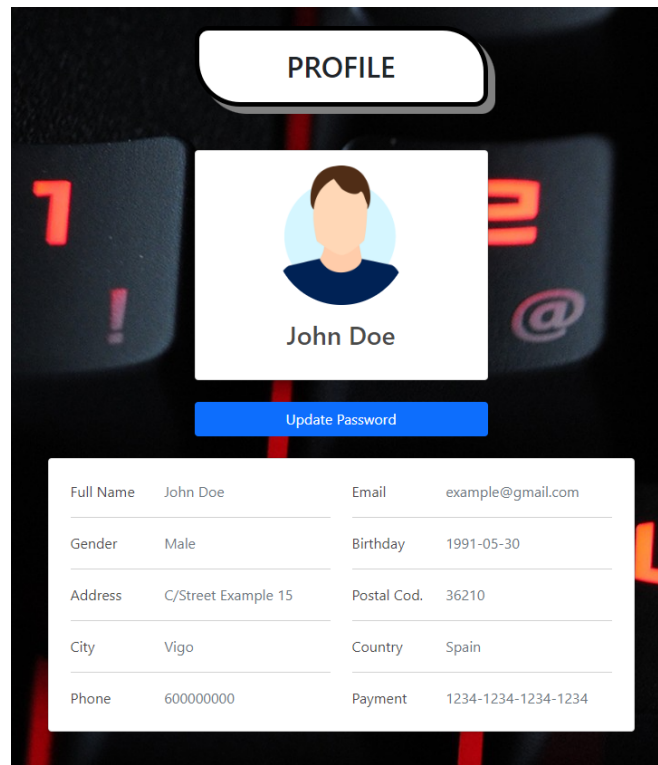


Figura 5.17: Interfaz de perfil del SERU “Pc Shop”.

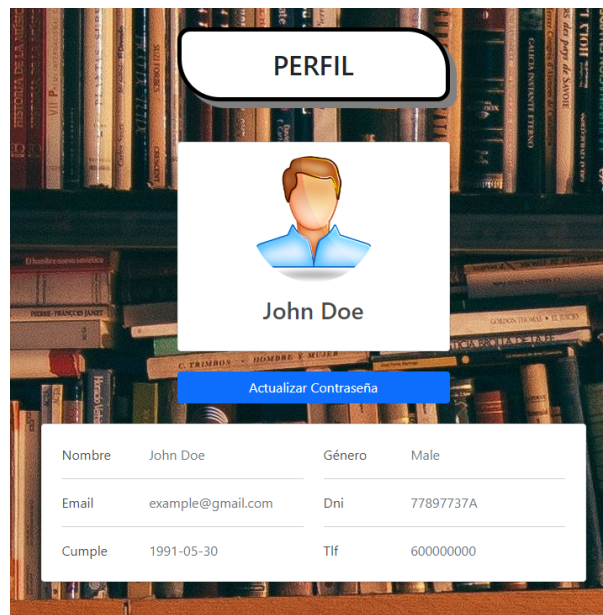


Figura 5.18: Interfaz de perfil del SERU “Biblioteca”.

```
function addAccess (string memory _Webname) public returns (bool){  
    if (checkWebRegistered(_Webname)){  
        if (checkWebConsent(_Webname)){  
            webs[getWebIndex(_Webname)].access.push(block.timestamp);  
            return true;  
        }  
    }  
    return false;  
}
```

Figura 5.19: Función de registro de acceso del SERU de los datos del usuario permitidos.

Como hemos comentado anteriormente, es necesario que el usuario indique su clave privada al SERU para realizar funciones que creen nuevas transacciones en la blockchain y registren información en el contrato. Para este caso existen dos situaciones diferentes. Si el usuario es la primera vez que accede a su cuenta, la clave privada se almacena en una variable temporal cuando acepte el consentimiento anteriormente comentado. De esta manera no será necesario que deba introducir este valor cada vez que acceda a su vista de perfil. Por otro lado, en el caso de que el usuario ya haya registrado el consentimiento del SERU previamente, cuando vuelve a iniciar sesión en el SERU, éste le solicita dicha clave a través del formulario que podemos ver en la figura 5.20 para registrar el acceso a sus datos. Al igual que en el caso anterior, el SERU almacena temporalmente hasta que el usuario cierre sesión.

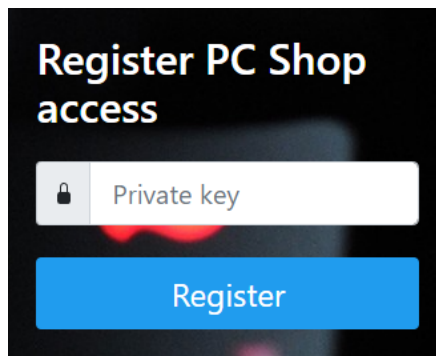


Figura 5.20: Interfaz de solicitud de la clave privada para registrar el acceso del SERU.

