UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Prototipado de un sistema de gestión de identidad autosoberana

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Illán García, Ismael

Tutor/a: Muñoz Escoí, Francisco Daniel

Cotutor/a externo: ARJONA AROCA, JORGE

CURSO ACADÉMICO: 2022/2023

Resumen

Este trabajo aborda la creciente preocupación sobre la gestión de la identidad personal en el mundo digital. La identidad digital se ha convertido en un elemento esencial de la vida cotidiana, pero su crecimiento ha generado inquietudes acerca de la privacidad y el control de los datos personales. La Identidad Autosoberana es un concepto innovador que pretende devolver el control sobre su identidad digital a cada persona.

Este trabajo tiene como objetivo explorar en profundidad el concepto de Identidad Autosoberana, analizando su evolución histórica, los beneficios que presenta respecto a sus predecesoras y sus componentes técnicos. Se investigan las tecnologías que hacen posible esta nueva forma de identidad digital, como la tecnología de la cadena de bloques (blockchain), la criptografía y los estándares de las diferentes organizaciones involucradas.

Palabras clave: Identificadores Descentralizados, DID, Credenciales Verificables, VC, Identidad Digital, Cadena de bloques

Abstract

This work addresses the growing concern about the management of personal identity in the digital world. Digital identity has become an essential element of everyday life, but its growth has raised concerns about privacy and control over personal data. Self-Sovereign Identity is an innovative concept that aims to give back control over digital identity to each individual.

This work aims to delve deeply into the concept of Self-Sovereign Identity, analyzing its historical evolution, the benefits it presents compared to its predecessors, and its technical components. The technologies enabling this new form of digital identity are investigated, such as blockchain technology, cryptography, and the standards of various involved organizations.

Key words: Decentralized Identifiers, DID, Verificable Credentials, VC, Digital Identity, Blockchain

Índice general

Índice de figuras

Índice de tablas

Índice de listados

CAPÍTULO 1
Introducción

En la era de la información y el mundo digital, la gestión y protección de la identidad personal ha adquirido una relevancia cada vez más notoria. Con el crecimiento exponencial de servicios en línea, redes sociales, plataformas electrónicas y aplicaciones, la identidad digital se ha convertido en una parte fundamental de la vida cotidiana. Este aumento en la digitalización de nuestras vidas ha generado preocupaciones sobre la privacidad, la seguridad y el control que las personas tienen sobre sus datos.

Uno de los principales problemas existentes es la falta de conciencia que las personas tienen sobre su identidad digital. Al inicio de Internet, las identidades digitales se limitaban a perfiles con nombres de usuario inventados y avatares. Con el tiempo, el mundo digital ha crecido exponencialmente, pasando de simples foros a la compleja red que conocemos hoy en día. Esta evolución ha llevado a un aumento significativo en la cantidad de información que se comparte en línea.

En la actualidad, la identidad digital de cada individuo no se limita solo a su información personal, sino que también está formada por las interacciones que tienen en línea y los datos generados por los dispositivos que utilizan para acceder a la red. Desafortunadamente, la mayoría de las personas no son conscientes de lo que realmente sucede con sus datos en este entorno digital.

Toda esta nueva información disponible ha dado lugar a nuevos modelos de negocio basados en el manejo y uso de la gran cantidad de datos recopilados en el mundo digital. Utilizando o revendiendo dichos datos, los comerciantes pueden financiar productos ofrecidos gratuitamente o con descuento a los consumidores, que "pagan" con su atención. Si bien esto ha permitido el desarrollo de experiencias más personalizadas para los usuarios, como el inicio de sesión único, también ha surgido el problema de organizaciones que hacen un uso indebido de la información de sus usuarios.

Ante esta preocupante situación, diversas organizaciones de todo el mundo han estado investigando mecanismos para combatir estos casos y proteger la privacidad y dignidad de las personas. El objetivo es encontrar soluciones que permitan un manejo más responsable y transparente de los datos personales, de manera que los usuarios tengan un mayor control sobre su identidad digital y se sientan seguros al interactuar en línea. La protección de la privacidad se ha convertido en un pilar fundamental para garantizar una experiencia digital más ética y confiable para todos.

Es aquí donde surge el concepto de identidad autosoberana como una solución innovadora para abordar los desafíos actuales de la gestión de las identidades en el mundo digital. La identidad autosoberana es un paradigma emergente que coloca el poder y el control directamente en manos del individuo, permitiéndole tomar decisiones informadas sobre cómo, cuándo y con quién compartir su identidad digital, así como qué fragmentos de esta.

1.1 Motivación

Este trabajo comenzó durante mis prácticas curriculares en el ITI, durante mi tercer año de carrera. El objetivo de estas prácticas fue el aprendizaje sobre las tecnologías blockchain. Durante este período, participé en el proyecto IVACE-FEDER Épsilon. Este proyecto era una continuación de un proyecto anterior llamado Delta, cuyo objetivo era la sincronización del contenido de una red blockchain con una base de datos relacional o NoSQL para facilitar su lectura. El añadido que se propuso en Épsilon fue permitir la emisión de transacciones en la blockchain a partir de una interfaz de alto nivel similar a SQL, estableciendo así una fuerte abstracción entre el usuario y el uso de las blockchain. Mis tareas consistieron en la revisión de los contratos inteligentes, realización de pruebas de rendimiento y visualización de los datos obtenidos. Estas prácticas me permitieron obtener experiencia en el uso y despliegue de dos redes blockchain permisionadas: Hyperledger Fabric y Quorum.

Al final de mi periodo de prácticas, me ofrecieron un puesto en la empresa, el cual comencé en septiembre de 2022. Durante el primer mes contratado en el ITI, colaboré en la investigación y redacción de una publicación sobre DELTA [33], para la cual revisé los diferentes borradores del artículo y participé en el diseño y realización de la evaluación experimental correspondiente. El artículo se presentó y fue aceptado para la conferencia internacional sobre blockchain y criptodivisas de IEEE, ICBC.

Cuando terminamos el artículo, el grupo se centró en el nuevo proyecto IVACE-FEDER para el año 2022-2023: DLT4AITOYS. Este proyecto se presentó en conjunto con AIJU. En este contexto tuve que realizar un análisis sobre el estado del arte relacionado con la gestión de la identidad federada. Durante la realización de esta tarea me encontré con la existencia de un nuevo paradigma sobre la gestión de identidades. La identidad autosoberana aparecía apoyada por W3C, ya que acababa de generar hacía unos meses un par de recomendaciones sobre DID y credenciales verificables (VC), componentes centrales de la identidad autosoberana. También me permitió descubrir la falta de implementaciones que respetasen dichas especificaciones, que aún recientes sus ideas llevaban en desarrollo desde hace años. Cumplir con dichas especificaciones no supone cumplir una serie de estrictas reglas, son simples recomendaciones con carácter neutro con el ánimo de facilitar la interoperabilidad.

Mi primer acercamiento con la identidad autosoberana fue la lectura de las recomendaciones comentadas. Esta fue mi primera experiencia real leyendo especificaciones tan extensas (casi 250 páginas entre las dos). Ya en la primera lectura, nos dimos cuenta de que la identidad autosoberana no estaba vinculada únicamente a las tecnologías blockchain y que podía implementarse en una variedad de tecnologías y arquitecturas diferentes.

Una de las principales motivaciones de este trabajo es comprender por qué la identidad autosoberana aún no se ha implementado realmente en el día a día de las personas, a pesar de llevar en desarrollo más de siete años. Por este motivo, será necesario descubrir las dificultades que presentan las soluciones basadas en identidad autosoberana, así como revisar los proyectos que están trabajando en su progreso.

1.2 Objetivos

Este trabajo tiene como principal objetivo llevar a cabo el prototipado de un sistema de gestión de identidad autosoberana. Para poder realizar este desarrollo, se llevará a cabo un análisis de los objetivos y dificultades de los componentes necesarios en el siste-

ma. Además, se deberán comparar los diferentes proyectos y estándares disponibles para la implementación del prototipo. La solución a desarrollar deberá estar respaldada por los avances de la comunidad, por lo que se presentarán otras opciones disponibles y las diferencias que presentan para cada elección.

El logro de estos objetivos fortalecerá el aprendizaje sobre cómo llevar a cabo dicha gestión y proporcionará experiencia en el área. Esto será especialmente útil en un futuro cercano, ya que es previsible que la identidad autosoberana se convierta en el modelo predominante en las interacciones que tanto las personas como las empresas realicen diariamente en relación con nuestra identidad.

1.3 Estructura de la memoria

En esta sección, se presentarán los contenidos que se abordarán en el resto de la memoria. Esta parte marca el final del primer capítulo, en el cual se ha introducido el tema de este proyecto, explicado la motivación detrás de su realización y establecido los objetivos que se pretenden lograr al final de este.

El siguiente capítulo presentará el estado del arte respecto a la identidad autosoberana. Este será un capítulo considerablemente más extenso que los demás, destinado a adquirir los conocimientos necesarios para comprender el resto del proyecto. Además, se destacarán las organizaciones y proyectos más relevantes en la comunidad relacionada.

Una vez completado el estado del arte, se realizará un análisis para comparar todas las opciones disponibles. Con base en dicho análisis, se tomarán decisiones importantes, incluyendo nuestra propuesta de proyecto, por lo que habrá que estudiar en qué aspectos podremos realizar aportaciones.

Una vez decidida la propuesta, se explicará el diseño que se ha preparado para el sistema de gestión. Durante este capítulo, se describirán en detalle los protocolos elegidos. Posteriormente, en el capítulo de desarrollo, se detallará el proceso de implementación y los posibles cambios que presentará nuestra solución en comparación con las especificaciones elegidas durante el diseño. Concluida la fase de desarrollo, se presentarán unas guías de despliegue para los diferentes componentes, así como los requisitos necesarios para la integración. En el siguiente capítulo, se realizarán pruebas sobre el sistema desplegado y se presentará una demostración de sus funcionalidades.

Al final, se desarrollarán las conclusiones sobre el trabajo realizado, presentando nuestros pensamientos finales sobre el tema tratado. También se explicará la relación que pueda tener el trabajo con los estudios cursados durante la carrera. Como último capítulo, se detallarán los posibles caminos que se podrían elegir para continuar y expandir el trabajo realizado durante este proyecto.

CAPÍTULO 2
Estado del arte

Con el nacimiento de la web y de las primeras comunidades virtuales, la identidad digital comenzó a manifestarse como un usuario y una foto de perfil. Internet ha permitido expandir nuestro concepto del "yo" hasta el punto de poder conectarnos con un número impensable de gente hace unas décadas, pero a la vez ha crecido sin proporcionar al sujeto control total sobre su propia identidad.

El crecimiento de la "red de redes" que conocemos hoy en día se produjo gracias a tecnologías como los websockets o TCP/IP, protocolos que permiten identificar a las máquinas durante su comunicación, pero no había forma de identificar a las personas que las controlaban.

Internet se construyó sin una base de identidad, como dijo Kim Camerom [9], jefe de identidad de Microsoft: «Internet se creó sin una forma de saber a quién y a qué te conectas. Esto limita lo que podemos hacer con ella y nos expone a crecientes peligros. Si no hacemos nada, nos enfrentaremos rápidamente a incidentes de robo y engaño que irán erosionando la confianza del público en Internet». Lo que Kim intentó predecir en su artículo era que Internet aún no estaba preparada para resolver el problema de la identidad digital.

2.1 Evolución de la identidad digital

Una posible definición de qué es la identidad digital podría ser: el conjunto de afirmaciones o "claims" sobre un sujeto, hechas por sí mismo u otra entidad. Por definición entendemos un sujeto como la persona u objeto sobre la que se está discutiendo, describiendo o tratando. El uso principal de esta identidad digital es la autenticación y autorización del sujeto de la identidad para acceder a recursos y servicios.

2.1.1. Modelo centralizado

Al principio de internet eran pocas las organizaciones que empezaron a ofrecer algún tipo de servicio donde se necesitara identificar a sus usuarios, así que cada empresa comenzó a gestionar las identidades de una forma centralizada. Los usuarios debían registrarse en la organización, aportando una serie de datos sobre ellos mismos, y la organización era la encargada de crearles un perfil en su gestor de identidades. Todos los datos de los usuarios eran registrados en la base de datos y por lo tanto centralizados en el lado de la organización. El usuario dependía completamente del buen funcionamiento del gestor de identidades para el uso del servicio, ya que en caso de fallo perdía totalmente el acceso. El usuario también confiaba en la buena fe de la organización a la hora

de administrar sus datos, ya que estos pasaban a estar controlados por los gestores centralizados de cada empresa. En sus primeras etapas Facebook fue un buen ejemplo, tanto de un sistema centralizado, pues cada usuario tenía que crearse una cuenta en su plataforma para acceder a su servicio (la red social), como de una organización que aprovechó la centralización de los datos de los usuarios para su comercialización [5].

Este modelo de gestión centralizado se volvió muy popular por la sencillez que ofrecía a organizaciones y empresas, para el usuario ofrecía una experiencia muy simple donde este no debía gestionar sus datos, solo tenía que recordar su contraseña, y en caso de robo de identidad el daño era muy limitado, ya que la identidad estaba limitada a solo un servicio. Para las empresas ofrecía la comodidad de poder tener un acceso constante a los datos almacenados de los usuarios. La centralización tiene también sus inconvenientes, al reunir todos los datos de los usuarios en las bases de datos, estas se volvían objeto de ataques para agentes maliciosos o hackers con el objetivo de filtrar toda esa información.

Conforme más proveedores de servicios surgían, más identidades digitales o cuentas debía mantener cada usuario, haciendo más complicada su experiencia. Además, al estar las identidades aisladas en cada servicio muchas veces se producía la duplicación de identidades, cuando el usuario tenía dos cuentas en dos servicios diferentes para las que proporcionó los mismos datos durante el registro.

2.1.2. Modelo federado

Con el tiempo algunas organizaciones que crecieron mucho empezaron a ofrecer sus servicios de gestión de identidad a otras organizaciones, haciendo de conector entre las personas y las empresas [41]. A diferencia del modelo centralizado donde el proveedor de servicio hacía a la vez de proveedor de identidades, en el modelo federado la identidad del usuario está almacenada por una organización que hace de proveedor de identidades y con esta identidad el usuario podría acceder a distintos servicios. Un ejemplo de la gestión federada de la identidad es Google: un usuario, a través de su cuenta puede acceder a los servicios de la "federación" de Google, organizaciones que aceptan la cuenta de Google como identificación suficiente para usar sus servicios sin necesidad de otro registro.

La existencia de una federación de servicios planteó el siguiente problema. A pesar de poder acceder a los diferentes servicios dentro con la misma identidad había que seguir iniciando sesión en cada uno de ellos. Para solucionar esto se presentó el "Single sign-on". Esta nueva función para la gestión de sesiones permite pasar a un nuevo servicio de la federación sin necesidad de autenticarte si ya habías iniciado sesión en otro previamente. Por ejemplo, un usuario de una empresa podría iniciar sesión con su cuenta empresarial en Outlook y luego abrir una pestaña de Teams sin tener que volver a iniciar sesión. Una vez dentro de los servicios de la federación el usuario podrá utilizar el resto de los servicios sin tener que realizar el proceso de iniciar sesión para cada servicio.

La centralización o unión de lo que antes eran varias cuentas a una sola también supone un aumento proporcional en el daño para el usuario en el caso de robo de identidad, ya que el suplantador tendría acceso a todos los servicios de la federación. Aparte, estas empresas encargadas de proporcionar la identidad federada tienen mucha más responsabilidad que en el esquema centralizado, ya que en caso de caída del servicio el usuario perdería acceso no a uno, sino a todos los servicios que accedía con esa cuenta.

Al igual que con la identidad centralizada es muy difícil que exista solo un proveedor de servicio, en la identidad federada no se puede asumir que solo se formará una federación. Por lo tanto, el usuario tendrá que crearse más cuentas, una para cada federación, lo que no supone un avance, sino solo una reducción en las cuentas que deberá mante-

ner cada persona. En este contexto se han propuesto mejoras para la interoperabilidad. Varias federaciones podrían formar ciertos acuerdos de confianza, lo que permitiría a los usuarios traspasar una cuenta de una federación a otra [40].

La identidad federada reduce en cierta medida el grado de centralización que teníamos con el modelo anterior, ya que los datos ya no los gestiona la empresa proveedora del servicio, sino otra organización diferente que hace el papel de gestor de identidades. El problema es que estas organizaciones siguen siendo entidades centralizadas, pero ahora con el añadido de que el usuario puede elegir a qué entidad centralizada le da el control de sus datos.

2.1.3. Identidad autosoberana

La identidad autosoberana (SSI, self-sovereign identity) es el último paradigma de gestión de identidades. Al modelo SSI también se le hace referencia como identidad descentralizada, al eliminar del esquema a las organizaciones a las que pertenecían las identidades, para devolver su propiedad y control a los usuarios. A partir de este momento cada persona es su propio gestor de identidades. Como la definió Christopher Allen ya en 2016[4]: «El usuario debe ser el centro de la administración de la identidad. Requiere no sólo la interoperabilidad de la identidad del usuario a través de múltiples ubicaciones, con el consentimiento del usuario, sino también el verdadero control de su identidad digital, creando así la autonomía del usuario. Para lograr esto, una identidad autosoberana debe ser transportable; no se puede limitar a un sitio o localidad. Una identidad autosoberana debe también permitir que los usuarios hagan afirmaciones, que podrían incluir información personal o hechos sobre capacidades personales o pertenencias a grupos. Incluso podría contener información sobre el usuario que fue afirmada por otras personas o grupos». En ese mismo artículo, Allen también propuso los 10 principios de la identidad autosoberana:

1. Existencia. Los usuarios tienen una existencia independiente a la identidad digital. La identidad autosoberana está basada en la identidad real de las personas y por lo tanto no puede existir solo de forma digital. La identidad autosoberana solo hace públicos algunos aspectos del "yo" que ya existían.

2. Control. Los usuarios deben tener el poder de controlar sus identidades. Deben tener la autoridad para publicar, modificar y ocultar cualquier aspecto sobre ella. Esto no quiere decir que controlen también todas las "claims" que forman su identidad, ya que estas pueden haber sido generadas por otros usuarios.

3. Acceso. Los usuarios deben tener acceso a sus datos. Un usuario siempre tiene que poder acceder a su información y debe poder ser consciente de todas las afirmaciones que hay respecto a él.

4. Transparencia. Los sistemas y algoritmos deben ser transparentes. Los sistemas usados para administrar los datos y la red de identidades deben ser gratis y de código abierto, cualquiera debe ser capaz de examinar cómo funcionan.

5. Persistencia. Las identidades deben poder ser longevas. Las identidades deben durar toda la vida, o hasta que el usuario desee. En un mundo de constante desarrollo como es Internet esto resulta muy difícil por lo que las identidades deben durar al menos hasta que aparezca otro sistema de identidades que lo sustituya. Este principio no debe ser incompatible con el derecho "al olvido" [3].

6. Portabilidad. La información de las identidades debe poder ser transportada. No deben ser mantenidas por una entidad de terceros, incluso aunque el usuario confíe en dicha entidad. Poder transportar las identidades permite a los usuarios protegerlas ante cambios externos y por lo tanto también aumenta su persistencia.

7. Interoperabilidad. Las identidades deben poder ser ampliamente utilizadas. Las identidades no tienen valor si se mantienen disponibles solo para nichos. El objetivo es conseguir identidades que puedan ser utilizadas globalmente manteniendo el control de sus usuarios.

8. Consentimiento. Los usuarios deben aceptar el uso de sus identidades. La interoperabilidad supone el intercambio y compartición de la información relacionada con las identidades, pero esto solo debe ocurrir bajo el consentimiento de sus dueños.

9. Minimización. La divulgación de la información debe ser mínima. En el caso de tener que publicar información esta debe ser la mínima posible para conseguir el objetivo. Por ejemplo, ante una prueba de mayoría de edad no se debería compartir la edad exacta, y en el caso de la fecha de nacimiento, no se debería compartir con completa precisión. Se deben utilizar medidas para preservar la privacidad al máximo.

10. Protección. Los derechos de los usuarios deben ser protegidos. Ante un conflicto de intereses entre los sistemas de identificación y los derechos de los usuarios, el sistema siempre debería elegir mantener los derechos.

Para apoyar el desarrollo del concepto de la identidad autosoberana, han ido apareciendo grupos con sus aportaciones. El World Wide Web Consortium (W3C) ha trabajado para la estandarización de estas ideas, dando como resultado su recomendación para los identificadores descentralizados [45] y las credenciales verificables [43]. Estos conceptos son el apoyo técnico para respaldar el concepto de autosoberanía de la identidad. La Decentralized Identity Foundation (DIF) ha formado grupos de trabajo encargados del desarrollo de los elementos necesarios para el ecosistema de la identidad descentralizada, generando así implementaciones de referencia para los desarrolladores. Otra de las organizaciones más influyentes en la comunidad es la Hyperledger Foundation, organización encargada de crear proyectos blockchain de código abierto y colaborativos, que desde hace años ha formado todo un stack de proyectos en torno a la identidad autosoberana. En los siguientes apartados se explicarán los elementos necesarios sobre los que se espera que avance la SSI.

2.2 Criptografía

Para comprender mejor los elementos que conforman la identidad autosoberana vamos a presentar una serie de conceptos criptográficos. Entender los límites y las propiedades de los sistemas criptográficos permite entender por qué se utilizan en diferentes situaciones.

2.2.1. Clave secreta

La criptografía de clave privada utiliza una serie de claves para cifrar el mensaje, el cual solo podrá ser leído por otra persona con la misma clave con la que se cifró. Este sistema también se conoce como criptografía simétrica, ya que ambos extremos de la comunicación comparten la misma clave.

Uno de los factores más importantes que afectan a la seguridad del sistema es la longitud de la clave privada. Algunos de los ejemplos más conocidos son el *Advanced Encryption Standard* (AES) [31] o el *International Data Encryption Algorithm* (IDEA) [23]. Una de las recomendaciones a tener en cuenta a la hora de utilizar algoritmos criptográficos es elegir siempre implementaciones realizadas por expertos en criptografía. Por norma general, un desarrollador de aplicaciones nunca deberá encargarse del software encargado de la criptografía.

Uno de los principales problemas de este sistema y que plantea más dudas sobre la seguridad es cómo compartir la clave privada que ambos usuarios deben tener. En el caso de enviar dicha clave por internet, esta podría ser interceptada y comprometer el canal de comunicación. Otro de los ataques más comunes es almacenar una gran cantidad de mensajes cifrados con la misma clave para, a través de un análisis, obtener la clave. Para solucionar este problema, la clave debe renovarse con frecuencia, lo que nos devuelve al primer problema comentado.

2.2.2. Clave pública

Como se ha explicado, el principal problema que existe con el sistema anterior es el intercambio de claves. Para solucionar este problema, se presentaron los sistemas basados en la criptografía de clave pública. Estos sistemas utilizan dos claves distintas: una clave pública y una clave privada. La clave privada debe ser almacenada por el dueño y no se debe revelar, mientras que la clave pública puede ser compartida con otros usuarios, a través de mensajes o incluso expuesta en una página web. Las dos claves están relacionadas matemáticamente, por lo que se les suele referenciar como un par [20]. Esta relación matemática permite que un mensaje encriptado con una clave pública pueda ser desencriptado solo por la clave privada correspondiente. Por este motivo, este sistema también se conoce como criptografía asimétrica. Es importante tener en cuenta que es imposible descifrar un mensaje con la misma clave con la que se cifró.

El flujo de comunicación propuesto sería el siguiente: Alicia desea enviar un mensaje a Bob. Por lo tanto, descarga la clave pública de Bob para cifrar el mensaje antes de enviarlo. Aunque el mensaje fuese interceptado, únicamente Bob sería capaz de descifrarlo utilizando su clave privada. Como se mencionó previamente, también existe la posibilidad de cifrar el mensaje con la clave privada de Alicia, lo que se conoce como firma digital. En esta situación, se sacrifica la confidencialidad de la comunicación, ya que cualquier persona podría acceder a la clave pública de Alicia. Sin embargo, esta técnica puede ser relevante en diversas circunstancias ya que, al cifrar con su clave privada, Alicia certifica que el mensaje proviene de ella y no puede negar su autoría. Además, esta práctica permite demostrar que el contenido del mensaje no ha sido alterado durante su envío, dado que, en caso contrario, no podría ser descifrado utilizando su clave pública.

2.2.3. Sistemas híbridos

Los sistemas basados en clave pública son computacionalmente mucho más costosos que los basados en claves simétricas. Por este motivo no se suelen utilizar para cifrar grandes cantidades de información, en su lugar, se utilizan para negociar una clave secreta entre ambos extremos de la comunicación. Esto es lo que se conoce como sistemas híbridos o intercambio de claves de Diffie-Hellman.

Durante un intercambio de claves basado en Diffie-Hellman, los extremos utilizan un sistema de clave pública para compartir un par de números primos considerablemente grandes, el más pequeño del par tiene el nombre de generador. Una vez ambos tengan

el mismo par de números ambos podrían calcular la clave secreta de su comunicación. La clave se calcula a partir de la clave privada propia, la clave pública del otro extremo y el par de primos compartido. El proceso es conmutativo por lo que la otra parte de la comunicación realizaría lo mismo y ambos obtendrían la misma clave sin necesidad de haberla compartido [24].

2.2.4. Funciones Hash

A veces es suficiente saber si un documento se ha modificado o no. En estos casos, la criptografía de hashes permite ahorrar en el coste computacional de la encriptación. Las funciones criptográficas hash consisten en una función que, a partir de una cadena de tamaño variable, genera otra cadena de tamaño fijo a través de una serie de operaciones matemáticas [34]. Este tipo de funciones presentan una serie de propiedades muy importantes:

- Irreversibilidad: debe ser imposible obtener el mensaje original a partir del hash generado. La irreversibilidad nos asegura poder compartir el hash de un mensaje sin comprometer el contenido original.
- Imposibilidad de correlación: pequeños cambios en la cadena de entrada deben producir cambios drásticos en el hash. Sin esta propiedad, un atacante podría deducir la entrada utilizada a partir de entradas similares.
- Unicidad: encontrar dos entradas que produzcan el mismo hash debe ser extremadamente improbable. Lograr hashes únicos evita la posibilidad de sustituir un mensaje con otro que tenga el mismo hash.

Estas propiedades son fundamentales para garantizar la integridad y la autenticidad de los datos mediante la verificación de hashes.

2.3 Identificadores descentralizados

Los identificadores descentralizados o DID son un nuevo tipo de identificadores diseñados para apoyar el esquema de la identidad autosoberana. Estos DID permiten la creación de identidades digitales verificables y descentralizadas. Los DID permiten relacionar al sujeto de la identidad con el documento DID. Estos documentos contienen una serie de claves públicas, métodos de verificación y servicios que permiten al controlador del identificador demostrar su propiedad [45].

Los identificadores descentralizados están compuestos por tres partes (Figura 2.1): el nombre del esquema representado por el prefijo “did:”, el identificador del método DID con el que se creó, y el identificador específico asignado por el método DID. La especificación del W3C es una recomendación por lo que habrá proyectos que no la respeten del todo para poder adaptarse mejor a sus necesidades.

Los métodos DID son mecanismos que especifican como se deben crear, leer, modificar y borrar los DID. Por ejemplo, un método podría establecer que el identificador específico esté basado en la clave pública del par de claves (pública y privada) usado para crear el DID, ya que al ser el grado de colisión tan bajo entre claves públicas, se asegura la propiedad de unicidad de los identificadores. Actualmente hay una gran cantidad de métodos, el W3C recomienda utilizar métodos ya definidos para poder así mejorar la interoperabilidad. Como se ha mencionado antes, algunos métodos hacen variaciones

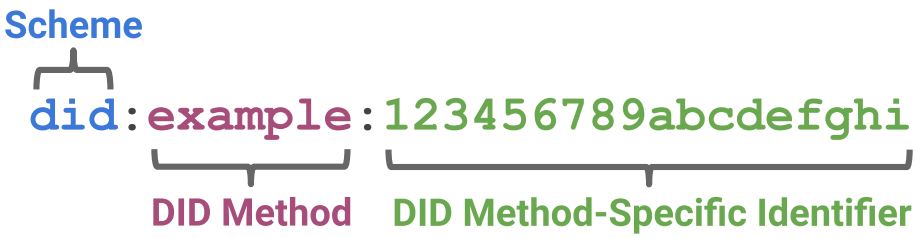

Figura 2.1: Estructura de un DID.

respecto a la recomendación, como aquellos que funcionan en la red de Ethereum y sus derivados, que emplean un campo más para definir la red donde está registrado: <Esquema>:<Método>:<Red>:<Identificador>. Por ejemplo en los identificadores de tipo: "did:ethr:goerli:" o "did:ethr:polygon:".

Los DID son un tipo de identificadores de recurso universales, o URI. Su principal función es relacionar al sujeto con información sobre él, esta información se guarda en el documento DID. Dicho documento suele contener el DID del sujeto junto con una serie de métodos de verificación y servicios, establecidos por el controlador del identificador, para poder realizar acciones como la autenticación o autorización.

```
0  {
1    "@context": [
2    "https://www.w3.org/ns/did/v1",
3    "https://w3id.org/security/suites/ed25519-2020/v1"
4    ]
5    "id": "did:example:123456789abcdefghi",
6    "controller": "did:example:12345678controlador",
7    "authentication": [{
8      "id": "did:example:123456789abcdefghi#keys-1",
9      "type": "Ed25519VerificationKey2020",
10     "controller": "did:example:12345678controlador",
11     "publicKeyMultibase": "zH3C2AVvLMv6gmMNam3
         uVAjZpfkcJCwDwnZn6"
12   }]
13 }
```

Listado 2.1: Ejemplo de documento DID

En el documento de ejemplo (Listado 2.1) se puede ver que el identificador sería el primer campo "id", y también se puede observar que justo abajo aparece un atributo controller. Esto podría dar a entender que el sujeto puede no ser el controlador de la identidad, y precisamente los DID permiten este comportamiento.

En la mayoría de los casos el sujeto del DID será su controlador, pero la recomendación del W3C pretende soportar diferentes escenarios de uso. Por ejemplo, una situación en la que un niño necesite identificarse, pero necesite que su tutor legal administre su identidad hasta que tenga la mayoría de edad. El DID no está limitado a un controlador, puede establecerse un conjunto, como un grupo que hace de representante legal de una empresa, en estos casos es necesario diferenciar entre sujeto y controlador. El controlador asumiría el papel de gestor y sería único con autoridad para hacer actualizaciones sobre el documento DID. Aunque los ejemplos son sobre personas un DID no está limitado a identificar solo a estas. La recomendación está definida para identificar cualquier enti-

dad, como mascotas, empresas u objetos permitiendo así la creación de esquemas más complejos que podrán dar soporte a más variedad de escenarios.

Como se ha definido antes los DID sirven para identificar el documento que contiene la información del sujeto, pero no se ha comentado dónde se va a guardar ese documento. Los DID persiguen un esquema descentralizado de identificación y por lo tanto el sistema que se emplee para almacenar los documentos tiene que ser también descentralizado. En la recomendación, este elemento que se encarga de guardar los documentos se llama registro verificable. Este registro debe permitir la acción de resolver un DID en su documento. Para esto existen muchas opciones: ledgers distribuidos, sistemas de ficheros descentralizados, bases de datos o redes P2P. De todas estas opciones la elección más habitual son las redes blockchain, ya que debido a su naturaleza aseguran que el registro siempre estará disponible para los usuarios y que los documentos no sean alterados sin el consentimiento de su dueño. Muchos de los métodos anteriormente mencionados están basados en redes como Bitcoin, Ethereum, Sovrin o Alastria.

Para visualizar su funcionamiento se va a plantear un ejemplo. Alicia consultar qué información tiene su amigo Pedro publicada en el documento de uno de sus DID. El identificador de Pedro que Alicia quiere consultar es de tipo Alastria. Para poder consultar el documento Alicia deberá acceder a un nodo de la red que tenga permisos de lectura e introducir el DID de Pedro. Acto seguido el nodo le deberá devolvor toda la información relacionada con dicho identificador.

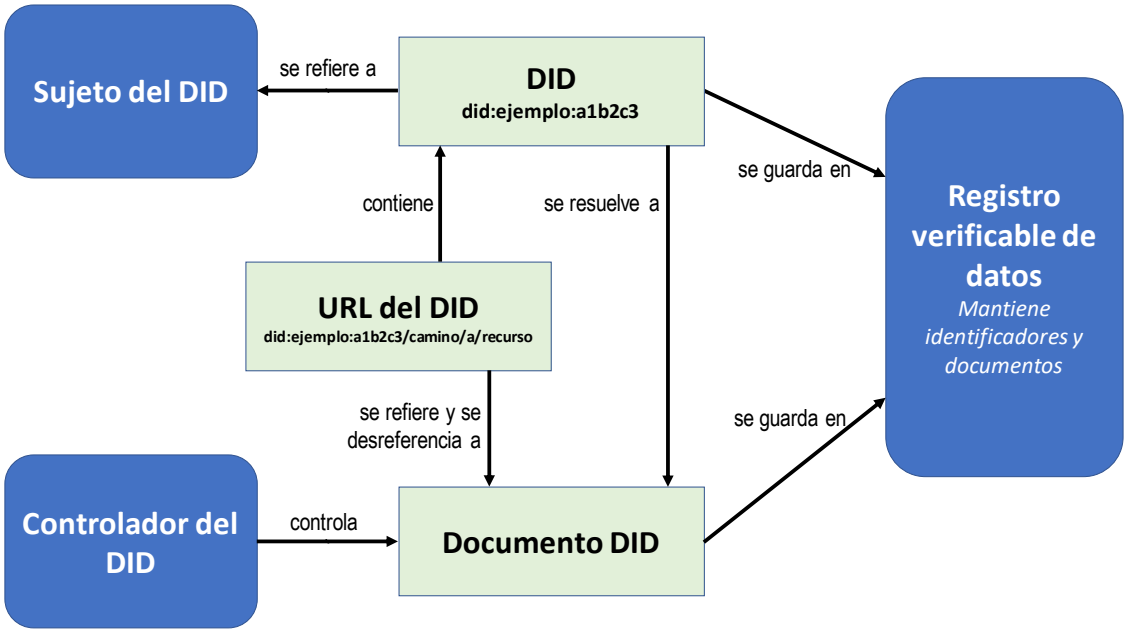

Figura 2.2: Elementos principales en la identificación mediante DID.

Para acompañar a la especificación del W3C sobre los identificadores descentralizados, DIF y el proyecto Hyperledger Aries han trabajado para crear otros estándares en paralelo que permitan el desarrollo de los DID, como el protocolo "DIDComm Messaging" que, a través de los DID, sus documentos y la criptografía disponible en estos, permite a dos controladores de DID establecer una comunicación segura [19, 12]. A partir de dicha comunicación se puede establecer un sistema de mensajería basado en identificadores descentralizados. Esto es especialmente útil para el siguiente pilar de la identidad autosoberana, las credenciales verificables.

2.4 Credenciales verificables

Las credenciales son un elemento que usamos en nuestro día a día en forma de documentos como el DNI, el carnet de conducir o certificados como una baja médica. Estas credenciales son los elementos que forman nuestra identidad, registros de texto que contienen la información que nos define. Normalmente una de estas credenciales contiene información sobre el sujeto de la credencial para poder identificarle, información sobre la entidad que autorizó la credencial y los datos que pretende representar la credencial, estos datos se suele conocer como atestaciones.

Estas credenciales que tenemos existen de forma física y aunque desde hace unos años se empezaron a usar los certificados digitales, estos no cumplen los objetivos de la identidad autosoberana. Los certificados digitales se pueden conseguir en España a través de entidades como la Fábrica Nacional de la Moneda y Timbre (FNMT). Para obtenerlo antes te has de autentificar a través de otro método, como puede ser presentar el DNI, y luego la entidad te proporciona un certificado firmado por ella confirmando tu identidad. Esto hace que los certificados digitales no cumplan el propósito de existencia de la identidad, la identidad de cada persona existe por sí misma y no debería haber una entidad centralizada que te la otorgue.

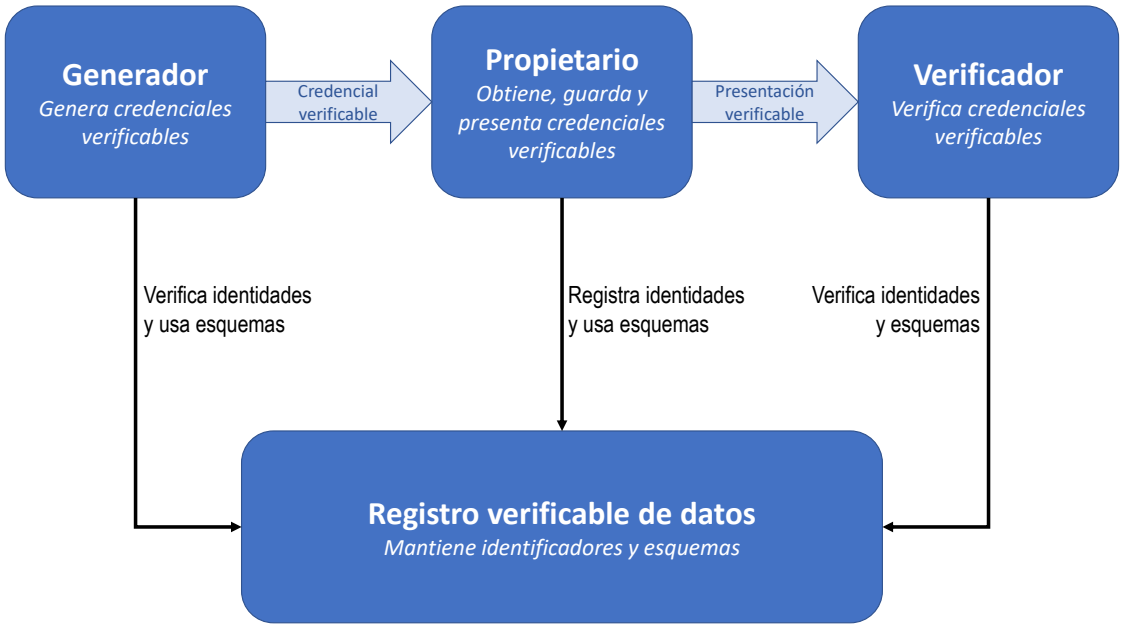

Figura 2.3: Arquitectura de gestión de credenciales verificables.

Para sustituir a los actuales sistemas de credenciales surgieron las credenciales verificables (VC), uno de los pilares de la identidad autosoberana, estandarizadas por el W3C ya en 2019. Este nuevo tipo de credenciales permite representar toda la información que ya contienen las credenciales físicas, pero con el añadido de tecnologías como las firmas, haciendo este tipo de credenciales a prueba de manipulaciones y más confiables que sus contrapartes físicas [43]. Cada persona podría llevar todas sus credenciales en su teléfono móvil, y podría generar presentaciones verificables para demostrar el control sobre las credenciales. Las credenciales verificables están pensadas también para preservar la privacidad de la información de cada persona. Sobre el conjunto de datos que puede tener una credencial, se presentarán los mínimos necesarios para cumplir con el proceso de verificación, cumpliendo así el principio de minimización que propuso Allen. Esto se puede complementar con nuevas tecnologías como las pruebas de conocimiento cero, que

a través de una serie de procesos criptográficos se puede confirmar un dato sin revelarlo, como demostrar que se reside en una ciudad sin especificar la dirección personal.