Adicionalmente, en cada SERU se implementa la funcionalidad que permite modificar la contraseña establecida en el proceso de registro. Para ello el usuario accede a la interfaz de modificación de contraseña a través del botón ubicado en su perfil. En la figura 5.21 se muestra un ejemplo del formulario que se muestra al usuario para actualizar la contraseña de su cuenta en cada SERU. Una vez modificada, podrá comprobar que se ha aplicado dicho cambio autenticándose de nuevo con sus nuevas credenciales.

Change Password

Current password

New password

Repeat new password

Change password

Figura 5.21: Formulario de actualización de contraseña del SERU “Pc Shop”.

5.4.2 Visualización del registro de consentimientos aprobados en el SGIDU

La interfaz de perfil del SGIDU (ver figura 5.22), permite visualizar la información que ha sido registrada en cada caso en el panel de “Web history”. En este panel se muestran los diferentes atributos a los que cada SERU tiene permitido el acceso, así como los diferentes instantes de tiempo en los que ha accedido a la información.

PROFILE

John Doe

0x52292D28D387271805dad05796a0427c9b42a184

Edit profile

Full Name	John Doe	Address	C/Street Example
Gender	Female	City	Vigo (36210), España
Dni	77897737A	Country	España
Email	example@gmail.com	Phone	666069897
Birthday	1997-08-11	Credit Card	1234-1234-1234-1234

Web History

Name	Permission	Consent Date	Param Consent	Accessed
Pc Shop	True	2022-09-05 10:40:05	Email, Name, Surname, Gender, Birthday, Address, City, PostalCode, Country, PhoneNumber, CreditCard,	2022-09-05 10:40:07
Biblioteca	True	2022-09-05 10:48:15	Email, Dni, Name, Surname, Birthday, Gender, PhoneNumber,	2022-09-05 10:48:16

Data History

Figura 5.22: Interfaz de perfil del SGIDU.

Para mostrar esta información, el SGIDU realiza la llamada a la función del contrato `getHistory()` (ver figura 5.23), que devuelve la información registrada en la lista de webs del contrato.

```
function getHistory() public view returns (DataHistory[] memory) {  
    require(msg.sender == owner);  
    return dHistory;  
}
```

Figura 5.23: Función de obtención del histórico de registro de SERU con consentimiento.

5.5 Actualización de los datos personales del usuario

Una vez registrados los datos, por el motivo que fuera, el usuario puede necesitar actualizar la información que ha registrado previamente. Para ello el SGIDU dispone de una interfaz en la se muestra un simple formulario (ver figura 5.24) en el que el usuario debe introducir los nuevos datos con los que quiere actualizar sus datos personales.

The screenshot shows a mobile application interface with a dark, geometric patterned background. At the top, there is a white button labeled 'UPDATE DATA'. Below it is a white rounded rectangle containing the form. The form has a title 'UPDATE PERSONAL INFO' and several input fields: 'Email' (highlighted with a blue border), 'Dni', 'Name', 'Surname', 'Gender' (with radio buttons for Male, Female, and Other), 'Birthday' (with a date picker icon), 'Address', 'Postal Code', 'City', 'Country', 'PhoneNumber', 'Credit Card', 'Wallet', and 'PrivateKey*'. At the bottom of the form, there is a small text note: '*Your private key will not be stored, its necessary to create the contract' and a large black button labeled 'UPDATE'.

Figura 5.24: Formulario de actualización de datos del usuario en el SGIDU.

Una vez el usuario haya los nuevos datos actualizados y aceptado el formulario, el SGIDU recibe esta información y la almacena en el contrato del usuario. Posteriormente, en su interfaz de perfil del SGIDU podrá ser aplicada esta actualización (ver figura 5.25). En este caso, podemos comprobar que algunos valores como la ciudad, número de teléfono o su dirección se han actualizado. Además, los datos anteriores a la actualización son mostrados en el panel de “Data History” junto con la fecha en la que se ha realizado la actualización.