El uso de las credenciales verificables se basa en el esquema de los tres agentes presentado en la figura 2.3. En dicho esquema hay tres roles: generador, propietario y verificador. El generador o emisor es el encargado de emitir credenciales, firmarlas y para luego proporcionárselas a su propietario, el cual será el encargado de almacenarlas y tendrá la autoridad para administrarlas, para luego poder presentarlas a un verificador para poder obtener algún recurso o servicio. Para representar el flujo de las operaciones, vamos a ver un caso de uso.

Alicia quiere inscribirse a un curso de Máster en la Universidad de Alicante, durante el procedimiento se le exige presentar el certificado de que tiene un grado universitario. Para obtenerlo, Alicia necesitaría iniciar sesión en el portal de la universidad donde cursó el grado y presentar un DID para que la universidad pueda generar la credencial. Antes de generar la credencial, la universidad verifica que Alicia es la dueña de ese DID a través de una prueba con la clave pública de ese DID, si Alicia no tuviera la clave privada del par no sería capaz de superar la prueba y el proceso de generación de la credencial se detendría. Una vez autenticada Alicia y su DID, la universidad genera la credencial verificable, un documento que contiene el certificante de que Alicia posee el título universitario y la firma digital de la universidad como prueba de su autenticidad. Esa firma aparecería en la sección "proof" de la credencial, tal como ilustra la figura 2.4.

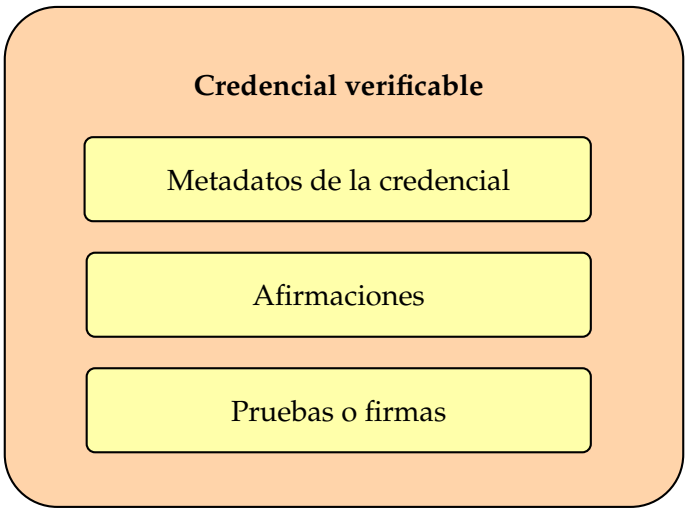

Figura 2.4: Estructura de una credencial verificable [43].

Después de recibir la credencial, Alicia podría generar una presentación verificable (VP), documento que contiene la misma información que la credencial verificable, pero con el añadido de la firma de Alicia (Figura 2.5), pues así de demuestra que la credencial se está empleando bajo su consentimiento. Cuando reciba la presentación verificable la universidad de Alicante actuaría como agente verificador. Primero comprobará la prueba de que la presentación es válida, utilizando la clave pública de Alicia para desencriptar la firma. Luego obtendrá la credencial que venía en la presentación para comprobar la firma de la otra universidad. Y por último, comprobará que la información que contiene la credencial, es efectivamente el certificado de que Alicia tiene un título universitario.

Otra mejora que se consigue respecto a las credenciales verificables es la velocidad en el flujo. Para poder realizar el caso de uso propuesto, Alicia debería pedir cita en su universidad, teniendo que esperar al menos un par de días, para luego tener que esperar a que la otra universidad tarde otro par de días en tramitarlo y verificarlo. Al tener las

credenciales verificables un formato entendible por las computadoras, este proceso se reduciría a minutos o incluso segundos.

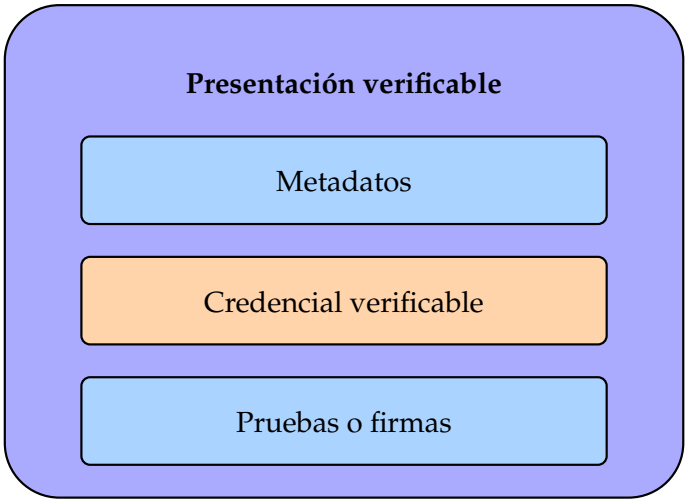

Figura 2.5: Estructura de una presentación verificable [43].

En este esquema de los tres agentes falta por mencionar el papel del registro verificable. Igual que con los identificadores el registro es el encargado de almacenar y mantener públicos los métodos de verificación, pero además con el añadido de las credenciales verificables también debe registrar todo lo necesario para la creación y verificación de estas. El esquema de datos de cada tipo de credencial para establecer un acuerdo de confianza entre emisor y verificador o el hash de la credencial generada por el emisor para comprobar que esa credencial no sufre modificaciones malintencionadas por el propietario o un tercero.

Además, de lo representado en el caso de uso anterior, las credenciales verificables emitidas por cualquier emisor pueden tener, igual que las credenciales físicas, una fecha de validez. En el caso de que la Dirección General de Tráfico (DGT) expida un permiso de conducción como credencial verificable puede establecer que sea válido durante 10 años. Además, cuando se generan las presentaciones verificables, el propietario podría establecer un tiempo de vida durante el cual la presentación es válida. Este mecanismo parece simple, pero es muy importante para solventar problemas como empresas almacenando estas presentaciones para luego comercializar los datos, igual que ahora con los datos de las cuentas de los usuarios. Dado que, al establecer un plazo de vigencia breve, o al menos el tiempo necesario para llevar a cabo el proceso de verificación, las empresas no obtendrían beneficio alguno al almacenar dichas presentaciones. Esto se debe a que, en poco tiempo, estas presentaciones perderían su validez.

Desde el punto de vista de la seguridad, las credenciales verificables están diseñadas para ser seguras ante cualquier tipo de robo. En el caso de que un agente malicioso intercepte la credencial emitida por un emisor e intente utilizarla, la credencial está relacionada con el DID del propietario legítimo por lo que el agente malicioso no tendrá acceso a las claves privadas para poder firmar la presentación, y aunque la firme con otra clave, el verificador al ver que no puede descifrar la firma con ninguna de las claves públicas del dueño de la credencial descartará la presentación generada por el atacante. Al tener las credenciales tanta seguridad y comprobaciones ante diferentes tipos de ataques, y ser la criptografía lo que otorga dicha seguridad, el objetivo de los atacantes se convertiría en conseguir dichas claves privadas, ya que con ellas podría suplantar la identidad del usuario. Este problema convierte el lugar donde se guarden las claves privadas en un elemento central para la seguridad del sistema.

La cartera digital o "wallet" es la herramienta digital que permite a los usuarios almacenar y administrar tanto sus pares de claves público-privadas como la colección de credenciales verificables. En el contexto de la identidad autosoberana se considera uno de los elementos técnicos fundamentales para proteger la privacidad y la seguridad de los usuarios.

Las carteras para credenciales verificables son especialmente útiles en entornos donde se requiere la validación frecuente de credenciales, como instituciones académicas, empleadores, organismos gubernamentales y organizaciones profesionales. Además, estas carteras también pueden impulsar la portabilidad de las credenciales, permitiendo a los individuos tener un mayor control y propiedad sobre su historial educativo y laboral.

2.5 Blockchain

Las recomendaciones del W3C sobre los identificadores descentralizados y las credenciales verificables se basan en un componente al que se refieren como registro verificable, el cual es una tecnología de almacenamiento que debe cumplir los requisitos de la SSI. Si bien estas recomendaciones no imponen un tipo específico de registro a utilizar, al observar los proyectos que han surgido, se puede notar claramente un favorito: las redes blockchain. Desde el surgimiento de la identidad autosoberana, estas dos áreas han estado estrechamente relacionadas, ya que comparten el objetivo de lograr un sistema descentralizado en el que los agentes puedan actuar de manera independiente.

El concepto de una red blockchain nació en el año 2008, cuando se publicó anónimamente bajo el pseudónimo de Satoshi Nakamoto un artículo que define un sistema de dinero electrónico entre pares [28]. La nueva tecnología introducida por Bitcoin permitía realizar pagos mediante transacciones que podían ser validadas y registradas por los agentes que participaban en la red. De esta manera, el protocolo de Bitcoin lograba establecer confianza entre agentes que no necesariamente se conocían en un sistema distribuido.

2.5.1. Componentes

La estructura de datos de una blockchain consiste en una lista creciente de registros de datos (bloques), los cuales están relacionados entre sí gracias a la criptografía hash. Cada bloque contiene un conjunto de datos nuevos (transacciones), así como el hash del bloque anterior. Este hash es lo que relaciona el bloque actual con el anterior, formando así la cadena de bloques. El hash, junto con una marca de tiempo (timestamp), hace que la modificación de los datos sea extremadamente difícil. En caso de que un atacante intente modificar un bloque, tendría que actualizar todos los bloques siguientes en la cadena [48].

Las transacciones son uno de los elementos principales que conforman las redes blockchain. Son registros de datos de bajo peso que se almacenan en las blockchains. Estas transacciones registran interacciones, como transferencias de fondos entre cuentas o la invocación de un contrato inteligente.

El siguiente elemento son las funciones hash. En el contexto de las redes blockchain el hash se usa como un puntero. Cada bloque que se genera en la red contiene el hash del bloque anterior, uniéndolos y formando una secuencia de bloques o cadena. Este puntero también serviría para verificar la integridad del bloque anterior, por lo que sistemáticamente se podría comprobar la integridad de los datos de toda la cadena de bloques.

En el contexto de las redes blockchain el uso de las firmas digitales asegura a un usuario que la transacción la ha emitido otro usuario en concreto. Una transacción en la red requiere que el emisor firme el hash de la transacción anterior y la clave pública del receptor. Esto permite verificar al emisor, y en el caso de que la transacción sea una transferencia de criptomonedas, asegurar que el nuevo dueño es el único que puede emitir la siguiente transacción que utilice esas criptomonedas.

Otro componente de las blockchain son los "Merkle trees". El objetivo de esta estructura es relacionar todos sus nodos con una raíz común. Para ello, se parte de una serie de nodos iniciales (hojas) que deben estar identificados por un hash. Estos nodos se asocian con un nodo superior (rama), el identificador de este nodo se generará a partir de la combinación de sus hojas. Este proceso se repite hasta conseguir el último elemento, el nodo raíz. En una red blockchain las hojas serían las transacciones que se van registrando, y el nodo raíz al estar formado por la combinación de los hashes, se utiliza para identificar al conjunto de transacciones que forman dicho árbol. Al utilizar una estructura basada en hashes, los Merkle trees son especialmente útiles para comprobar la integridad de un grupo de transacciones sin tener que hacerlo individualmente.

Los bloques son la estructura de datos contenedora de las blockchain. Tiene un tamaño fijo por lo que el número de transacciones que contenga dependerá del tamaño de estas. Normalmente un bloque está formado por una cabecera y un cuerpo. Cada implementación puede tener cambios, pero por lo general la cabecera contiene datos como el hash del bloque anterior, el hash del nodo raíz del Merkle tree y un timestamp. El cuerpo por otra parte está formado por un contador de las transacciones que incluye y el listado de dichas transacciones.

Por último, habiendo explicado cada componente queda presentar la unión de todo. A la colección de transacciones y bloques es lo que conocemos como blockchain. En el momento en el que se produce una transacción, esta se transmite a la red para que el resto de los nodos puedan verificarla, validarla y añadirla a un bloque. Cuando se alcance el tamaño requerido el bloque se publicará y el resto de los nodos lo añadirán a su cadena. Este proceso que se acaba de comentar no es precisamente sencillo, ya que se necesitan de mecanismos de consenso para decidir el bloque de qué nodo será el elegido para añadirse a la cadena.

2.5.2. Consenso

El consenso es una de las características más importante de este tipo de redes. Cuando hablamos de consenso nos referimos a los mecanismos que permiten llegar a un acuerdo entre nodos que no tienen por qué conocerse. Esto genera confianza entre los usuarios, ya que el algoritmo de consenso nos asegura que el sistema funcionará correctamente aún en el caso de que existan agentes malintencionados. Aunque existen muchas variantes de algoritmos de consenso, al final, estos tienen que obtener el mismo resultado: determinar qué bloques son válidos para añadirse a la cadena. El consenso suele implicar un proceso de minado, donde para poder generar un bloque válido un nodo deberá resolver un problema difícil de calcular, pero sencillo de validar por los demás nodos de la red. En los siguientes apartados vamos a presentar diferentes algoritmos de consenso para ver cómo funcionan y en qué contexto se utilizan.

Proof-of-work (PoW)

Proof-of-work (o prueba de trabajo en castellano) es uno de los algoritmos más famosos en las blockchain. Cuando Satoshi Nakamoto presentó la primera red blockchain,

decidió utilizar el protocolo de prueba de trabajo y desde entonces han aparecido más redes que también lo implementan. El algoritmo consiste en un proceso de minado en el cual se añade un nonce al bloque y se calcula su hash. Este hash debe comenzar con un número definido de ceros. Cuantos más ceros se definan, más costoso será el cálculo computacionalmente. Esto es necesario para asegurar la seguridad del algoritmo ya que, si se propone un problema más sencillo, podrían aparecer otras soluciones válidas. Varias soluciones válidas pueden generar una separación en la cadena (fork), lo que resulta en diferentes cadenas para cada solución. Sin embargo, el problema de tener varias cadenas se resolverá eventualmente, ya que a medida que se agregan bloques, los nodos siguen la regla de la cadena más larga para determinar cuál es la correcta. En una red con un volumen muy elevado, pueden producirse una gran cantidad de forks. Tener muchas cadenas al mismo tiempo puede dar lugar a un problema de doble gasto. Esto significa que, mientras no se haya elegido cuál cadena es la principal, una moneda digital puede estar duplicada en diferentes cadenas y, por lo tanto, podría gastarse varias veces. En resumen, la seguridad de este algoritmo se basa en la dificultad de encontrar un nonce específico. Existen variantes más eficientes de prueba de trabajo para evitar los forks, pero suelen implicar un gasto computacional demasiado elevado.

Proof-of-Stake (PoS)

Una de las principales críticas a PoW ha sido el elevado costo energético que supone mantener la cantidad de nodos mineros. Para reducir dicho costo, se creó proof-of-stake o prueba de participación. En este sistema, el valor del stake (el saldo de su cuenta) de los mineros se bloquea para que no puedan retirarlo. A medida que el minero agrega más fondos a su nodo, la dificultad del proceso de minado disminuye. Al final, este sistema resulta en un grupo de mineros con un stake muy elevado encargados del consenso de la red. En PoS se presenta un problema llamado "nothing-at-stake" (nada en juego), en el cual los mineros con cuentas vacías pueden conspirar para producir un fork en la red. Para abordar este problema, se han propuesto diferentes soluciones, como obligar a los mineros a comprar o depositar una cantidad de saldo, que se retirará si el algoritmo detecta comportamientos incorrectos. PoS se utiliza en redes como Tendermint o Tezos, y redes como Ethereum ya han realizado la transición a este nuevo algoritmo.

Practical Byzantine Fault Tolerance (PBFT)

A diferencia de los algoritmos presentados antes, PBFT es determinista, lo que significa que cuando se incluye un bloque a la red este es final. El algoritmo tiene tres fases. Primero, en la propuesta, el líder que inicia el proceso transmite al resto de los nodos una solicitud de operación a ejecutar. Durante la preparación, el resto de los nodos comparten los valores que ellos pretenden transmitir. Después de recibir los mensajes de dos tercios de los nodos de la red, cada nodo envía un mensaje de compromiso al resto. Cuando un nodo recibe el mismo mensaje de compromiso de dos tercios de los nodos, este ejecuta la operación y envía un mensaje al nodo que inició el algoritmo. Al final cuando el nodo líder ha recibido los mensajes de dos tercios de la red, puede confirmar que la operación se ha realizado con éxito y que la operación es válida. Como se puede ver, este algoritmo supone varias rondas de comunicación donde cada nodo envía $n - 1$ mensajes donde n es el total de nodos de la red. Este mecanismo escala bastante mal, ya que con un número elevado de nodos la cantidad de mensajes es insostenible. Por este motivo PBFT se suele utilizar en redes donde el número de nodos está definido y por lo tanto se puede limitar el número de mensajes. Algunas variantes de este algoritmo proponen que el voto de algunos nodos tenga más valor en función de algún criterio.

2.5.3. Tipos de blockchain

En este apartado vamos a presentar los diferentes tipos de blockchain y sus características.

- Pública. Bitcoin es el mejor ejemplo y el más famoso para este caso. Cualquier persona con acceso a internet puede acceder a la red. Los participantes de la red son sus nodos, los cuales pueden acceder a todos los bloques registrados, validarlos y participar en los procesos de minado de la red. Los nodos son una pieza de software publicada en internet por lo que cualquier persona puede descargarse una copia y mantener su propio nodo. Los usuarios de la red solo necesitan una wallet con un par de claves para poder firmar transacciones y emitirlas a través de dichos nodos. El uso más común para este tipo de redes es el intercambio de criptomonedas. Las ocasiones en las que la seguridad de la red se ve comprometida suelen venir por un mal uso por parte de los usuarios, normalmente situaciones en las que un usuario pierde el acceso a su wallet y por lo tanto se pierden las criptomonedas que esta contenía. Otro ejemplo puede ser cuando un usuario comparte su clave privada, con esta clave otro usuario podría firmar transacciones por él suplantando su identidad. Este tipo de redes están abiertas y son transparentes para que cualquier usuario pueda validar la información en cualquier momento.
- Privada. Este tipo de redes son restrictivas y están permisionadas. Funcionan como las redes públicas a diferencia de que normalmente el conjunto de nodos es mucho menor y está definido antes del despligue de la red. Normalmente el número de nodos puede ampliarse, pero el nuevo nodo debe ser aprobado por el resto. Al tener el número de nodos de la red controlado este tipo de redes puede optar por mecanismos de consenso diferentes, como PBFT, evitando así el coste computacional del minado. Las redes privadas suelen desplegarse dentro de organizaciones o empresas para casos como votaciones o auditoría de archivos. Algunos ejemplos este tipo de redes son el proyecto de Hyperledger Fabric, o Quorum, una variante de Ethereum donde al no tener minado el costo de emitir transacciones en la red es cero.
- Híbrida. Son una combinación de las redes públicas y privadas. Las características de ambas pueden aparecer en este caso. Podemos tener una red abierta y pública donde cualquiera puede leer y escribir, pero que una organización controle y establezca los nodos. Otro caso puede ser una red donde una parte sea pública para que los usuarios puedan compartir información, y otra parte privada para cuando se necesite que la información sea confidencial para cierto grupo. Las redes híbridas suponen un aumento en el número de hashes a utilizar, así como en la cantidad de verificaciones necesarias.
- Consorcio. En este caso la red es semidescentralizada, lo que significa que más de una organización controla la red blockchain. Es diferente a una red privada, la cual está controlada por una única organización. En un consorcio el conjunto de organizaciones actúa como la autoridad para mantener el minado. Este tipo de redes se utilizan en sectores como la banca u organizaciones gubernamentales.

2.6 Organizaciones relevantes

Primero vamos a presentar a una serie de organizaciones que se pueden considerar como las más influyentes en estos momentos. No entraremos en detalle aún sobre soluciones, ya que alguna de estas organizaciones no se dedica a generar implementaciones.

2.6.1. World Wide Web Consortium (W3C)

El W3C es un comité encargado de generar recomendaciones y estándares para su uso y desarrollo en Internet. Fue fundado en 1994 por el MIT (Instituto Tecnológico de Massachusetts). El objetivo del W3C es unificar las especificaciones de las tecnologías web de forma que se mantengan en línea con la idea de la *World Wide Web*.

El comité se basa en una política de código abierto. Las recomendaciones que proporciona este consorcio se consiguen gracias a la participación de grupos de trabajo de todo el mundo. Alguna de estas recomendaciones puede llegar a tardar años en publicarse, ya que los estándares sufren muchas revisiones e iteraciones. Dichos estándares no son normas ISO, pero la comunidad las suele aceptar como recomendaciones a seguir por la buena reputación del W3C. Algunas de las recomendaciones del W3C se consideran protocolos básicos de Internet, como la arquitectura Cliente-Servidor, el lenguaje de marcado HTML, HTTP o los navegadores web. En el contexto de este TFG los más importantes serían los vistos anteriormente: los identificadores descentralizados y las credenciales verificables.

En el caso de los DID la especificación empezó a escribirse en 2019 y se aceptó como recomendación oficial en 2022 [45], estando aún en su versión 1.0 a fecha de junio de 2023. Esta especificación aún tiene muy poco tiempo, pero se ha escrito de una forma muy abierta para su posible aceptación en el mayor número de contextos. En el caso de las credenciales verificables, estas aparecieron unos años antes, en 2017, y en 2019 [42] se aceptaron como recomendación. Desde que se aceptó ha recibido dos versionas nuevas (ambas llamadas 1.1), una en 2021 y otra en 2022 [43]. A fecha de hoy ya se está generando una versión 2.0 [44] que plantea cambios más importantes respecto a sus anteriores versiones. La gran cantidad de cambios y versiones en tan poco tiempo representa la dificultad de definir la estructura y el uso de este tipo de credenciales.

Una última cosa a tener en cuenta es que el objetivo del W3C con este par de recomendaciones no es definir cómo se desarrollarán las aplicaciones que los usen ni sus interacciones, sino establecer un metasistema a partir del cual los desarrolladores puedan implementar sus propias soluciones [47].

2.6.2. Decentralized Identity Foundation (DIF)

DIF es una organización centrada en el desarrollo de los componentes necesarios para conseguir un ecosistema abierto para la identidad autosoberana y asegurar la interoperabilidad entre todos los agentes. Al igual que W3C, DIF genera recomendaciones y estándares para protocolos emergentes, componentes y formatos de datos, estando más centrado en el aspecto técnico de estos. Sobre estas recomendaciones los miembros de DIF también se encargan de proporcionar implementaciones de referencia [14]. DIF está formado por grupos de trabajo, cada uno especializado en un área de la identidad autosoberana. Su función es investigar y presentar avances para la comunidad en sus respectivas áreas. Algunas de sus aportaciones más importantes son las siguientes:

- El estándar para la comunicación entre agentes "DIDComm Messaging v2" [12] que, aunque inicialmente DIDComm nació en la comunidad de Hyperledger Aries (se presentará en las siguientes secciones), DIF tomó la iniciativa para generar una nueva versión.

- Su aplicación "Universal Resolver" [13], la cual permite resolver un DID en su documento para una gran cantidad de métodos DID diferentes, además la comunidad puede aportar sus propios conectores para soportar más tipos de DID.

- El estándar "Credential Manifest" [6], el cual tiene como objetivo normalizar el proceso de obtención de las credenciales, aunque actualmente se encuentra en una etapa de "borrador aprobado", sin haber llegado todavía a su primera versión estable.
- El estándar "Presentation Exchange" para la gestión de las presentaciones verificables [7]. Se encuentra en su segunda versión, pero ya han iniciado la evolución de esta versión, actualmente se encuentra en una etapa de pre-borrador [8].

A pesar de ser una de las organizaciones líderes y más relevantes en la comunidad, DIF no cuenta con el prestigio del W3C por lo que sus recomendaciones no suelen aceptarse de manera inmediata como una norma a seguir. DIF es una de varias organizaciones que se especializan en generar recomendaciones sobre identidad autosoberana. Algunas veces colaboran entre ellas para generar estándares en común, pero en muchos casos acaban generando diferentes recomendaciones para el mismo componente, dificultando la interoperabiliad.

2.6.3. Internet Identity Workshop (IIW)

El Internet Identity Workshop es un evento que reúne a expertos de la identidad digital para explorar y discutir sobre temas relacionados con la identidad digital. El IIW se ha enfocado en la identidad digital autónoma y descentralizada, y se ha convertido en uno de los eventos más importantes para compartir conocimientos, presentar proyectos y promover el avance de tecnologías y estándares en este campo.

El IIW se lleva a cabo dos veces al año y cuenta con una estructura de participación abierta, donde cualquier persona interesada en la identidad digital puede asistir y contribuir. Durante el evento, se organizan sesiones de trabajo, presentaciones, debates y talleres en los que los participantes pueden compartir sus ideas, investigaciones y experiencias.

Una de las características distintivas del IIW es su enfoque en la colaboración y la participación. Los participantes tienen la oportunidad de proponer temas de discusión, liderar sesiones y formar grupos de trabajo en torno a áreas específicas de interés. Esto fomenta la generación de nuevas ideas, la resolución de problemas y la creación de redes entre profesionales del ámbito de la identidad digital.

2.6.4. Rebooting the web of Trust (RWOT)

RWOT es un proyecto que tiene como objetivo explorar y desarrollar soluciones para mejorar la identidad digital y la privacidad en línea. El proyecto busca abordar los desafíos actuales en torno a la identidad en la web y promover la adopción de tecnologías descentralizadas y autosoberanas.

Su iniciativa reúne a expertos de diversos campos, como criptografía, seguridad, identidad digital y tecnología blockchain. Estos expertos colaboran en talleres y eventos para discutir, investigar y desarrollar propuestas técnicas y estándares que aborden los problemas de identidad en línea. Este año celebrarán la edición número doce de su taller. Durante los meses previos a este evento, los participantes podrán proponer lecturas previas para el resto del grupo. Una de las primeras acciones en el taller es elegir una serie de temas de entre todas las propuestas y formar grupos para buscar soluciones en cada uno de los temas seleccionados.

RWOT se ha convertido en una plataforma importante para la innovación y el avance en el campo de la identidad digital y la privacidad en línea. A través de la colaboración y

el intercambio de conocimientos, el proyecto busca impulsar el desarrollo de soluciones más seguras y confiables para proteger la identidad en el entorno digital.

2.6.5. Hyperledger

Hyperledger es una comunidad de código abierto que se enfoca en el desarrollo de tecnologías blockchain. Fue establecida en 2015 por la Fundación Linux y desde entonces ha reunido a numerosas organizaciones, empresas y desarrolladores que trabajan en conjunto para impulsar la adopción y el avance de soluciones blockchain en el ámbito empresarial.

Hyperledger ofrece varios proyectos y frameworks blockchain que se adaptan a diferentes necesidades y casos de uso. Algunos de los proyectos más destacados incluyen Hyperledger Fabric, Hyperledger Sawtooth, Hyperledger Indy, Hyperledger Iroha y Hyperledger Besu. Cada uno de estos proyectos se enfoca en diferentes aspectos y características de la tecnología blockchain, como la escalabilidad, la privacidad, la identidad descentralizada y la ejecución de contratos inteligentes.

Además, Hyperledger proporciona herramientas, documentación y recursos para facilitar el desarrollo y despliegue de soluciones blockchain empresariales. Esto incluye bibliotecas, APIs, marcos de trabajo de pruebas y otras utilidades que permiten a los desarrolladores crear, probar e implementar aplicaciones basadas en blockchain de manera eficiente y confiable.

2.6.6. OpenID Foundation

La OpenID Foundation es una organización que se dedica a promover y facilitar la adopción de tecnologías de identidad y autenticación. Fundada en 2007, la OpenID Foundation es una de las organizaciones más relevantes en el desarrollo de especificaciones y protocolos relacionados con la identidad digital.

El siguiente párrafo resume brevemente la gestión federada de la identidad que la OpenID Foundation ha estandarizado mediante una amplia colección de protocolos accesibles aquí.

La gestión federada de identidades propuesta por OpenID se basa en el intercambio de información entre un proveedor de identidad y un sitio web que solicita la autenticación. Cuando un usuario intenta iniciar sesión en un sitio web con su identidad OpenID, el sitio web redirige al usuario al proveedor de identidad. Allí, el usuario proporciona sus credenciales de inicio de sesión y el proveedor de identidad verifica la autenticidad de esas credenciales. Una vez verificada la identidad del usuario, el proveedor de identidad envía una respuesta al sitio web, confirmando la autenticación exitosa del usuario. Con esta información, el sitio web puede permitir el acceso al usuario y ofrecer servicios personalizados basados en su identidad [37].

Como se puede ver este flujo se parece mucho al presentado para las credenciales verificables de W3C. La principal diferencia es que durante el flujo de comunicación el usuario no tiene control de sus credenciales, estas están controladas por su proveedor de identidad y este se las envía directamente al verificador sin consultar al sujeto de la identidad. Al hacerlo de esta manera el sujeto no puede controlar qué información sobre él es compartida.

La fundación ha empezado a analizar la gestión de identidad autosoberana y para ello ha iniciado un grupo de trabajo preliminar sobre credenciales verificables: OID4VC. De momento ese grupo está trabajando en cinco borradores de especificación: “OpenID

for Verifiable Credential Issuance", "OpenID for Verifiable Presentations", "Self-Issued OpenID Provider v2", "OpenID for Verifiable Presentations over BLE", "OpenID Connect UserInfo Verifiable Credentials".

OpenID ha sido una solución para la gestión de identidades que ha sido ampliamente utilizada en contextos federados para simplificar la experiencia de los usuarios. Con la salida de la recomendación sobre credenciales verificables de W3C y el cambio a un paradigma de identidad que busca ofrecer más control a sus usuarios, OpenID se ha sumado también generando sus propios estándares en identidad autosoberana [10]. Teniendo ya un flujo de comunicación muy parecido al recomendado por W3C, OpenID solo ha tenido que actualizar el que ya tenían.

2.7 Crítica al estado del arte

En este apartado, presentaremos una serie de organizaciones y proyectos relacionados con la identidad autosoberana. Los elegidos no son los únicos en la comunidad, pero se han seleccionado debido a su relevancia o madurez en el tema. Durante la investigación, han surgido muchos proyectos, pero no tiene sentido mencionarlos todos, ya que al tratarse de un campo tan nuevo y cambiante, muchos de ellos han sido abandonados debido a cambios en las recomendaciones o a nuevas versiones que rompían la interoperabilidad con las anteriores.

2.7.1. Proyectos en SSI

En esta sección se van a presentar las principales soluciones que se han analizado durante la etapa de investigación de este TFG. Los siguientes proyectos se han elegido por diferentes motivos. El consorcio Hyperledger tiene varios proyectos que forman un conjunto, entre ellos el más conocido es Indy por ser uno de los primeros en ofrecer una solución completa de identidad autosoberana. Luego se hablará de Sovrin por su implicación con Indy. También se verá Uport, que es una solución basada en una red pública como es Ethereum. Y por último se analizará la infraestructura europea de servicios blockchain, un proyecto muy reciente desarrollado por la Comisión Europea para introducir las tecnologías blockchain en la administración y así poder digitalizar los procesos.

Hyperledger SSI suite

La fundación Hyperledger ha dedicado desde hace años esfuerzos para el desarrollo de tecnologías que den soporte y permitan el uso de la identidad autosoberana. En este apartado vamos a presentar los proyectos Hyperledger que han surgido con este objetivo.

Indy fue el primer proyecto de Hyperledger dedicado a la identidad digital. Inicialmente su código fue desarrollado por la fundación Sovrin, la cual se presentará más tarde, y fue cedido a Hyperledger. Indy fue aceptado en 2017 como proyecto oficial. La red de Indy se centra en la identidad descentralizada, soportando los conceptos que se han presentado antes: identificadores descentralizados y credenciales verificables, aunque no utiliza las de W3C, sino que desarrollaron las suyas propias. La blockchain de Indy se llama Plenum, está diseñada para que cualquiera pueda interactuar con ella (pública), pero solo una serie de nodos previamente aprobados pueden participar en el proceso de validación de las transacciones (permisionada), estos se conocen como *stewards* (mayordomos en castellano).

Conforme Indy fue creciendo, se llegó a la conclusión de que la criptografía desarrollada para el proyecto podría ser útil en otros. Por ello, la biblioteca que encapsulaba todo el software criptográfico se trasladó a su propio proyecto. Hyperledger Ursa se propuso con el objetivo de proporcionar un espacio a los expertos en criptografía para que fueran los encargados de desarrollar el código correspondiente, permitiendo que otros expertos lo revisaran y lo probaran. Ursa encapsula estas primitivas de criptografía y las ofrece de forma fácil de usar para otros desarrolladores, sin necesidad de que comprendan en profundidad cada elemento. Por lo tanto, permite su utilización tanto dentro como fuera de otros proyectos de Hyperledger.

El SDK de Indy permitía el desarrollo de agentes que podían interactuar entre sí y con la blockchain de Indy. Inicialmente, el desarrollo de estos agentes se realizaba de forma aislada (en silos), cada proyecto creaba el suyo y solo podían interactuar dentro de su propio entorno. Con el objetivo de abordar este problema de interoperabilidad se presentó Hyperledger Aries. El cual tiene como objetivo el desarrollo de un agente capaz de interactuar con cualquier otro a través de unos mecanismos estandarizados. Actualmente muchos de los conceptos desarrollados en Indy se han trasladado a Aries, pero realizando cambios para soportar el uso de cualquier tipo de ledger.

Figura 2.6: Hyperledger SSI suite.

Hyperledger Aries no es un proyecto que se base en una solución de código. Consiste en una serie de estándares que sirven como guía para los desarrolladores que deseen implementar sus soluciones dentro del ecosistema de Aries. Su repositorio contiene una gran cantidad de especificaciones que estandarizan diferentes aspectos de la identidad autosoberana. Estas especificaciones se conocen como Request for Comment (RFC). Estos documentos no especifican cómo se deben implementar dichas funcionalidades, sino cómo se deben manejar. No obstante, al final de cada RFC existe una sección para referenciar posibles implementaciones que la comunidad haya desarrollado del estándar.

Aries es un proyecto de código abierto, al igual que el resto de los proyectos de Hyperledger, por lo tanto, cualquier persona puede crear los estándares que vayan apareciendo. Para controlar qué estándares son válidos y cuáles aún necesitan más maduración, estos pasan por un ciclo de vida. Como se ha mencionado, cualquier persona puede proponer un nuevo estándar. Para avanzar al estado de "demostrado", la especificación debe contener al menos un par de implementaciones que demuestren su funcionalidad. En este estado, el estándar se considera aún en progreso, ya que todavía no ha sido aceptado por la comunidad. Para lograr esto, se debe llegar a un consenso entre los demás participantes, presentando la idea a través de chat o en las reuniones de la comunidad. Además,

se debe demostrar su funcionamiento proporcionando los resultados de los tests necesarios. Finalmente, un estándar propuesto se considerará adoptado si se implementa en proyectos relevantes y se socializa con el resto de RFC, lo que significa que se referencia y se utiliza en otros estándares de la comunidad.

Actualmente, la comunidad ha adoptado un RFC y ha aceptado otros 41, como se puede ver en una lista de RFC disponible en el repositorio. La mayoría de las especificaciones aceptadas son estándares en su primera versión, las cuales suelen contar con algunas implementaciones. Sin embargo, también podemos encontrar segundas versiones de estándares que han sido aceptados, aunque aún no cuenten con implementaciones que demuestren su funcionamiento.

Es muy difícil que aparezca una solución que se utilice por todo el mundo y en todos los contextos. Aunque Indy fue una de las primeras soluciones de identidad autosoberana y bastante completa, lo más probable era que con el tiempo surgieran más implementaciones. El objetivo de Aries es estandarizar todos los procesos relacionados con las credenciales verificables, permitiendo así la comunicación entre agentes desarrollados por diferentes organizaciones. Esto facilitaría la creación de un ecosistema donde agentes con identificadores registrados en blokchain diferentes, pudieran intercambiarse mensajes a través de un canal estandarizado, que también permitiera el uso de diferentes tipos de credenciales verificables.

Sovrin

Sovrin es una organización sin ánimo de lucro con alcance global que organizó el despliegue de Hyperledger Indy en una serie de nodos para crear una instancia pública. Esta red mantenida por Sovrin lleva activa desde 2017. La fundación Sovrin define tres componentes fundamentales para la red, a los cuales hacen referencia como BLT:

- Negocio (Business), un marco de gobernanza legal establecido por la fundación.
- Legal, una serie de acuerdos legales firmados por los participantes de la red
- Técnico, el software que da soporte por debajo a la red de Sovrin, Hyperledger Indy.

Los marcos de gobernanza son muy comunes en el mundo digital, pero también suelen aparecer en otros contextos, normalmente con nombres parecidos como políticas o regulaciones operativas. En todos los casos estas políticas definen los mecanismos disponibles para que diferentes participantes puedan realizar transacciones con cualquier otro participante de una manera consistente y predecible. Por poner un ejemplo, las tarjetas de crédito operan dentro de un marco de gobernanza. Los participantes del marco, usuarios de tarjetas de crédito, negocios, bancos y empresas de crédito operan independientemente, pero de acuerdo a unas reglas establecidas. Por este motivo un negocio puede aceptar el pago de un producto a través de la tarjeta de crédito de un cliente y estar seguro de que el resto de los participantes harán su parte del trabajo y enviarán los fondos correspondientes a su cuenta. No existe una solución de software que se encargue de ejecutar todo el proceso, en cambio, cada participante ha desarrollado el código necesario para realizar su parte.

Ya hemos definido que la red de Sovrin consiste en una única instancia global de Hyperledger Indy. Cada nodo está controlado por una organización participante que ha aceptado los acuerdos que definen cómo debe operar el nodo. Esto hace referencia al hardware necesario que deberá tener, la conexión a internet, su monitorización o el mantenimiento requerido. Dentro de las organizaciones que hoy en día se encargan de mantener nodos se encuentran bancos, universidades y empresas tecnológicas de todo el

mundo. La fundación Sovrin también ha formado consejos encargados de supervisar los aspectos legales y tecnológicos de la red.

Con esta red global desplegada, Sovrin pretende solucionar el problema de la identidad en Internet. Los usuarios podrán crear y registrar sus propios identificadores descentralizados y las organizaciones correspondientes podrán emitir las credenciales verificables necesarias. La fundación Sovrin cobra una cierta cantidad por cada transacción emitida en la red. El coste de estas transacciones es muy reducido, argumentan que será necesario para evitar un uso abusivo y los fondos serán utilizados para mantener la red operativa. Además, como en Sovrin los emisores de credenciales son los únicos que necesitan escribir en la blockchain, serán los encargados de financiar el coste de la red.