PROFILE

John Doe
0x69a8fd2145d673D2D8ef00D64199Dda49e3C8CEB

Edit profile

Full Name	John Doe	Address	C/ San Sebastián 1
Gender	Male	City	A Coruña (15007)
Dni	77897737A	Country	Spain
Email	email2@example.com	Phone	612345678
Birthday	1991-05-30	Credit Card	1111-1111-1111-1111

Web History

Name	Permission	Consent Date	Param Consent	Accessed
Pc Shop	True	2022-09-05 16:42:27	Email, Name, Surname, Gender, Birthday, Address, City, PostalCode, Country, PhoneNumber, CreditCard,	(2022-09-05 16:42:29)
Biblioteca	True	2022-09-05 16:49:56	Email, Dni, Name, Surname, Birthday, Gender, PhoneNumber,	(2022-09-05 16:49:57)

Data History

Full Name	John Doe	Gender	Male	Dni	77897737A	Email	example@gmail.com
Birthday	1991-05-30	Address	C/Street Example 15	City	Vigo	Country	Spain
Phone	600000000	CreditCard	1234-1234-1234-1234	Update	2022-09-05 16:54:15		

Figura 5.25: Interfaz de perfil del SGIDU completa.

Vamos a analizar lo que ocurre durante en este proceso. Cuando el usuario acepta los cambios en el formulario e indica su clave privada, el SGIDU realiza dos llamadas a tres funciones del contrato. Primeramente la función `saveData()` (ver figura 5.26), la cual añade la información registrada antes del cambio a la lista de modificaciones “*dHistory*” del contrato. Una vez almacenada esta información, llama a las funciones `setData1()` y `setData2()`, vistas anteriormente (ver figura 5.6), para almacenar los datos actualizados.

```
function saveData () public {
    require(msg.sender == owner);
    dHistory.push(DataHistory(d, block.timestamp));
}
```

Figura 5.26: Función de registro de los anteriores datos personales del usuario en la blockchain.

Esta funcionalidad es sin duda una de las más interesantes. Debido a que la actualización de los datos del usuario se realiza en la blockchain, todos aquellos SERUs en los que el usuario se haya registrado previamente actualizarán esta información automáticamente. Si tras aplicar la actualización el usuario accede de nuevo a la interfaz de perfil del usuario en los SERUs (ver figuras 5.27 y 5.28), podrá ver que se han actualizado sus datos en los perfiles de cada SERU.

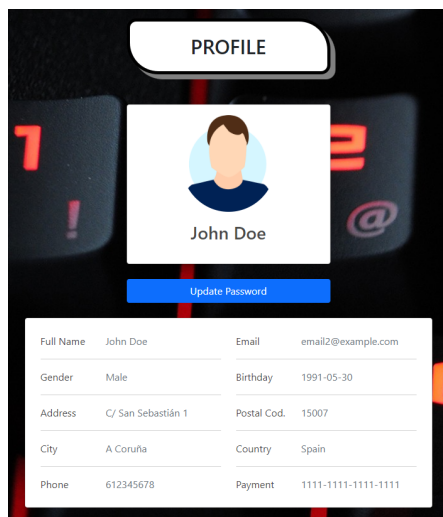


Figura 5.27: Interfaz de perfil del SE-RU “Pc Shop” actualizada.

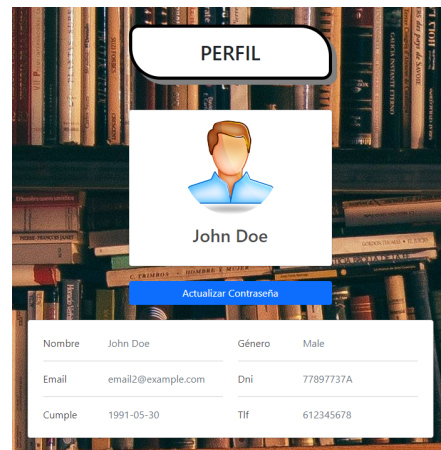


Figura 5.28: Interfaz de perfil del SE-RU “Biblioteca” actualizada.

Este es uno de los puntos más interesantes de la prueba de concepto pues permite gestionar la información de los usuarios en los todos los SERUs desde un único punto. En un sistema de registro de usuarios tradicional, planteando el caso de que el usuario necesitase modificar algún atributo, el usuario debería aplicar la modificación manualmente en cada sistema web que almacenase el parámetro modificado.

El código del de la prueba de concepto implementada está disponible en mi repositorio personal de github [59]. Además del código, se detallan las instrucciones para realizar una prueba funcional de los sistemas implementados.

Conclusiones y trabajo futuro

Tras las cuestiones tratadas a lo largo de este trabajo, la investigación realizada y el aprendizaje obtenido a lo largo de la elaboración de este proyecto, se concluye con una valoración final. Adicionalmente, se añade un apartado de líneas de futuro, referido a las diferentes ideas y propuestas planteadas para la prueba concepto desarrollada en el futuro.