Alastria

Alastria es una asociación sin ánimo de lucro compuesta por administraciones públicas, como el Gobierno de España, empresas privadas como Telefónica, Iberdrola, Deloitte y varias universidades españolas. Estas entidades colaboran entre sí con el propósito de fomentar la economía digital mediante la creación de un nuevo ecosistema que facilite el desarrollo de sistemas descentralizados basados en la tecnología blockchain. Esta asociación promueve la colaboración entre sus miembros para acelerar el establecimiento de una infraestructura común que sirva como base para las aplicaciones del futuro. Con este fin, propone un novedoso ecosistema compuesto por cuatro redes distintas: la red T, la red B, la red H y la recién lanzada red H+.

AlastriaID [1] es el proyecto de la comisión de identidad de Alastria. La propuesta para la identidad digital en blockchain tiene como objetivo ofrecer una infraestructura y un marco de desarrollo para llevar a cabo proyectos de identidad autosoberana, con plena validez legal en la zona euro.

Sobre el modelo de AlastriaID también se ha generado un método DID [2], el cual describe el esquema y el Documento DID de Alastria. También incluye el ciclo de vida de las credenciales y presentaciones en la red actual de Alastria T, basada en GoQuorum.

Uport

El proyecto de Uport nació en 2015 por parte de la empresa Consensys. Primero se propuso como una cartera digital en la web y un sistema de gestión de identidades. Uport fue evolucionando mucho durante los años y ha ido experimentando muchos cambios en su arquitectura. Pero la idea principal se ha mantenido, conseguir un sistema de identidad autosoberana utilizando Ethereum como registro de datos público.

La arquitectura de Uport consiste en tres elementos principales: los contratos inteligentes, librerías de desarrollo y su aplicación móvil [27]. La aplicación se encarga de guardar las claves de los usuarios. Los contratos inteligentes son el componente central del proyecto de identidad autosoberana y permiten funcionalidades como la recuperación de la identidad ante pérdida del dispositivo. Las librerías de desarrollo es lo que permitiría a los desarrolladores incorporar Uport en sus proyectos. En 2019 Uport publicó una nueva arquitectura, siendo uno de los pioneros en implementar las propuestas del W3C sobre los identificadores descentralizados y las credenciales verificables. Desde entonces Uport ha participado en el desarrollo de librerías de código abierto que se han convertido en pilares de para la comunidad como did-resolver o did-jwt. Con la aparición de los DID también surgieron los métodos DID y Uport hizo su propia propuesta: did:ethr.

El método de Uport se ha convertido en el principal de Ethereum y en unos de los más apoyados por la comunidad. En este método cada dirección de cuenta en la red de Ethereum es un posible DID, asegurando así la unicidad de los identificadores. Su funcionamiento se basa en el smart-contract ethr-did-registry. Éste estaba pensado para su uso en las redes principales de Ethereum: mainnet, Goerli o alguna testnet. Pero también se puede utilizar en redes que sean compatibles con la Ethereum Virtual Machine (EVM). El contrato permite la autenticación de clave pública, así como la rotación de claves y añadir delegados que puedan emitir transacciones con tu permiso. También facilita la actualización de información de atributos off-chain como pueden ser los endpoints a servicios del sujeto del DID. Estas interacciones con el contrato se complementan con la librería ethr-did-resolver, desarrollada por la Decentralized Identity Foundation, a partir de su predecesora de Uport.

Actualmente lo que era el proyecto de Uport está dividido en otros dos: Veramo y Serto. Para entender las diferencias entre ellos vamos a explicarlos. Uport siempre ha participado mucho en la comunidad open-source aportanto librerías y participando en otros proyectos. En esta trayectoria, se presenta Veramo. El objetivo de este proyecto es crear una arquitectura modular y flexible que se base en una librería central que pueda utilizarse en cualquier escenario y permita ampliar funcionalidades a partir de paquetes añadibles. Esta permitiría a los desarrolladores implementar las necesidades específicas de sus proyectos sin tener que modificar la librería central. Algunos escenarios pueden ser: realizar el soporte para un nuevo método DID, añadir la interoperabilidad con más protocolos o personalizar el manejo de las claves privadas. La librería principal de Veramo consiste en una serie de interfaces abstractas sobre las cuales los desarrolladores pueden trabajar. Además, el equipo de Veramo desarrolló una implementación para cada una de esas interfaces, proporcionando así una referencia de cómo utilizarlas. Actualmente el proyecto soporta varios métodos DID sin necesidad de un desarrollo extra, solo hay que añadir el paquete correspondiente. Los métodos disponibles son: *did:ethr*, *did:web* y *did:key*.

A diferencia de Veramo, Serto se anuncia como una solución completa de manejo de identidad autosoberana. El proyecto ofrece tres productos diferentes. Un agente que *out-of-the-box* permite generar y administrar DID y credenciales verificables de una forma rápida y sencilla sin necesidad de codificar, así como un explorador para descubir emisores de credenciales registrados en la red y poder comprobar cómo de confiables son.

EBSI

El objetivo de la Infraestructura Europea de Servicios Blockchain (EBSI) es habilitar y facilitar el uso de este tipo de servicios para los ciudadanos y las empresas europeas. EBSI está dirigido por la Comisión Europea y la *European Blockchain Partnership* (EBP). Este proyecto pretende mejorar los servicios transfronterizos existentes para así favorecer la movilidad ciudadana y empresarial, cumpliendo con toda la normativa europea actual. El diseño de la infraestructura se guía por los siguientes principios:

- La red blockchain utilizada tiene que ser pública y permisionada. Esto significa que tiene que estar disponible para cualquier persona o entidad que necesite utilizarla, tanto para lecturas como para escrituras. Además, los nodos estarán definidos e identificados y será necesario su previa aprobación para añadirlos a la red. La elección de una red permisionada viene de que la Comisión Europea también busca alcanzar cierto grado de sostenibilidad medioambiental por lo que al definir la lista de nodos no harían falta los costosos procesos de minado actuales de las redes públicas.

- La red tiene que ser escalable. Ahora mismo hay alrededor de unos 30 nodos y no demasiadas transacciones, ya que todavía está en proceso de implantación, pero con el tiempo la red tiene que ser capaz de soportar el futuro escenario donde los millones de ciudadanos y empresas de la Unión Europea la utilicen en su día a día.

- Al estar la infraestructura dirigida por una organización central hay que buscar mecanismos para conseguir establecer confianza en la red, para asegurar que la organización encargada de controlar la red no haga usos maliciosos, esto se consigue empleando tecnologías de código abierto. Estas tecnologías permiten que cualquiera tenga acceso a su código fuente y por lo tanto suelen estar analizadas y revisadas por la comunidad, asegurando así su validez y fiabilidad.

Aparte de los principios mencionados, el más importante que persigue EBSI, es la interoperabilidad. Al final esta infraestructura que está en desarrollo tiene que poder permitir servicios transfronterizos para organizaciones y empresas a través de la digitalización y descentralización de la administración gracias al blockchain. Para poder conseguirlo la comisión encargada ha publicado un kit de despliegue de nodos para facilitar el proceso y ha establecido una serie de APIs definidas en su web para permitir la integración de los servicios actuales con EBSI. Observando los casos de uso que tiene aprobados actualmente se pueden apreciar mejor todos los principios comentados. El caso de uso más relacionado con este TFG es su propuesta de identidad digital autosoberana (European Self-Sovereign Identity, ESSIF). Esta permitiría a los ciudadanos gestionar sus propias identidades sin depender de la entidad centralizada de su lugar de residencia, buscando así la naturaleza transfronteriza mencionada anteriormente. Este caso de uso es transversal con el resto y necesario para el uso de los servicios, ya que sin una identidad digital que represente al sujeto no se podrán recibir las credenciales necesarias para usar los otros servicios disponibles en EBSI. Algunos de los otros casos de uso incluyen la gestión digital a través de credenciales verificables de diplomas en educación, seguridad social o certificados de residencia.

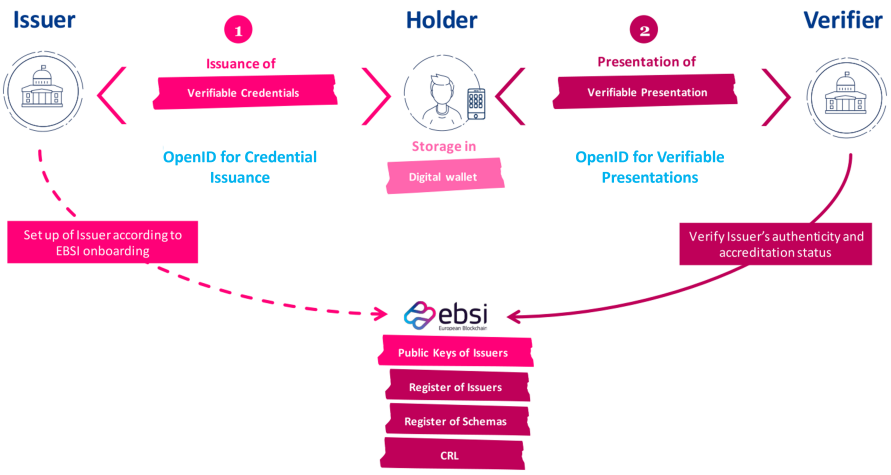

Figura 2.7: Visión general del ecosistema EBSI.

Respecto a las tecnologías que implementa para permitir la emisión y verificación de información, EBSI utiliza las recomendaciones del W3C sobre identificadores descentralizados, credenciales y presentaciones verificables, y también estándares en identidad como OpenID. EBSI también cumple con las regulaciones sobre identidad de la Unión Europea, GDPR y eIDAS. EBSI propone su propio método DID para el manejo de estos identificadores. En este caso EBSI plantea una diferencia entre los identificadores para

personas naturales y para entidades legales estableciendo un método diferente para cada uno [15].

- Los DID para las entidades son identificadores públicos por lo que deben disponer de un DID y su documento registrados y disponibles en la red. La unicidad de los identificadores se consigue a través los smart contracts de la blockchain.
- Los identificadores para las personas buscan mantener la privacidad de estas por lo tanto ningún DID, documento o claves públicas deben registrarse en la red. Todos estos elementos se manejarán a través de la cartera digital de cada ciudadano, que podrá decidir generar, actualizar o revocar sus DID. Ya que estos identificadores no se registran en la red y por lo tanto no se puede comprobar si ya existe uno con ese valor, la unicidad de los DID para personas se consigue a través del uso de una función hash sobre la clave pública con la que se genera el DID.

La elección de no registrar los DID de las personas en la red hace que EBSI no cumpla del todo la recomendación del W3C, ya que no se podría resolver uno de sus DID en un documento para obtener sus claves, pero hay que tener en cuenta que debe cumplir con todas las normativas y reglamento que impone la Unión Europea por lo que será más difícil que cumpla todos los objetivos de la identidad autosoberana.

2.7.2. Trabajos relacionados

Para poder valorar mejor este TFG, se van a presentar una serie de trabajos que parecen tener unos objetivos similares o que también tratan sobre la identidad autosoberana. En primer lugar, se analizarán algunos trabajos de fin de grado para poder comparar el trabajo realizado, y luego se presentarán algunos trabajos finales de máster que se han encontrado bastante parecidos a este TFG.

Trabajos de fin de grado

Los TFG encontrados presentan una similitud notable. A pesar de tener como objetivo y título el prototipado de un sistema de gestión de identidad autosoberana, en realidad, se limitan a integrar algún sistema ya desarrollado, como Indy o AlastriaID, con una aplicación web [38, 17]. Además, en muchos casos, la elección de la solución a utilizar está pobremente argumentada, lo cual suele deberse a un escaso estado del arte que, en ocasiones, ni siquiera menciona las recomendaciones de W3C [35]. Estas especificaciones deberían ser esenciales en proyectos relacionados con DID y credenciales verificables. Se podría argumentar que algunos de estos TFG se realizaron antes de que se publicaran las versiones estables de las recomendaciones en julio de 2022, pero sus ideas y borradores ya estaban en desarrollo desde 2017-2019.

Es importante destacar que todos los TFG citados en el párrafo anterior pertenecen a otras universidades. El único trabajo relacionado encontrado en la UPV es el de George Loladze en 2020, que presenta un sistema de notarización de documentos en Ethereum [26]. En la sección de trabajo futuro de dicho trabajo, se plantea el uso de los DID de W3C para la identificación de los usuarios del sistema.

Trabajos de fin de máster

Por otro lado, los Trabajos Fin de Máster (TFM) encontrados sí plantean un trabajo más completo y argumentado. El primer trabajo encontrado es “Identidad Descentralizada Aplicada” por María Ruiz [36]. En este, se aborda un estudio más profundo sobre

las diferentes propuestas de modelos de datos para las credenciales y los protocolos de comunicación. Luego, presenta un demostrador de los tres agentes de W3C, que consiste en el despliegue de uno de los agentes de Aries desarrollados por la comunidad sobre la red de Hyperledger Indy.

El otro trabajo que comentar es el desarrollado por Guillermo Sanz [39], en el cual realiza un trabajo prácticamente igual que el anterior, pero especificando el demostrador para un caso de uso basado en los pasaportes de vacunación COVID definidos por la Comisión Europea.

2.8 Propuesta

Después de haber analizado y explicado las soluciones de identidad autosoberana más relevantes y otros elementos también importantes ya podemos empezar a plantear el objetivo de este trabajo. Este TFG tiene una visión tanto investigadora como aplicada y su principal objetivo es aprender sobre nuevos conceptos que están ahora en alza como lo es la identidad autosoberana, la criptografía necesaria, además de la infraestructura que más posiblemente habilite su implantación a nivel mundial, el blockchain.

Para poder entender todos estos conceptos al completo se desarrollará un prototipo de gestión de identidades siguiendo el paradigma autosoberano. Para esto habrá que trabajar con los recientes identificadores descentralizados, eligiendo un método y red blockchain sobre los que trabajar. Para poder hacer un prototipo completo también se debería soportar el uso de credenciales verificables para poder soportar los escenarios de emisión y verificación de estas. A través de este desarrollo se aprenderá cómo trabajar con las redes blockchain, así como los problemas que presentan por su complejidad, también se pondrán en práctica conceptos de criptografía vistos durante el grado además de otros que se ven como teoría sobre la gestión de identidades.

El prototipo por desarrollar obligará a implantar y utilizar algún protocolo de interacción entre los agentes, tanto a la hora de generar credenciales verificables como para presentarlas y verificarlas. Esos protocolos no se concretan en las especificaciones del W3C por lo que ha habido diferentes propuestas, que hemos presentado en este capítulo: Aries, DIF y OpenID. Deberemos analizarlas cuidadosamente para elegir una de ellas y usarla en nuestro sistema.

CAPÍTULO 3
Análisis del problema

En este capítulo vamos a discutir unos de los mayores problemas que hay en la comunidad de la identidad autosoberana, el problema de la interoperabilidad. Cuando presentamos el concepto de identidad autosoberana y sus diez principios ya mencionamos la interoperabilidad. Para recordarlo un poco, esta era el requisito por el cual las identidades deben ser ampliamente utilizables en la mayor cantidad de contextos posible. Desde un punto de vista más técnico, dicha identidad podría estar formada por un conjunto de identificadores descentralizados y credenciales verificables que su controlador utilizaría para interactuar con otros agentes.

Entonces podemos entender el problema de interoperabilidad como conseguir que nuestros identificadores y credenciales puedan utilizarse en cualquier plataforma. Esto significa que se tiene que conseguir la estandarización de los procesos, lo cual requiere un esfuerzo internacional para llegar a un acuerdo e intentar generar protocolos que permitan trabajar con distintos tipos de identificadores y credenciales. En este aspecto el grupo de trabajo encargado de la interoperabilidad de DIF generó un tutorial donde presentan los diferentes aspectos y problemas a tener en cuenta [49].

Ahora vamos a realizar un análisis de los diferentes componentes necesarios para establecer un sistema de gestión de identidad autosoberana. Para cada componente se compararán las opciones disponibles para su desarrollo. Algunas de estas opciones son projectos que no permiten realizar ninguna implementación relevante, ya que o se anuncian con soluciones completas ya implementadas o todavía están en etapas muy tempranas de su desarrollo. La opción que se elija debería permitir cierta libertad para realizar una implementación que sirva para representar el trabajo realizado en este TFG. Realizar una solución de identidad autosoberana desde cero conlleva una cantidad de trabajo muy por encima de la carga de este proyecto ya que habría que tener en cuenta muchos aspectos que ahora veremos. Por este motivo se valorarán mejor los proyectos que faciliten el desarrollo de una implementación.

3.1 Identificadores descentralizados

Primero vamos a analizar la situación de los DID. En estos momentos existen más de 100 métodos DID diferentes [46]. Como se ha visto los métodos especifican los mecanismos para manejar sus respectivos DID. Dichos mecanismos pueden ser muy diferentes entre sí. Es muy difícil que aparezca un método que se convierta en el universal por lo que los agentes deberán ser capaces de trabajar con todos los tipos posibles. Para poder soportar los diferentes tipos de DID que vayan apareciendo, el software del agente que utilicemos debe ser fácilmente extensible.

La resolución de DID en su documento también es otro elemento que plantea ciertos problemas. Muchos DID utilizan una blockchain como registro de datos verificable donde almacenar los DID, claves y servicios públicos. Esto supone que si nuestro agente pretende soportar uno de estos DID debe ser capaz de acceder a la blockchain correspondiente, tanto para leer como para poder emitir transacciones. Este tipo de DID públicos serán normalmente utilizados por entidades legales que necesiten simplificar el acceso de sus datos al mayor número posible de usuarios.

En el caso de las personas existen legislaciones, como la GDPR [16], que exigen asegurar cierta privacidad sobre sus datos. Por este motivo es más conveniente emplear DID efímeros. Este tipo de DID no se registran en la blockchain, cada usuario se encarga de almacenar los suyos en sus dispositivos. Estos DID se conocen como *did:peer* y se generan por cada relación que desee crear el usuario. Si estos identificadores se utilizan para interacciones sencillas pueden ser efímeros y eliminarse una vez finalizada esta.

3.2 Comunicación entre agentes

En este ecosistema los agentes tendrán los roles presentados de emisor, propietario y verificador. Para poder cumplir sus funciones estos deben ser capaces de comunicarse entre ellos. Esta comunicación se hará a través de mensajes que puedan contener tanto credenciales como sus presentaciones y que a su vez puedan ser firmados y encriptados.

En el capítulo anterior se ha presentado DIDComm Messaging. Este estándar generado por DIF es actualmente uno de los más utilizados y soportados. Actualmente presenta dos versiones incompatibles entre sí. La segunda versión se encuentra ya desarrollada y en proceso de revisión, aunque la mayoría de los proyectos se encuentran aún en la primera. Además, OpenID Connect (OIDC) ha presentado su propio formato para los mensajes basados en DID, el cual también es incompatible con los de DIDComm. Para poder conseguir interoperabilidad entre los diferentes formatos estos se deben soportar entre sí y los agentes que los utilicen deben ser capaces de reconocerlos y utilizarlos.

Aparte del propio formato de los mensajes también hay que tener en cuenta el protocolo de transporte. Si un agente A solo puede recibir mensajes por HTTP, el resto debe ser capaz de comunicarse con él. Pero la comunicación DID no está especificada que deba darse a través de un protocolo o tecnología en específico, esta se podría hacer a través de Bluetooth, NFC, QR o incluso protocolos que aún no se hayan desarrollado.

3.3 Credenciales verificables

Como hemos visto los identificadores descentralizados y la comunicación presentan problemas debido a la existencia de una diversidad de protocolos y estándares posibles. Las credenciales en este sentido no son diferentes, pero como añadido presentan aún más componentes con diferentes estandarizaciones posibles, lo cual significa tener muchos vectores de fallo posibles. Una simple diferencia en el tipo de firma puede hacer que un agente ya no sea capaz de verificar una credencial, incluso aunque el resto de los elementos sean comunes para ambos.