6.1 Conclusión

Como conclusión, tras el trabajo realizado, podemos afirmar que la identidad digital descentralizada es una solución muy interesante en cuanto al ámbito de la identidad digital en Internet. Existen una gran cantidad de inconvenientes en la identidad digital actual como hemos analizado a lo largo de esta memoria y la identidad digital descentralizada es sin duda una de las soluciones más a tener en cuenta para tratar solventar todas estas desventajas. Sin duda el aspecto que más destacamos de este nuevo sistema de identidad digital es la capacidad para permitir devolver la auditoría al usuario de sus datos personales y gestionarlos de manera auto soberana en Internet. La identidad de una persona es un valor propio e intransferible, la identidad digital debe seguir los mismos pasos.

Tanto la tecnología blockchain como el concepto de Internet descentralizado es una de las grandes novedades que poco a poco serán adoptadas y normalizadas en el Internet del futuro. La descentralización de la información de los usuarios en Internet permitirá construir una red más segura, privada y centrada en el usuario.

Con respecto a la implementación de la prueba de concepto, creemos que ha sido de gran utilidad para tener una primera toma de contacto con el mundo de la tecnología blockchain y los *Smart Contracts*. Su desarrollo nos ha permitido aprender las bases de esta nueva tecnología sobre la que se construirán cada vez más aplicaciones en el futuro.

6.2 Líneas de futuro del trabajo

De cara al futuro de este trabajo, hemos establecido una serie de mejoras e ideas para aplicar en la prueba de concepto implementada. Algunas de las más interesantes son:

- **Implementación de la funcionalidad para revocar el consentimiento de acceso de un sitio web:** Con esta funcionalidad el usuario podría eliminar el consentimiento de un servicio web registrado para impedir que acceda a su información personal.
- **Implementación del sistema de registro del consentimiento en el SGIDU:** En el sistema implementado es el SERU quien, con la confirmación de la clave privada indicada por el usuario, ejecutaba la función para añadir el consentimiento aceptado por el usuario en el contrato. Una posible mejora de la prueba de concepto es crear un sistema recepción de solicitudes en el SGIDU en el que el usuario podrá aprobar y registrar el consentimiento de cada SERU desde su SGIDU.
- **Actualización e integración de una blockchain privada descentralizada para el usuario:** Esta mejora permitiría establecer diferentes roles y permisos en los usuarios para ejecutar diferentes acciones o funciones en la blockchain. También añadiría la capacidad de añadir nuevas *wallets* para cada uno de los servicios registrados. Con ello aumentaría la seguridad y privacidad del sistema implementado.
- **Mejorar el *Front-End* y el aspecto visual:** El *Front-End* no ha sido una prioridad en la prueba de concepto debido a que el foco se ha puesto en comprender como funciona la tecnología blockchain y los *Smart Contracts*. Uno de los aspectos a tratar en el futuro será mejorar el aspecto visual de ambos sistemas.
- **Análisis de las ventajas y desventajas de la prueba de concepto en un caso real de producción:** Uno de los aspectos más a tener en cuenta en el futuro es analizar la arquitectura, implementación y funcionalidades de la prueba de concepto propuesta escalada a un entorno de producción real. En este aspecto también es bastante interesante identificar las ventajas e inconvenientes de la arquitectura para tratar de buscar posibles mejoras o presentar propuestas a las diferentes desventajas identificadas.

Apéndices

Material adicional

A.1 Código del contrato inteligente “RegisterData.sol”

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity 0.8.15;
3
4 contract RegisterData {
5
6     struct Data {
7         address wallet;
8         string email;
9         string dni;
10        string name;
11        string surname;
12        string gender;
13        uint birthday;
14        string addr;
15        string city;
16        string postalCode;
17        string country;
18        string phoneNumber;
19        string creditCard;
20    }
21
22    struct WebsInfo{
23        string name;
24        bool permission;
25        uint256[] access;
26        string[] dataAccessed;
27        uint256 acceptionDate;
28    }
29
30 }
```

```
31 struct DataHistory {
32     Data data;
33     uint256 lastUpdate;
34 }
35
36
37 // address who deploys contract will set as owner
38 address owner;
39
40 constructor() {
41     owner = msg.sender;
42 }
43
44 DataHistory[] internal dHistory;
45 WebsInfo[] internal webs;
46 Data private d;
47
48 //First id is 1, 0 value is default in mapping references
49 mapping (string => uint) stringToWebIndex;
50
51 function setData1(address _wallet, string memory _email, string
memory _dni, string memory _name, string memory _surname,
string memory _gender, uint _birthday, string memory _addr)
public {
52     require(msg.sender == owner);
53     d.wallet = _wallet; d.email = _email; d.dni = _dni; d.name
= _name; d.surname = _surname; d.gender = _gender; d.birthday =
_birthday; d.addr = _addr;
54 }
55
56 function setData2 (string memory _city, string memory
_postalCode, string memory _country, string memory _phoneNumber,
string memory _creditCard) public {
57     require(msg.sender == owner);
58     d.city = _city; d.postalCode = _postalCode; d.country =
_country; d.phoneNumber = _phoneNumber; d.creditCard =
_creditCard;
59 }
60
61 // Owner get personal data function
62 function getData() public view returns (Data memory){
63     require(msg.sender == owner);
64     return d; //retrieve last user data registered
65 }
66
67 function getWebsInfo() public view returns (WebsInfo[] memory){
```



```
68         require(msg.sender == owner);
69         return webs;
70     }
71
72     function getWallet (string memory _WebName) public view returns
73     (address) {
74         require(checkWebConsent(_WebName));
75         require(checkParameterConsent(_WebName, 'Wallet'));
76         return d.wallet;
77     }
78     function getEmail (string memory _WebName) public view
79     returns(string memory) {
80         require(checkWebConsent(_WebName));
81         require(checkParameterConsent(_WebName, 'Email'));
82         return d.email;
83     }
84
85     function getDni (string memory _WebName) public view returns
86     (string memory) {
87         require(checkWebConsent(_WebName));
88         require(checkParameterConsent(_WebName, 'Dni'));
89         return d.dni;
90     }
91
92     function getName (string memory _WebName) public view returns
93     (string memory) {
94         require(checkWebConsent(_WebName));
95         require(checkParameterConsent(_WebName, 'Name'));
96         return d.name;
97     }
98
99     function getSurname (string memory _WebName) public view
100    returns (string memory) {
101        require(checkWebConsent(_WebName));
102        require(checkParameterConsent(_WebName, 'Surname'));
103        return d.surname;
104    }
105
106    function getGender (string memory _WebName) public view
107    returns (string memory) {
108        require(checkWebConsent(_WebName));
109        require(checkParameterConsent(_WebName, 'Gender'));
110        return d.gender;
111    }
112
113    function getBirthday (string memory _WebName) public view
114    returns (uint) {
```

```
107         require(checkWebConsent(_WebName));
108         require(checkParameterConsent(_WebName, 'Birthday'));
109         return d.birthday;
110     }
111
112     function getAddress (string memory _WebName) public view
113     returns (string memory){
114         require(checkWebConsent(_WebName));
115         require(checkParameterConsent(_WebName, 'Address'));
116         return d.addr;
117     }
118
119     function getCity (string memory _WebName) public view returns
120     (string memory){
121         require(checkWebConsent(_WebName));
122         require(checkParameterConsent(_WebName, 'City'));
123         return d.city;
124     }
125
126     function getPostalCode (string memory _WebName) public view
127     returns (string memory) {
128         require(checkWebConsent(_WebName));
129         require(checkParameterConsent(_WebName, 'PostalCode'));
130         return d.postalCode;
131     }
132
133     function getCountry (string memory _WebName) public view
134     returns(string memory) {
135         require(checkWebConsent(_WebName));
136         require(checkParameterConsent(_WebName, 'Country'));
137         return d.country;
138     }
139
140     function getPhoneNumber (string memory _WebName) public view
141     returns (string memory){
142         require(checkWebConsent(_WebName));
143         require(checkParameterConsent(_WebName, 'PhoneNumber'));
144         return d.phoneNumber;
145     }
146
147     function getCreditCard (string memory _WebName) public view
148     returns (string memory){
149         require(checkWebConsent(_WebName));
150         require(checkParameterConsent(_WebName, 'CreditCard'));
151         return d.creditCard;
152     }
```

```
147
148     function saveData () public {
149         require(msg.sender == owner);
150         dHistory.push(DataHistory(d, block.timestamp));
151     }
152
153     function getHistory() public view returns (DataHistory[]
memory) {
154         require(msg.sender == owner);
155         return dHistory;
156     }
157
158     function addWeb(string memory _Webname, string[] memory
_dataAccessed) public {
159         require (!checkWebRegistered(_Webname));
160         uint[] memory access;
161         if (checkEmptyWebs()){
162             // Add genesis webInfo without info (index 0)
163             string[] memory dataAux;
164             webs.push(WebsInfo ("None", false, access, dataAux, 0));
165         }
166         stringToWebIndex[_Webname] = webs.length;
167         webs.push(WebsInfo (_Webname, true, access, _dataAccessed,
block.timestamp));
168     }
169
170     function addAccess (string memory _Webname) public returns
(bool){
171         if (checkWebRegistered(_Webname)){
172             if (checkWebConsent(_Webname)){
173
174                 webs[getWebIndex(_Webname)].access.push(block.timestamp);
175                 return true;
176             }
177         }
178         return false;
179     }
180
181     function getWebParams(string memory _Webname) public view
returns (string[] memory ){
182         require(msg.sender == owner);
183         return getWebInfo(_Webname).dataAccessed;
184     }
185
186     function getWebInfo(string memory _Webname) public view returns
(WebsInfo memory){
```