En el capítulo del estado del arte se ha presentado el modelo de datos de las credenciales verificables recomendado por W3C. Esta estandarización de las credenciales no es la primera en aparecer ya que Indy usaba su propio tipo de credenciales mucho antes de que W3C comenzara a generar su recomendación. Además del primer formato de Indy, también se presentó una evolución de este con las credenciales *AnonCreds*, las cuales buscan conseguir la máxima privacidad del usuario empleando predicados y pruebas

de conocimiento cero [11]. OIDC no utiliza credenciales verificables como las de W3C, sino que en su flujo de comunicación utiliza unos tokens basados en JWT [22]. Estos contienen una cabecera con metadatos, un cuerpo con información sobre la identidad del propietario y una firma para poder verificar la información.

Para poder verificar una credencial es necesario añadir algún tipo de firma digital. Al igual que pasa con los protocolos y los formatos, los agentes encargados de verificar deben soportar el tipo de firma que recibe para poder realizar el proceso.

3.4 Registro de datos verificable

El registro de datos verificable es el elemento encargado de almacenar los documentos DID, por una parte, y los esquemas de las credenciales verificables por otra. Este debería ser algún tipo de registro distribuido y persistente. Las soluciones habituales lo implantan como una blockchain, pero eso no tiene por qué ser forzosamente así. Hasta ahora ninguna de las especificaciones de mayor difusión ha llegado a una versión estable que determine cuál debe ser el esquema a utilizar en las credenciales verificables, así como el formato en el que se deben registrar o los mecanismos a través de los cuales se realizará este proceso. Si se opta por utilizar una blockchain para guardarlos, habría que analizar con calma cómo descartar los esquemas mantenidos en ella que no cumplan con la especificación que finalmente se acepte como definitiva (y la probabilidad de que eso suceda será alta).

Por ello, en nuestro prototipo, la gestión de los esquemas de credenciales se emulará mediante SQLite. En un futuro, cuando algún organismo genere una especificación estable y ampliamente aceptada para los esquemas de credenciales, reemplazaremos esa emulación por una implementación en la blockchain seleccionada.

En la sección del estado del arte se ha presentado la tecnología blockchain y su estrecha relación con la identidad autosoberana. Por esos motivos el registro de datos verificable que se utilizará para almacenar los documentos DID será una red blockchain. El siguiente punto será decidir cuál de todas las redes utilizar. Debido al elevado número de redes diferentes, las opciones planteadas se han elegido por la experiencia y familiaridad que se tenía con estas. Acelerando así el tiempo de aprendizaje necesario y recortando costes de tiempo en el caso de tener que aprender el funcionamiento de una blockchain nueva.

3.4.1. Redes público-permisionadas

Muchas soluciones actuales de gestión de identidades implementan blockchain público permisionadas. Los ejemplos más importantes son los analizados en la sección del estado del arte: Indy, Sovrin y Alastria. La elección de una red permisionada sobre una *permisionless* es para acercarse a las ventajas de una red pública donde todos los usuarios pueden leer y escribir en la cadena de bloques pero manteniendo el número de nodos controlados. Gracias a esto se pueden utilizar otros algoritmos de consenso, evitando así los elevados costes de transacción que generan los procesos de minado. Aunque no se tengan que pagar costes de transacción como en una red pública, las organizaciones que se encargan de mantener estas redes permisionadas ponen precio a cada tipo de transacción posible como compensación por mantener los nodos de la red. Además, estos costes se suelen aplicar solo a las organizaciones emisoras de credenciales y no a los usuarios normales. Para el desarrollo de este TFG queremos evitar tener que dedicar presupuesto para poder trabajar sobre una red blockchain y hacer todas las pruebas que se necesiten.

Por lo que, aunque algunos proyectos ofrezcan redes de prueba a un menor coste, sigue sin interesarnos.

Otro punto a tener en cuenta es que los proyectos mencionados en el párrafo anterior no son herramientas o *frameworks* sobre los que desarrollar. Sovrin y Alastria se anuncian como soluciones completas de identidad autosoberana, por lo que la única opción para un desarrollo sería llevar a cabo un despliegue de ese sistema, tal como han hecho algunos de los otros TFG mencionados. Indy se llegó a publicar como solución independiente pero actualmente se ha reducido a solo la red especializada en identificadores descentralizados.

3.4.2. Hyperledger Fabric

Primero vamos a recordar en qué consistía Fabric. La blockchain es de tipo privada y está pensada para entornos empresariales donde un conjunto de empresas con un objetivo común forma un consorcio para compartir información. La autenticación en Fabric se controla a partir de certificados X.509 generados por cada empresa para sus usuarios y administradores. En secciones anteriores de este documento ya se ha explicado que los certificados no cumplen con los requisitos de la identidad autosoberana, por lo que habría que buscar una manera de trabajar sobre estos sin depender de ellos.

Para poder emitir transacciones a la red se deberán firmar con un certificado válido por lo que cada usuario debería registrarse en una empresa para obtener su certificado que representaría su identidad. Para evitar esto se podría plantear un diseño en el cual se publique una API con un certificado de usuario capaz de emitir transacciones. Los usuarios generarían sus DID localmente y las transacciones para actualizar su documento se firmarían todas con el certificado de la API. Esta idea plantea ciertos inconvenientes. Ahora las empresas aparte de tener que mantener un nodo de la red también deberán publicar un endpoint para que los usuarios pueden emitir transacciones. Esta funcionalidad es igual que tener una blockchain público permisionada, pero teniendo que desplegar más elementos. Otro inconveniente es que cada organización controlará su propio *endpoint* pudiendo hacer usos maliciosos, como bloquear las transacciones de ciertos usuarios.

Una ventaja que presenta Fabric es su capacidad como base de datos clave valor. La mayoría de las blockchain funcionan solo como cadena de bloques y para formar el documento DID hay que recorrer los bloques que contienen transacciones de actualización e ir aplicando los cambios sobre un documento predeterminado. En Fabric este proceso se simplifica mucho ya que aparte de la cadena de bloques también hay una base de datos clave valor con el estado actualizado de la blockchain, pudiendo obtener directamente la última versión del documento DID.

3.4.3. Ethereum

Los defensores más estrictos de la identidad autosoberana argumentan que con las redes público-permisionadas no se puede conseguir un verdadero sistema autosoberano y descentralizado, ya que la red sobre que la que se implementa sigue estando bajo el control de alguna organización central. Utilizar una red pública significaría tener que pagar costes de transacción para cada actualización que se haga en el documento DID. Desde hace tiempo Ethereum ha cogido mucha fama por sus altísimos costes de transacción. Por lo tanto, para poder desarrollar y hacer pruebas surgieron las "testnet", cadenas de bloques paralelas a la "mainnet" sobre las que poder hacer pruebas pagando unos precios muchos menores. Esta solución sigue suponiendo un aumento en el presupuesto del proyecto que se podría evitar. El método "did:ethr" está diseñado para poder funcionar

sobre la red principal de Ethereum y con cualquier otra red compatible con la "Ethereum Virtual Maquine" (EVM). Por lo que el comportamiento de la solución a desarrollar será el mismo tanto en una red pública como en una privada.

3.5 Solución propuesta

Una de las decisiones más importantes que se deberá tomar respecto a nuestra propuesta es qué protocolos para la emisión y presentación de credenciales se decidirán implementar. Actualmente, las organizaciones que han realizado esfuerzos para la estandarización de estos procesos son DIF, Aries y OpenID. En las siguientes tablas (Tabla 3.1 y Tabla 3.2), se presentará una comparativa realizada entre las propuestas de cada una de estas organizaciones.

Especificación	Estado	Implementaciones
DIF Credential Manifest	Borrador Aprobado	0
Aries RFC 0037: Present Proof Protocol 1.0	Aceptado	1
Aries RFC 0454: Present Proof Protocol 2.0	Aceptado	0
OpenID for Verifiable Credential Issuance	Borrador Publicado	3
OpenID for Verifiable Credential Issuance - draft 13	Borrador en construcción	0

Tabla 3.1: Comparativa de especificaciones sobre la emisión de credenciales.

Especificación	Estado	Implementaciones
DIF Presentation Exchange v1.0	Aprobado	1
DIF Presentation Exchange v2.0	Aprobado	0
DIF Presentation Exchange v2.X.X	Pre-borrador	0
Aries RFC 0036: Issue Credential Protocol 1.0	Aceptado	1
Aries RFC 0453: Issue Credential Protocol 2.0	Aceptado	0
OpenID for Verifiable Presentations	Borrador Publicado	2
OpenID for Verifiable Presentations - draft 20	Borrador en construcción	0

Tabla 3.2: Comparativa de especificaciones sobre la presentación de credenciales.

En ambas tablas se puede observar que la mayoría de las especificaciones no tienen ni siquiera una implementación de referencia. En el caso de OpenID, sus especificaciones son las que presentan más implementaciones, pero están basadas en borradores que, hoy por hoy siguen en desarrollo. En cambio, DIF y Aries presentan especificaciones en versiones estables, por lo que son opciones más seguras sobre las que realizar una implementación. De las dos organizaciones, DIF presenta una especificación muy avanzada sobre la presentación de credenciales que ya va por su segunda versión y sobre la que se está trabajando para continuar. Por el contrario, su especificación sobre la emisión de credenciales aún se encuentra en estado de borrador.

Por este motivo, se ha decidido realizar la implementación de los dos estándares de Hyperledger Aries que buscan estandarizar dichos procesos. Ambos RFC presentan una

segunda versión también aceptada, por lo que nos quedaremos con las versiones 2.0 para el sistema a desarrollar. Además, al no presentar implementaciones, el trabajo realizado tendrá más valor para la comunidad.

Al implementar estos estándares de comunicación junto con un sistema para la gestión de claves privadas y DIDs de los usuarios, se busca crear un prototipo de un sistema de gestión de identidades basado en identidad autosoberana. Como hemos visto, estos sistemas requieren de un registro de datos verificables, para lo cual utilizaremos una red blockchain. El objetivo de este prototipo es demostrar el funcionamiento de este tipo de sistemas y los mecanismos de comunicación que deben estar presentes. Para lograr esto, se debe realizar un diseño que garantice la seguridad en todas las partes de la arquitectura. Por último, este trabajo también contribuirá a una mejor comprensión de las redes blockchain y mostrará que son adecuadas para respaldar los sistemas de identidad autosoberana.

3.6 Plan de trabajo

En esta sección vamos a presentar las tareas que se han llevado a cabo para poder completar el trabajo. Para la realización de este TFG va a ser necesaria una importante tarea previa de investigación y aprendizaje. Primero habrá que leer sobre qué es la identidad, cómo se ha estado manejando hasta el momento y qué problemas presenta. Una vez entendido el problema que se presenta hay que ver qué ventajas plantea la identidad autosoberana. Luego se deberá entender en profundidad las recomendaciones sobre los identificadores descentralizados y las credenciales verificables. Como última parte de la investigación realizada habrá que analizar los proyectos existentes, tanto soluciones como proyectos de estandarización. Además, las blockchain son un tipo de redes complejas que no se han visto durante la carrera por lo que se deberá dedicar esfuerzo a aprenderlas desde cero. Será necesario tanto entender su funcionamiento como ser capaz de trabajar y desarrollar sobre este tipo de redes.

Una vez realizado el trabajo previo comentado arriba ya podremos empezar a tomar decisiones sobre el problema y el desarrollo que queremos plantear. Esto significa elegir qué estándares y tecnologías vamos a utilizar para nuestra solución. Como se ha explicado en los capítulos anteriores estas decisiones no son una tarea sencilla. La investigación realizada nos deja con bastantes estándares posibles a implementar, así como cantidad de redes blockchain y tipos de DID (es decir, métodos DID) posibles.

Habiendo tomado las decisiones pertinentes ya podremos comenzar el desarrollo del código de este proyecto. En el capítulo correspondiente a esta tarea se presentará el trabajo realizado en este aspecto, así como los problemas que se nos han ido presentando. Al final del desarrollo, se deberán realizar las pruebas correspondientes para cada una de las funcionalidades implementadas. Además, se presentará una demostración, así como la guía de instalación y uso para llevarla a cabo, todo esto dentro de la etapa de documentación. El diagrama de Gantt resultante se muestra en la figura 3.1 y en él se ofrece la planificación de todas las tareas mencionadas en esta sección.

3.7 Presupuesto

En esta sección se presenta el presupuesto total estimado para financiar la realización de este TFG. Para llevar a cabo la estimación, se calcularán las horas totales que se han dedicado a cada tarea.

Tarea	Horas
Formación blockchain	80
Investigación identidad autosoberana	550
Estudio sobre los diferentes métodos DID	40
Comparativa de los estándares sobre generación y presentación de credenciales	200
Informe de los análisis	80
Base de datos	40
Diagrama de clases	40
Modificación de la wallet	80
Desarrollo de nuevas interfaces	100
Despliegue de smart-contracts con Truffle	20
Integración con Ganache	20
Documentación del código	30
Pruebas	40
Documentación para usuarios	30
Redacción del TFG	140
Total	**1450**

Tabla 3.3: Horas del proyecto

Este trabajo se ha realizado dentro de una empresa fuera de un convenio de prácticas. Por este motivo, una forma sencilla de calcular el coste de este proyecto sería sumar el salario total de los meses trabajados. Para evitar compartir información sensible de la empresa, asumiremos un salario medio de mercado para un informático. Durante los primeros meses del proyecto, desde septiembre hasta enero, se dedicó una carga de trabajo a tiempo parcial (cuatro horas), y a partir de febrero, se pasó a una dedicación a jornada completa (siete horas y media).

Como se puede ver en la tabla 3.3, las horas totales del proyecto han sido 1450. Un TFG tiene una carga de 12 créditos en la carrera. Cada crédito representa 25 horas de trabajo, dando un tiempo estimado de 300 horas. Este número queda muy por debajo de las horas que se le han dedicado a este proyecto.

Para calcular el coste, se asumirá un salario medio de un ingeniero informático de 1640 euros al mes. Este dato ha sido tomado de un análisis publicado por la página Jobted. Con dicho salario, el trabajo realizado en este proyecto tendría un valor de 15.580 euros.

Las horas invertidas han sido aprovechadas por el ITI para que el esfuerzo dedicado cubra las tareas planteadas en dos proyectos de investigación anuales, donde también se ha colaborado en la redacción de los artículos a publicar, informes internos y documentos entregables correspondientes, en los que se ha podido integrar algunas secciones de este TFG. Por ello, a efectos prácticos las horas de dedicación exclusiva al TFG rondarían las 1000 y su coste rondaría los 10000€.

Es importante tener en cuenta que el cálculo del coste se ha realizado asumiendo un salario medio y que los costos reales pueden variar según las condiciones contractuales y la experiencia del personal involucrado. Además, el cálculo no incluye otros gastos asociados al proyecto, como materiales, equipos, o infraestructura, que también deben ser considerados en una estimación completa del presupuesto.

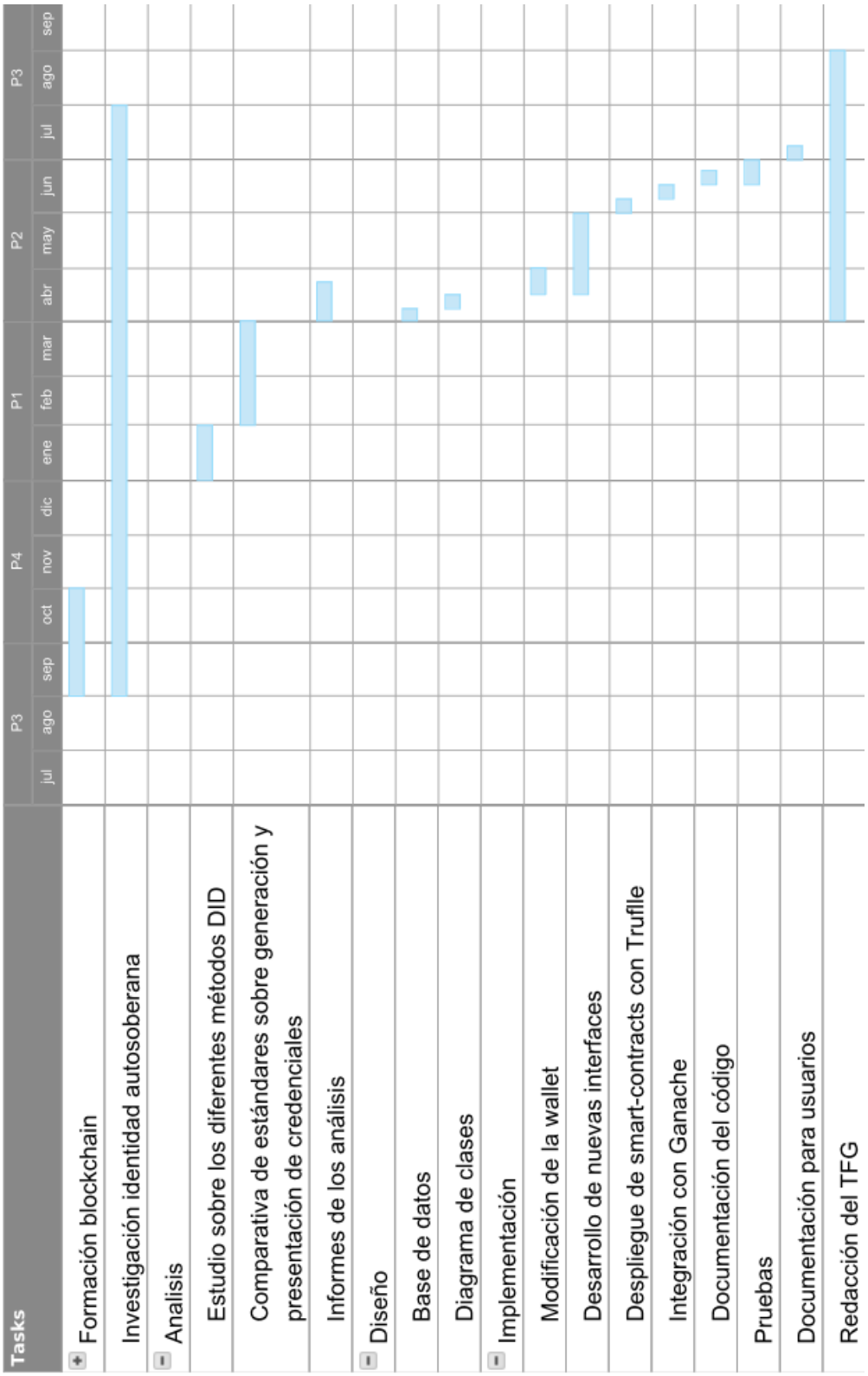

Figura 3.1: Diagrama de Gantt del proyecto.

CAPÍTULO 4
Diseño

Nuestra solución pretende demostrar cómo funcionaría un sistema de gestión de identidades basado en identidad autosoberana. Para conseguirlo trabajaremos con un framework que ya se ha presentado, Veramo. Utilizar este framework no soluciona el problema ya que tendremos que extenderlo para conseguir nuestro objetivo, pero nos proporcionará una base sobre la que trabajar.

4.1 Arquitectura del sistema

Para este proyecto vamos a seguir una arquitectura distribuida basada en agentes. Dichos agentes podrán interactuar entre ellos a través de mensajes basados en el protocolo "DIDComm Messaging v2". A su vez, dichos agentes utilizarán una red blockchain para registrar sus respectivos DID, claves públicas y endpoints. La red que se formaría entre los agentes tendría la forma de un peer-to-peer, una red entre iguales en la cual los participantes no dependen de servidores para establecer la comunicación.