```
186         require(msg.sender == owner);
187         uint index = getWebIndex(_Webname);
188         return webs[index];
189     }
190
191     function getWebIndex(string memory _Webname) internal view
192     returns (uint) {
193         return stringToWebIndex[_Webname];
194     }
195
196     function checkEmptyWebs () internal view returns (bool){
197         return webs.length == 0;
198     }
199
200     function checkWebConsent(string memory _Webname) public view
201     returns (bool){
202         if (checkWebRegistered(_Webname)){
203             return getWebInfo(_Webname).permission;
204         }
205         return false;
206     }
207
208     function checkWebRegistered (string memory _Webname) public
209     view returns (bool){
210         return getWebIndex(_Webname) > 0;
211     }
212
213     function checkParameterConsent (string memory _webName, string
214     memory parameter) internal view returns (bool){
215         string[] memory params = getWebParams(_webName);
216         for (uint i=0; i < params.length; i++) {
217             if ((keccak256(abi.encodePacked(params[i]))) ==
218             (keccak256(abi.encodePacked(parameter)))){
219                 return true;
220             }
221         }
222         return false;
223     }
224 }
```


Bibliografía

- [1] O. White, A. Madgavkar, J. Manyika, D. Mahajan, J. Bughin, M. McCarthy, and O. Sperling, “Digital identification: A key to inclusive growth,” *McKinsey Global Institute*, Apr 2019. [En línea]. Disponible en: <https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/digital-identification-a-key-to-inclusive-growth>
- [2] T. Group. (2018) Data breaches compromised 4.5 billion records in first half of 2018*. [En línea]. Disponible en: <https://www.thalesgroup.com/en/markets/digital-identity-and-security/press-release/data-breaches-compromised-4-5-billion-records-in-first-half-of-2018>
- [3] What is self-sovereign identity? [En línea]. Disponible en: <https://sovrin.org/faq/what-is-self-sovereign-identity/>
- [4] Self-sovereign identity (ssi): Autonomous identity management. [En línea]. Disponible en: <https://www.okta.com/identity-101/self-sovereign-identity/>
- [5] N. Roby, D. Yaga, P. Mell, and K. Scarfone, “Blockchain technology overview,” *NIST Interagency/Internal Report (NISTIR), National Institute of Standards and Technology, Gaithersburg*, 2019. [En línea]. Disponible en: <https://doi.org/10.6028/NIST.IR.8202>
- [6] A. Sherman, F. Javani, H. Zhang, and E. Golaszewski, “On the origins and variations of blockchain technologies,” *Sci-Hub*, pp. 72–77, 2019.
- [7] N. Rodriguez, “Historia de la tecnología blockchain: Guía definitiva,” *101Blockchains*, 2018. [En línea]. Disponible en: <https://101blockchains.com/es/historia-de-la-blockchain/>
- [8] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” *Nakamoto Institute*, Oct 2008. [En línea]. Disponible en: <https://nakamotoinstitute.org/bitcoin/>

- [9] (2019) Estado del arte de blockchain en la empresa española. [En línea]. Disponible en: https://ametic.es/sites/default/files//informe_el_estado_del_arte_de_blockchain_en_la_empresa_espanola.pdf
- [10] J. Granger and M. Foster, “The virtual enterprise blueprint,” *IBM Institute for Business Value*, 2021. [En línea]. Disponible en: <https://www.ibm.com/thought-leadership/institute-business-value/report/virtual-enterprise>
- [11] I. y. R. IDC, AMETIC, “Estado del arte del blockchain en la empresa española,” *IT Reseller*, 2020. [En línea]. Disponible en: <https://www.blog-idcspain.com/elestadodelartedelblockchainenlaempresaespanola/>
- [12] J. Frankfield, “What Is a Block (Blockchain Block)?” *Investopedia*, 2022. [En línea]. Disponible en: <https://www.investopedia.com/terms/b/block-bitcoin-block.asp>
- [13] D. Prasad. (2022, may) Una guía detallada sobre los tipos de nodos de cadena de bloques. [En línea]. Disponible en: <https://geekflare.com/es/finance/blockchain-nodes-guide/>
- [14] Unknown. (2022) What is a smart contract? [En línea]. Disponible en: <https://www.coinbase.com/es/learn/crypto-basics/what-is-a-smart-contract>
- [15] J. Frankfield, “Smart Contracts,” *Investopedia*, 2022. [En línea]. Disponible en: <https://www.investopedia.com/terms/s/smart-contracts.asp>
- [16] D. Tapscott and A. Tapscott, *La revolución blockchain*, 1st ed. Deusto, 2017.
- [17] Solidity v0.8.17 is here! [En línea]. Disponible en: <https://soliditylang.org/>
- [18] M. Allende and V. Colina, “¿Pública, federada o privada? Explora los distintos tipos de blockchain,” *Conocimiento Abierto*, 2018. [En línea]. Disponible en: <https://blogs.iadb.org/conocimiento-abierto/es/tipos-de-blockchain/>
- [19] F. Telefónica, *Identidad Digital: el nuevo usuario en el mundo digital*, 1st ed. Ariel S.A, 2013.
- [20] F. Georges, “Représentation de soi et identité numérique. Une approche sémiotique et quantitative de l’emprise culturelle du web 2.0,” *Réseaux*, pp. 165–193, 2009. [En línea]. Disponible en: <https://www.cairn.info/revue-reseaux-2009-2-page-165.htm>
- [21] M. (RUNDLE and P. TREVITHICK, “At a crossroads: ‘Personhood’ and digital identity in the information society,” *OECD*, 2008. [En línea]. Disponible en: <https://www.oecd.org/sti/ieconomy/40204773.doc>
- [22] Paypal. [En línea]. Disponible en: <https://www.paypal.com/es/home>

- [23] (2017, May) The world's most valuable resource is no longer oil, but data. [En línea]. Disponible en: <https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data>
- [24] (2005, Dic) Real Decreto 1553/2005, de 23 de diciembre, por el que se regula la expedición del documento nacional de identidad y sus certificados de firma electrónica. [En línea]. Disponible en: <https://www.boe.es/buscar/act.php?id=BOE-A-2005-21163>
- [25] Unknown, "Orden ETD/465/2021, de 6 de mayo, por la que se regulan los métodos de identificación remota por vídeo para la expedición de certificados electrónicos cualificados." *Boletín oficial del estado*, May 2021. [En línea]. Disponible en: <https://www.boe.es/eli/es/o/2021/05/06/etd465>
- [26] —, "Ley 6/2020, de 11 de noviembre, reguladora de determinados aspectos de los servicios electrónicos de confianza," *Boletín oficial del estado*, Nov 2020. [En línea]. Disponible en: <https://medium.com/caribou-digital/the-difference-between-digital-identity-identification-and-id-41580bbb7563>
- [27] —, "Ley 11/2007, de 22 de junio, de acceso electrónico de los ciudadanos a los Servicios Públicos." *Boletín oficial del estado*, Jun 2007. [En línea]. Disponible en: <https://www.boe.es/buscar/pdf/2007/BOE-A-2007-12352-consolidado.pdf>
- [28] —, "REGLAMENTO (UE) No 910/2014 DEL PARLAMENTO EUROPEO Y DEL CONSEJO de 23 de julio de 2014 relativo a la identificación electrónica y los servicios de confianza para las transacciones electrónicas en el mercado interior y por la que se deroga la Directiva 1999/93/CE," *Diario Oficial de la Unión Europea*, Jul 2014. [En línea]. Disponible en: <https://www.boe.es/doi/2014/257/L00073-00114.pdf>
- [29] —, "Payment services (PSD 2) - Directive (EU) 2015/2366," *Diario Oficial de la Unión Europea*, Nov 2015. [En línea]. Disponible en: <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32015L2366>
- [30] —, "Proposal for a Regulation of the European Parliament and of the Council amending Regulation (EU) No 910/2014 as regards establishing a framework for a European Digital Identity (SEC(2021) 228 final) - (SWD(2021) 124 final) - (SWD(2021) 125 final)," *Diario Oficial de la Unión Europea*, May 2021. [En línea]. Disponible en: <https://digital-strategy.ec.europa.eu/en/library/trusted-and-secure-european-e-id-regulation>
- [31] —, "RECOMENDACIÓN (UE) 2021/946 DE LA COMISIÓN de 3 de junio de 2021 sobre un conjunto de instrumentos común de la Unión para adoptar un enfoque coordinado de cara a un Marco para una Identidad Digital Europea ,"

- Diario Oficial de la Unión Europea*, Jun 2021. [En línea]. Disponible en: <https://www.boe.es/doue/2021/210/L00051-00054.pdf>
- [32] A. P. KUCHKOVSKY, D. D. GARCÍA, and R. F. HERGUETA, “Identidad digital y ‘blockchain’: como llave al cambio del mundo,” *El Pais*, Jun 2017. [En línea]. Disponible en: https://elpais.com/retina/2017/06/05/tendencias/1496646930_763686.html
- [33] M. Fusillo. (2021) Identidad digital y ‘blockchain’: como llave al cambio del mundo. [En línea]. Disponible en: <https://www.selfsovereignidentity.it/modelos-de-identidad-digital/>