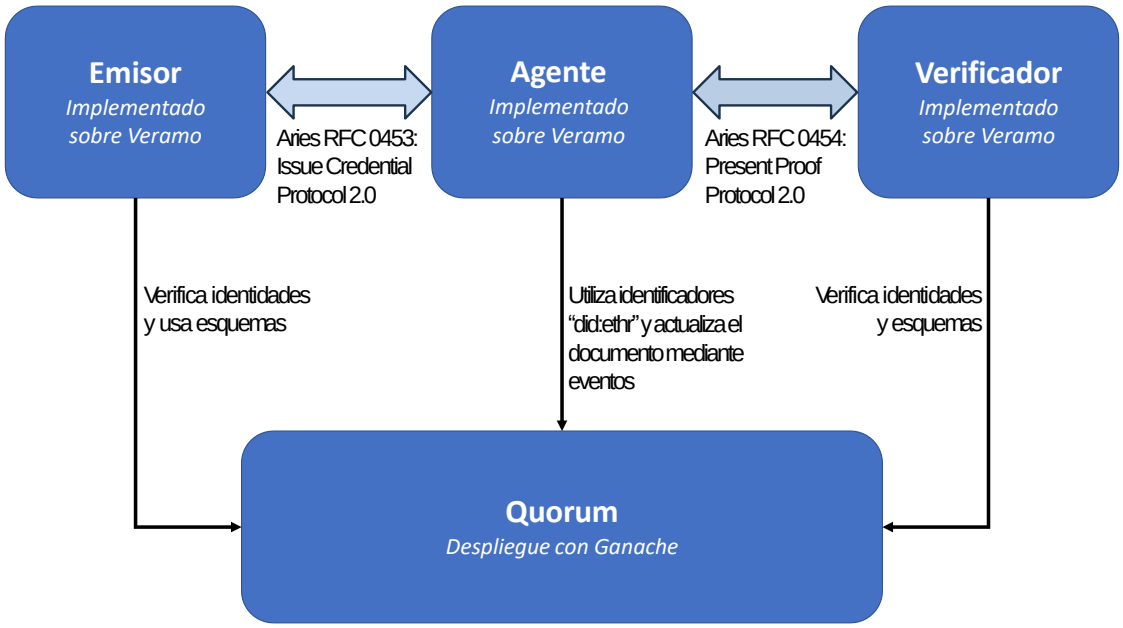

Figura 4.1: Diagrama del sistema.

El diseño de la arquitectura debe seguir el esquema propuesto por W3C (Figura 2.3). El diseño se complementará con la elección de las soluciones necesarias para desarro-

llar sus componentes, así como con los protocolos adecuados para generar y compartir las credenciales (Figura 4.1). Como se mencionó anteriormente, cada agente podrá desempeñar los tres roles especificados. Los agentes estarán implementados utilizando el framework de Veramo y la blockchain que utilizaremos como registro de datos verificables será Ethereum. Para nuestro caso de uso, no es necesario realizar un despliegue completo en la red principal de Ethereum con los costos asociados. En su lugar, utilizaremos Ganache, un software que nos permitirá desplegar un nodo Ethereum local con una lista de cuentas disponibles.

4.2 Diagrama de clases

En este apartado se va a presentar el diagrama de clases que permitirá ver cómo está organizado el código desarrollado. Al estar implementado sobre Veramo, se realizará una diferenciación entre las clases originales, las clases extendidas y las clases desarrolladas desde cero. Para esto, estarán marcadas en verde las clases que pertenecen al código original de Veramo. El diagrama está representado en la figura 4.2. Todas estas interfaces y clases ofrecen asimismo un buen número de métodos que también deberían incluirse en este diagrama de clases. Al haber tantos métodos, resulta inviable representarlos todos dentro del mismo diagrama. Por este motivo en el diagrama solo estarán especificados los atributos y métodos con los que se hayan extendido las clases de Veramo. Aparte del diagrama de clases también se presentará una tabla con los enlaces a la documentación oficial de Veramo para que el lector pueda consultar todos los métodos (Tabla 4.1). Obsérvese que las interfaces "IAgentPlugin" y "AbstractMessageHandler" no se incluyen en la tabla porque el diagrama ya incluye todas sus operaciones.

Nombre	Descripción
IDataStore	Interfaz básica de almacenamiento de datos.
IDataStoreORM	Interfaz de consulta predeterminada para los datos de credenciales almacenados por un agente Veramo.
DataStore	Esta clase implementa la interfaz IDataStore utilizando una base de datos compatible con TypeORM.
DataStoreORM	Esta clase implementa la interfaz de consulta IDataStoreORM utilizando una base de datos compatible con TypeORM.

Tabla 4.1: Interfaces de Veramo.

Como se puede observar en el diagrama, hemos utilizado Veramo como base para el diseño de todas las clases. En primer lugar, hemos utilizado la interfaz para plugins que proporciona Veramo y la hemos ampliado con los métodos necesarios para enviar los mensajes de los estándares de comunicación elegidos. Los mensajes, el flujo y el contenido se explicarán en mayor profundidad en la siguiente sección. Para poder soportar estos mensajes, hemos preparado dos controladores o 'handlers' para identificar el tipo de cada mensaje recibido y cómo proceder en cada caso.

Para cumplir el papel de cartera digital, Veramo utiliza una base de datos local SQLite. Esta base de datos está configurada para inicializarse con una serie de tablas diferentes para almacenar DID, credenciales, mensajes, así como los pares de claves público-privadas. En este aspecto, está bastante completa, pero la hemos ampliado para poder registrar los esquemas de las credenciales que se van a transmitir. Para conseguir esto se han modificado los ficheros de migración de la base de datos para definir la nueva tabla con sus columnas, y añadirla en el proceso de inicialización.

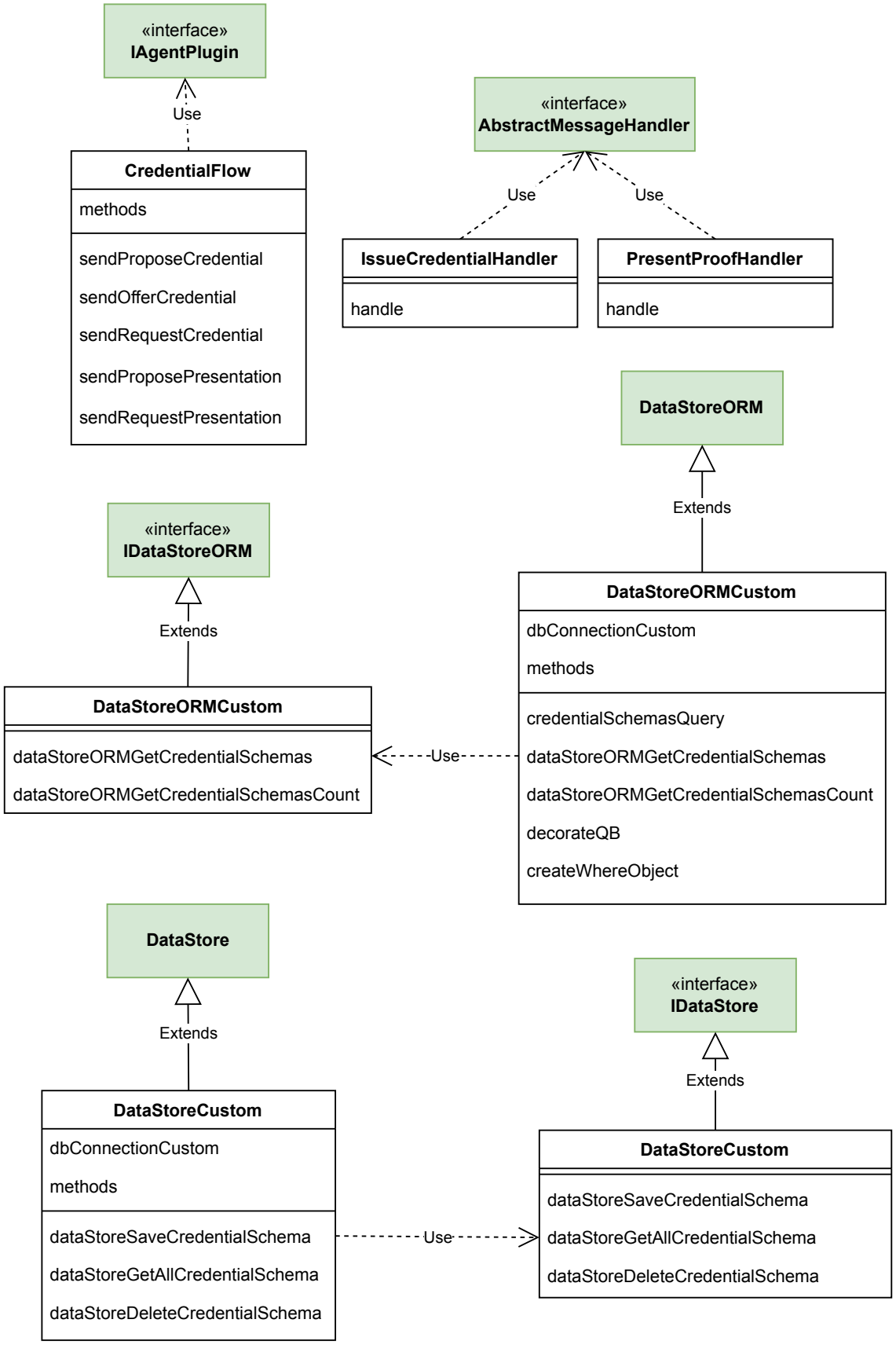

Figura 4.2: Diagrama de clases del proyecto.

Al registrar los esquemas, un generador de credenciales podrá informar a un posible propietario sobre el formato que tendrá la credencial. En caso de recibir una solicitud, podrá comprobar el tipo de credencial requerida para determinar si es capaz de emitirla. Esto también aplica al verificador de las credenciales: cuando recibe una propuesta de credencial para verificar, puede comprobar si es compatible con ese tipo.

Las funciones relacionadas con la base de datos se dividen en dos clases: DataStore y DataStoreORM. La primera permite trabajar con SQLite como un registro clave-valor, por lo que proporciona métodos básicos para admitir operaciones CRUD. Sin embargo, cuando se trabaja con tablas que tienen muchos valores posibles, como los DID o las credenciales, es necesario realizar consultas más complejas que las que proporciona una estructura clave-valor. Para esto, Veramo proporciona la clase DataStoreORM, que ofrece una capa de abstracción sobre la base de datos utilizando TypeORM. Gracias a esto, se pueden realizar consultas complejas especificando campos y valores concretos. En ambos casos, hemos extendido tanto la interfaz como la clase para agregar los métodos correspondientes para los esquemas de credenciales. En el caso de la clase DataStoreORM también habrá que modificar los ficheros de migración para añadir la entidad correspondiente a los esquemas.

4.3 Diagramas de actividad

Como ya se ha dicho antes, en esta sección se van a presentar los estándares de interacción que se van a implementar. Cuando se presentó el proyecto de Hyperledger Aries explicamos que actualmente tiene aceptados alrededor de 40 RFC. Parte de estos consistían en segundas versiones que todavía no habían recibido implementaciones de referencia. En nuestro caso nos interesan los estándares para la emisión y la presentación de credenciales [30, 29]. Estos especifican el formato que deberán seguir los mensajes que se enviarán a través de DIDComm Messaging.

Para poder comprender el sistema de comunicación implementado vamos a exponer los diagramas para los diferentes escenarios que se plantean. Al haber implementado un estándar de Aries, los diagramas de actividad ya vienen dados por la especificación: emisión de credenciales (Figura 4.3) y presentación de credenciales (Figura 4.4).

4.3.1. Emisión de credenciales

En este escenario se definen dos roles: el emisor y el propietario. Aunque se le llame 'propietario', el participante no necesita credenciales para esta comunicación. Técnicamente, se convertirá en propietario de la credencial solo cuando se complete exitosamente la secuencia de mensajes.

La especificación define cuatro mensajes para este flujo de comunicación. El primero es un tipo de mensaje llamado 'propose', que el propietario envía al emisor indicando qué espera recibir. A este mensaje, el emisor puede responder con una 'offer' para informar al potencial propietario qué es lo que pretende emitir, así como proporcionar información adicional, como el costo de la credencial. Ambos mensajes son opcionales, ya que su principal función es permitir la discusión entre ambos roles sobre cómo debería ser la credencial generada. Por ejemplo, una vez recibida una oferta, el propietario podría enviar otra propuesta si no está de acuerdo con los campos o la información contenida en la credencial.

Una vez que se haya llegado a un acuerdo sobre la credencial, el propietario podrá enviar un 'request' al emisor para que este le proporcione la credencial verificable. Si la

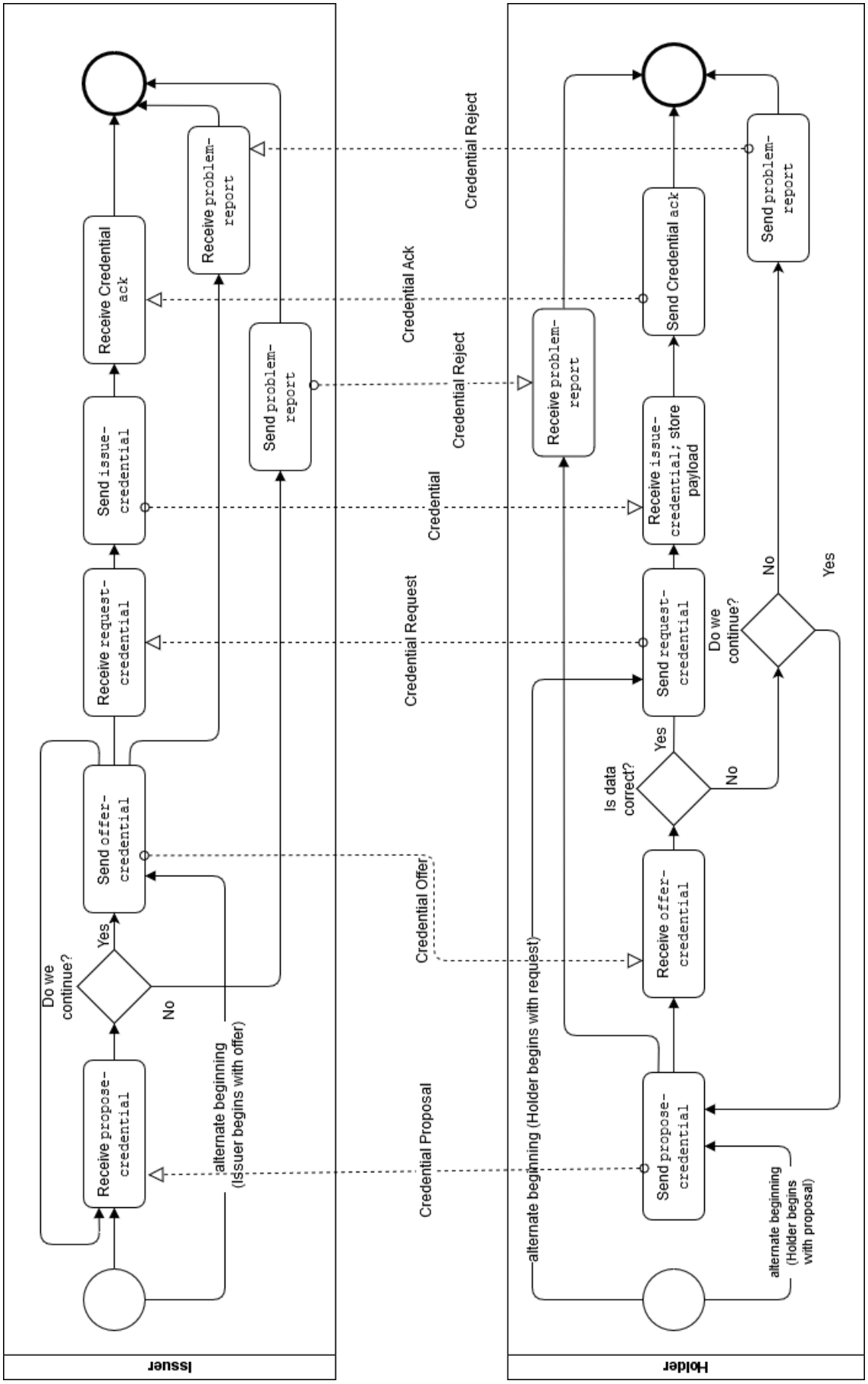

Figura 4.3: Flujo de comunicación para la emisión de credenciales [30].

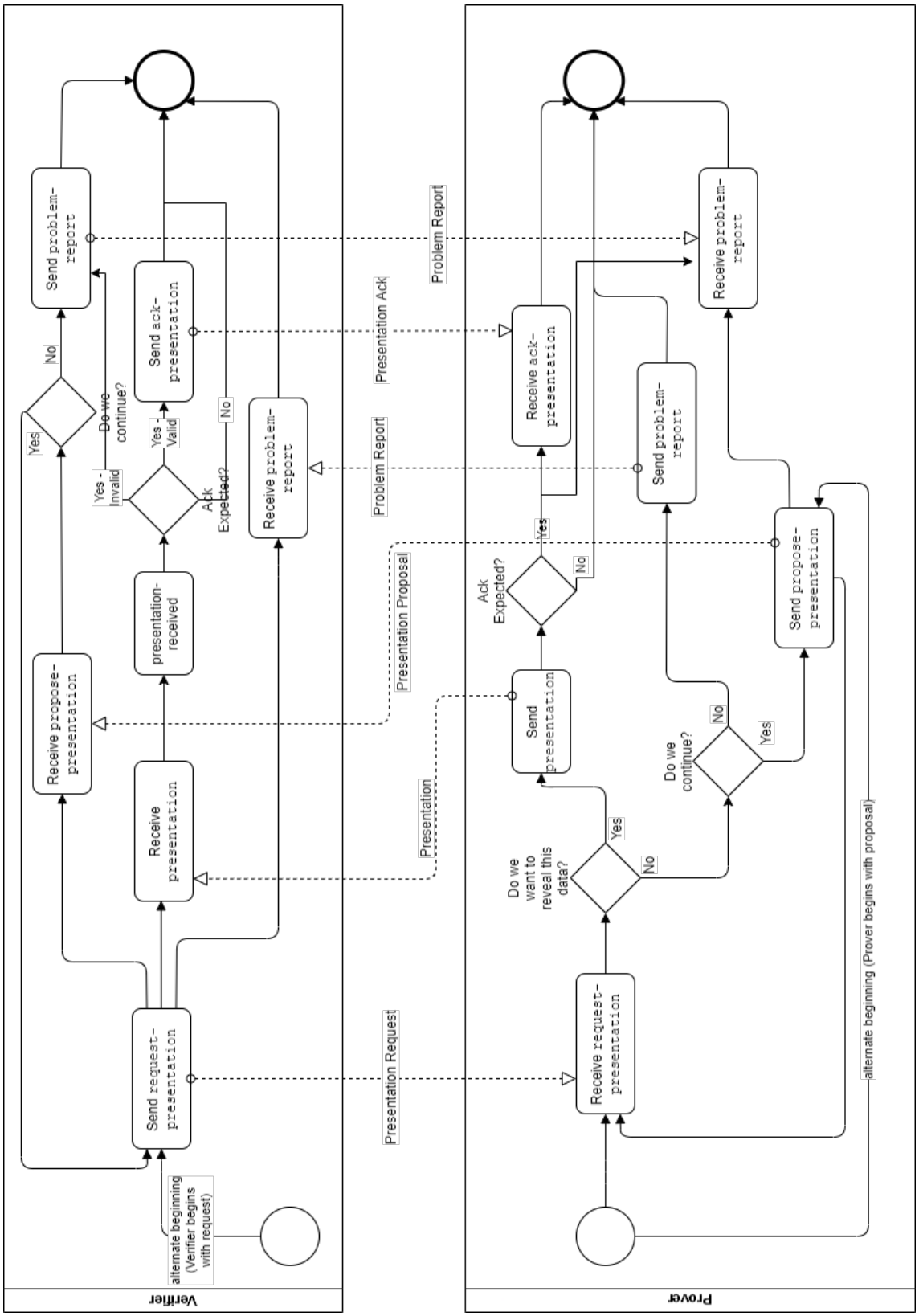

Figura 4.4: Flujo de comunicación para la presentación de credenciales [29].

discusión mencionada anteriormente no es necesaria, el propietario podría comenzar el protocolo de comunicación directamente con este mensaje. Por último, en respuesta a este mensaje, el emisor enviará un mensaje 'issue' que contendrá la credencial verificable adjunta. Además de estos cuatro mensajes, el estándar también soporta los mensajes 'ack' y 'problem-report' para confirmar o notificar un problema durante la comunicación.