- [34] single sign-on (SSO). [En línea]. Disponible en: <https://www.techtarget.com/searchsecurity/definition/single-sign-on>
- [35] I. Gómez, “Tres proyectos enfocados en identidad digital descentralizada sobre blockchain,” *Criptonoticias*, Jun 2020. [En línea]. Disponible en: <https://www.criptonoticias.com/tecnologia/tres-proyectos-identidad-digital-descentralizada>
- [36] A. Preukschat. (2018, Jan) Self Sovereign Identity — a guide to privacy for your digital identity with Blockchain. [En línea]. Disponible en: <https://medium.com/@AlexPreukschat/self-sovereign-identity-a-guide-to-privacy-for-your-digital-identity-5b9e95677778>
- [37] R. Bean, “Getting To Trusted Data Via AI, Machine Learning And Blockchain,” *Forbes*, Jun 2018. [En línea]. Disponible en: <https://www.forbes.com/sites/ciocentral/2018/06/17/getting-to-trusted-data-via-ai-machine-learning-and-blockchain/?sh=11d3c6d69b49>
- [38] Unknown. Decentralized Identity. [En línea]. Disponible en: <https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE2DjfY>
- [39] Decentralized identifiers (dids) v1.0. [En línea]. Disponible en: <https://w3c.github.io/did-core/>
- [40] Decentralized public key infrastructure (dpki): What is it and why does it matter? [En línea]. Disponible en: <https://hackernoon.com/decentralized-public-key-infrastructure-dpki-what-is-it-and-why-does-it-matter-babee9d88579>
- [41] (2018) Decentralized Identity: Own and control your identity. [En línea]. Disponible en: <https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE2DjfY>
- [42] Configuración del inquilino para las credenciales verificables Azure AD. [En línea]. Disponible en: <https://docs.microsoft.com/es-ES/azure/active-directory/verifiable-credentials/verifiable-credentials-configure-tenant>

- [43] Emisión de credenciales verificables de Azure AD desde una aplicación. [En línea]. Disponible en: <https://docs.microsoft.com/es-es/azure/active-directory/verifiable-credentials/verifiable-credentials-configure-issuer>
- [44] Configuración del comprobador de credenciales verificables de Azure AD. [En línea]. Disponible en: <https://docs.microsoft.com/es-es/azure/active-directory/verifiable-credentials/verifiable-credentials-configure-verifier>
- [45] Microsoft Entra Verified ID: Enable more secure interactions while respecting the privacy of individuals. [En línea]. Disponible en: <https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE4XVOH?culture=es-es&country=ES>
- [46] We build CyberTrus, We create CyberSecurity. [En línea]. Disponible en: <https://wsg127.com/>
- [47] General data protection regulation. [En línea]. Disponible en: <https://gdpr-info.eu/>
- [48] Alastria. [En línea]. Disponible en: <https://alastria.io>
- [49] ConsenSys. Build on Quorum, the complete open source blockchain platform for business. [En línea]. Disponible en: <https://consensys.net/quorum/>
- [50] La identidad digital de Alastria presenta su primer MVP. [En línea]. Disponible en: <https://alastria-es.medium.com/la-identidad-digital-de-alastria-presenta-su-primer-mvp-696750d687ac>
- [51] Welcome to AlastriaID. [En línea]. Disponible en: <https://github.com/alastria/alastria-identity/wiki/Artifacts-and-User-Stories-Definitions>
- [52] Microsoft project. conoce la versión sencilla, eficaz y reinventada de project para todos. [En línea]. Disponible en: <https://www.microsoft.com/en-us/microsoft-365/project/project-management-software>
- [53] Pallets. Flask. [En línea]. Disponible en: <https://flask.palletsprojects.com/en/2.2.x/>
- [54] D. software fundation. Django makes it easier to build better web apps more quickly and with less code. [En línea]. Disponible en: <https://www.djangoproject.com/>
- [55] Ganache ONE CLICK BLOCKCHAIN. [En línea]. Disponible en: <https://trufflesuite.com/ganache/>
- [56] web3 5.30.0. [En línea]. Disponible en: <https://pypi.org/project/web3/>
- [57] What Is SQLite? [En línea]. Disponible en: <https://www.sqlite.org/index.html>

BIBLIOGRAFÍA

- [58] T. Suite. Ganache UI. [En línea]. Disponible en: <https://github.com/trufflesuite/ganache-ui/releases>
- [59] P. Santos-Cabaleiro. SignUp Blockchain. [En línea]. Disponible en: <https://github.com/pablosantos1106/SignUpBlockchain>