La comunicación puede ser iniciada a través del primer par de mensajes para llegar a un acuerdo, o directamente desde el 'request'. La única diferencia es que este último mensaje no espera una negociación, sino que la petición es aceptada o rechazada.

Este estándar define los mensajes necesarios para la emisión de las credenciales, pero no define los diferentes esquemas de credenciales. Para soportar los diferentes formatos y esquemas existentes, este estándar hace uso de adjuntos o "attachments". Estos adjuntos sirven para separar la lógica de la comunicación de los formatos de las propias credenciales. Esto permite que ambos avancen en sus versiones sin depender el uno del otro. Los adjuntos se representan en forma de lista o 'array' dentro del propio mensaje. Dicha lista permite emitir la credencial en diferentes formatos, como JSON-LD, JWT o el DIF Manifest, pero representando siempre la misma información. La lista no debe utilizarse para emitir credenciales con diferente contenido.

4.3.2. Presentación de credenciales

En el caso de la presentación de credenciales, se pueden identificar dos roles: el verificador y el propietario o 'proveedor'. El verificador es responsable de solicitar la presentación al propietario. Si el propietario acepta la solicitud, generará la presentación y la enviará al verificador. Opcionalmente, el estándar también permite que el propietario inicie el protocolo de comunicación utilizando un tipo de mensaje específico.

En este escenario, la especificación define tres mensajes para el flujo de comunicación. El caso más común es que el verificador inicie el proceso con un 'request', en la cual especifique el tipo de credencial que espera recibir. En este mensaje, el verificador también debe incluir un campo de 'challenge' para verificar la firma del propietario. Una vez que el propietario reciba la solicitud, deberá dar su consentimiento para compartir su información. Si acepta, la billetera digital generará la presentación verificable basada en la credencial solicitada y la enviará al verificador. En caso de que el propietario decida no compartir la credencial solicitada, puede optar por finalizar la comunicación o continuarla. Para mantener la comunicación en curso, el propietario puede enviar un mensaje de 'propose' para indicar qué información le gustaría utilizar. Este mensaje, junto con la solicitud, permite establecer una discusión entre ambas partes. El mensaje de propuesta también puede servir para que el propietario inicie el flujo de comunicación.

Una vez que se llegue a un acuerdo entre los dos roles, el propietario generará la presentación verificable, la adjuntará en el formato acordado y la enviará al verificador en un mensaje 'presentation'.

4.4 Tecnologías utilizadas

En esta sección, presentaremos la lista de tecnologías que se ha decidido utilizar para este proyecto. Junto con cada presentación de una tecnología, también se proporcionará una explicación de por qué se ha elegido. Comprender estas tecnologías nos permitirá apreciar cómo se han abordado los desafíos técnicos y las diferencias con las especificaciones.

4.4.1. Veramo

Veramo es un framework de JavaScript/Typescript para credenciales verificables que fue diseñado para ser flexible y modular, lo que lo hace fácilmente adaptable a muchos flujos de trabajo complejos.

Permite crear un agente, agregar plugins, ejecutarlo en un servidor, en el frontend o en dispositivos móviles, o combinar todas estas opciones. Veramo funciona en Node, navegadores y React Native directamente desde el principio. Ahorra tiempo utilizando la misma API en todas las plataformas.

Consta de un núcleo y algunos complementos. El núcleo proporciona un punto de entrada a la API, une los complementos y les permite interactuar entre sí. Dependiendo de los complementos que se utilicen, una instancia de Veramo (un agente) puede desempeñar una variedad de funciones: crear y gestionar claves para firmar y encriptar, crear y gestionar DID, emitir credenciales verificables y presentaciones, verificar dichas VC y VP o comunicarse con otros agentes utilizando DIDComm Messaging entre otras cosas.

Hemos elegido Veramo debido a que nos brinda una sólida base sobre la cual trabajar, ya que implementar todos los componentes desde cero resulta imposible para una sola persona. Además, Veramo ofrece una gran facilidad para extender sus complementos y adaptarlos a nuestras necesidades. Podemos modificar sus bases de datos según sea necesario, así como ampliar las funcionalidades de algunos componentes para ajustarlos a nuestros requerimientos específicos.

4.4.2. SQLite

SQLite es un sistema de gestión de bases de datos relacionales (RDBMS) ligero y autónomo que se implementa como una biblioteca de software. A diferencia de otros sistemas de gestión de bases de datos que se ejecutan como procesos separados y requieren un servidor dedicado, SQLite se integra directamente en la aplicación que lo utiliza.

Es ampliamente utilizado en aplicaciones de software embebido y móviles, así como en aplicaciones de escritorio. Debido a su simplicidad, ligereza y capacidad para operar sin un servidor dedicado, es una opción popular para aplicaciones que requieren una base de datos local de tamaño moderado.

SQLite ya está implementado en Veramo, lo cual es perfecto para satisfacer sus necesidades. Es una solución ligera y simple que permite almacenar miles de entradas. En caso de ser necesario, se podría actualizar el componente para utilizar otra base de datos más avanzada, como PostgreSQL o MySQL.

4.4.3. TypeORM

TypeORM es un ORM (Object-Relational Mapping) de alto rendimiento para TypeScript y JavaScript que se utiliza para mapear objetos y clases a tablas y consultas de base de datos relacionales. Proporciona una capa de abstracción sobre la base de datos, lo que permite a los desarrolladores interactuar con la base de datos utilizando objetos y métodos familiares en lugar de escribir consultas SQL directamente.

Es ampliamente utilizado en proyectos de TypeScript y JavaScript para simplificar el acceso y la manipulación de datos en bases de datos relacionales. Proporciona una abstracción fuerte y flexible sobre la base de datos, lo que facilita el desarrollo de aplicaciones escalables y mantenibles.

TypeORM también está incluido en Veramo, lo que nos proporciona una de sus funcionalidades más útiles: la capacidad de realizar consultas más complejas filtrando por campos específicos, en lugar de tener que iterar manualmente a través de cada entrada en las tablas. Esto resulta especialmente útil en el caso de las credenciales verificables, dado que suelen tener una gran cantidad de campos. En ocasiones, nos encontraremos en la situación de necesitar encontrar una credencial de un determinado tipo, emitida por un emisor específico, para uno de nuestros DID.

4.4.4. TypeScript

TypeScript es un lenguaje de programación de código abierto y de alto nivel que se basa en JavaScript. Fue desarrollado por Microsoft y se caracteriza por agregar características de tipado estático a JavaScript, lo que permite detectar errores en tiempo de compilación y mejorar la calidad y robustez del código. Esto significa que TypeScript ofrece una capa adicional de seguridad y herramientas de desarrollo más avanzadas en comparación con JavaScript.

Una de las principales ventajas de TypeScript es su capacidad para ayudar a los desarrolladores a mantener grandes bases de código de manera más eficiente. Al especificar tipos de datos en las variables, funciones y objetos, TypeScript permite detectar posibles errores antes de que el código se ejecute. Esto facilita el trabajo en equipo, la colaboración y la refactorización del código, ya que los errores se pueden identificar y corregir rápidamente.

Además, TypeScript es compatible con la mayoría de los editores de código y entornos de desarrollo, y ofrece una amplia gama de herramientas y bibliotecas adicionales. Esto incluye características como autocompletado inteligente, navegación de código, refactorización automatizada y documentación mejorada. Además, al compilar el código TypeScript, se genera código JavaScript compatible, lo que significa que las aplicaciones TypeScript se pueden ejecutar en cualquier entorno que admita JavaScript, incluidos los navegadores web y los servidores. En resumen, TypeScript es una opción poderosa para aquellos desarrolladores que desean agregar tipado estático y funcionalidades avanzadas a sus proyectos JavaScript, mejorando así la calidad y el mantenimiento del código.

Además de la lista de ventajas mencionadas anteriormente, la razón principal para utilizar TypeScript ha sido nuestra experiencia previa con el entorno de desarrollo de Node.js, en el cual Veramo está implementado. Esta familiaridad con el entorno nos permite aprovechar al máximo las capacidades de Veramo y optimizar nuestro flujo de trabajo.

4.4.5. Truffle suite

Truffle es un framework de desarrollo de smart contracts para Ethereum. Proporciona una suite de herramientas y utilidades que facilitan la creación, compilación, implementación y prueba de contratos inteligentes. Ofrece una estructura de directorios predefinida para organizar el código del contrato y los archivos de configuración, lo que facilita la gestión del proyecto. Además, Truffle incluye un sistema de compilación que permite compilar los contratos a bytecode y generar artefactos que contienen información sobre los contratos y sus funciones.

Truffle también facilita la implementación de contratos inteligentes en una red de prueba o en la red principal de Ethereum. Proporciona una API que se puede utilizar para interactuar con los contratos desplegados y realizar pruebas automatizadas. Además, Truffle incluye un entorno de desarrollo integrado (IDE) llamado Ganache, que permi-

te simular una red de desarrollo local con cuentas de Ethereum preconfiguradas para facilitar las pruebas y la depuración de los contratos.

De entre todas las opciones disponibles para trabajar con una red de pruebas de Ethereum, hemos encontrado que la suite de Truffle es la más útil y sencilla. No requiere una configuración extensiva ni una larga lista de pasos para su implementación local. Truffle nos ha permitido agilizar nuestro desarrollo y realizar pruebas de manera eficiente en un entorno controlado.

CAPÍTULO 5
Desarrollo

En este capítulo, se explicará el trabajo realizado para el desarrollo de ese TFG. Para poder llevar a cabo este capítulo hemos tenido que realizar una importante investigación y un posterior análisis sobre cada uno de los aspectos del proyecto. El capítulo proporcionará una explicación detallada de cómo se han implementado cada uno de los componentes y que cambios o diferencias presentan respecto a los estándares que se han decidido utilizar.

5.1 Controladores de mensajes

Los controladores de mensajes son los componentes encargados de gestionar la recepción, procesamiento y envío de mensajes. Estos controladores actúan como intermediarios entre diferentes componentes, como el usuario, el servidor y otros sistemas, y se encargan de garantizar que los mensajes sean recibidos, interpretados y respondidos de manera adecuada. Una vez recibido un mensaje, el controlador aplica lógica de negocio, realiza validaciones, ejecuta operaciones y procesa la información contenida en el mensaje.

En nuestro caso es uno de los componentes centrales ya que es el encargado de filtrar los mensajes que soportan los estándares de Aries y ejecutar los mecanismos correspodientes para cada uno. En algunos este proceso será tan sencillo como validar una credencial y almacenarla en la wallet, pero en otros puede suponer realizar varias consultas a la base de datos Los controladores son los componentes encargados de procesar los mensajes recibidos por el endpoint. Se han implementado dos controladores diferentes, uno para cada estándar. En esta sección vamos a presentar cómo se ha implementado el código y las posibles diferencias que se plantean con la especificación.

5.1.1. Emisión de credenciales

La clase controladora para este escenario se llama “IssueCredentialHandler”. Soporta los mensajes de los tipos especificados en el RFC “IssueCredential v2”. El primer mensaje definido es el “credential-propose”. Si recordamos lo explicado en el diagrama de actividad, la propuesta servía para que el propietario pudiera comunicar al emisor qué información le gustaría recibir. Para esto, la especificación hace uso de un objeto “credential_preview”. Dicho objeto consiste en una lista de atributos con nombre, valor y tipo MIME. En caso de omitir el tipo, se entiende como string.

Utilizar una preview al comienzo de la comunicación es muy útil para un propietario para descubrir qué credenciales un emisor podría generar. Así, el propietario no tendría

```
0  {
1    "@type": "https://didcomm.org/issue-credential/2.0/
        credential-preview",
2    "attributes": [
3    {
4      "name": "grado",
5      "value": "Universitario"
6    },
7    {
8      "name": "dni",
9      "value": "20085452C"
10   }
11   ]
12 }
```

Listado 5.1: Ejemplo "credential_preview"

que consultar qué esquemas soporta el emisor, solo tendría que enviar la información que le gustaría recibir en una credencial, y sería trabajo del emisor comprobar si la preview es compatible con alguno de sus esquemas.

En un escenario real, el proceso de validación de la preview conlleva varios pasos. Tomando como ejemplo una universidad, primero esta deberá comprobar si tiene algún esquema que contenga los valores propuestos por el propietario. En caso de que haya coincidencia, además deberá verificar que la información solicitada es correcta, ya que una persona podría solicitar un título que no posee. También podría darse el caso de que la universidad tenga una credencial con algunos campos en común, por lo que sería decisión de la implementación si ofrecer esa posible credencial.

Para nuestro prototipo, hemos decidido optar por una coincidencia exacta al proponer una credencial. El emisor solo devolverá una oferta si tiene un esquema que contenga todos los campos de la preview y los nombres coinciden. Una vez finalizado este proceso, si hay una coincidencia, el emisor devolverá un mensaje de tipo "credential-offer".

Dentro del mensaje de oferta, el emisor también puede hacer uso del campo "replacement_id". Este campo se utiliza para informar que la posible credencial generada sustituirá a la indicada en el campo. Su principal función es en el caso de que el propietario haya solicitado dicha credencial en el pasado, para evitar acumular credenciales repetidas, el emisor le indica cuál debería sobrescribir.

El RFC recibió una actualización en la cual se añadió un campo "multiple_available" para permitir, según ellos, la emisión de múltiples credenciales. En la práctica, este campo se queda corto ya que solo indica al propietario que puede emitir la credencial ofertada con otros valores, como una universidad que ofrece una credencial de titulación a una persona con varias carreras. Este nuevo campo no sirve para informar al propietario de que existen varios esquemas compatibles y qué información contienen esos esquemas. Por este motivo, hemos planteado una implementación diferente. Ahora, cuando un emisor recibe una propuesta, responderá con una lista de las posibles credenciales compatibles que podría emitir y qué información contendrían. Este cambio se ha planteado basando en el borrador sobre la emisión de credenciales verificables de OpenID [25].

La imagen en el ejemplo A.3 representa una lista con dos credenciales ofertadas que el propietario ha recibido del emisor. La primera representa un certificado de graduación en Ingeniería Informática y la otra un certificado de estudiante en Ingeniería Electróni-

ca. Ambas credenciales contienen los campos establecidos en la preview: 'algo de grado universitario' para el DNI 20085452C.

Para que una credencial se considere verificable, debe estar firmada por el emisor para poder comprobar su validez. Sin embargo, como se puede ver en el ejemplo de las ofertas, estas no contienen el campo "proof" definido que contenga la firma, por lo que aún no son credenciales verificables. Si el propietario está interesado en alguna de ellas, enviará un mensaje "request" en el cual solo tendrá que especificar el tipo de credencial que desea, por ejemplo, "UniversityDegreeCredential".

Finalmente, cuando el emisor reciba el "request", volverá a comprobar todo lo mencionado anteriormente durante la generación de la oferta, pero esta vez solo devolverá la credencial verificable (Listado 5.2). El contenido completo del mensaje se encuentra en el apéndice A.1.

```
0  {
1    "@context": [
2    "https://www.w3.org/2018/credentials/v1",
3    {
4      "grado": "https://schema.org/",
5      "titulo": "https://schema.org/"
6    },
7    "https://w3id.org/security/suites/secp256k1recovery-2020/v2"
8    ],
9    "id": "3b800dc8-d8a3-48e8-8f48-4f1b2bc7aca8",
10   "issuer": "did:ethr:development:0x02a7b4984ecb19b7c5a2986735
       e5d51924a9767323f5d0e37516dd256e87420143",
11   "type": ["VerifiableCredential", "UniversityDegreeCredential
       "],
12   "credentialSubject": {
13     "id": "did:ethr:development:0x025862baf2cdb6937f9fef454da3
         da96fe925764b9b9362507f732d1bd09e9702f",
14     "grado": "Universitario",
15     "titulo": "Ingenieria Informatica",
16     "dni": "20085452C",
17     "curso": "2021-2022"
18   },
19   "issuanceDate": "2023-07-04T06:56:15.947Z",
20   "proof": {
21     "type": "EcdsaSecp256k1RecoverySignature2020",
22     "created": "2023-07-04T06:56:15Z",
23     "verificationMethod": "did:ethr:development:0x02a7b4984ecb
         19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143#
         controller",
24     "proofPurpose": "assertionMethod",
25     "jws": "eyJhbGciOiJFUzI1NkstUiIsImI2NCI6ZmFsc2
         UsImNyaXQiOlsiYjY0Il19..3YvBe4kvP3uao97-
         QMBquYWTjbsCHlbV8fuZHiw2hY1UAAur0y5Zv_coDmjCdPLX_G7zGUs
         9DeAccC7hvFrCQQE"
26   }
27 }
```

Listado 5.2: Ejemplo credencial verificable

5.1.2. Presentación de credenciales

En el escenario para presentación de credenciales vamos a comenzar con los mensajes de discusión: "propose" y "request". Ambos mensajes utilizan el mismo formato para los adjuntos de los mensajes, el formato DIF para el intercambio de presentación el cual está definido en uno de los RFC aceptados de Aries [18].

Este RFC registra un formato para cada uno de los mensajes definidos en el RFC "present-proof v2". Los formatos están basados en la especificación para el intercambio de presentaciones de DIF. Los primeros definen un contenedor para los objetos de los mensajes "propose" y "request", en los cuales definir los parámetros de la presentación, mientras que el tercer formato es solo un contenedor para la presentación final establecida.

El mensaje "propose" contiene un objeto "input_descriptors" el cual contiene información sobre la credencial que se pretende presentar, como el nombre y el esquema (Listado A.5). Para el mensaje "request" que el verificador envía al propietario se utiliza el objeto "presentation_definition", el cual contiene un campo "input_descriptors" también además de un campo "format" que servirá para especificar el tipo de firma que el propietario deberá generar (Listado A.6).

Finalmente, el propietario responde con un mensaje "presentation" que contendrá una "presentation_submission" (Listado A.7). Este objeto fue definido por el DIF en su especificación como una lista de claims y sus valores, los cuales deben respetar los definidos en el objeto 'presentation_definition'.

5.1.3. Formato de las credenciales

En esta sección se va a discutir el formato de las credenciales a utilizar. Primero se van a presentar los formatos que contiene la recomendación y luego se explicará nuestra experiencia con ellos, así como los problemas que se han encontrado durante el desarrollo y la justificación del formato que finalmente se utiliza.

En la recomendación sobre las credenciales verificables de W3C, se presentan los posibles formatos para la representación de estas. Las tres opciones que presentan son: JSON, JSON-LD y JWT. Aunque expliquen solo estos, se menciona que se podrían utilizar otros formatos de representación de datos como XML, YAML o CBOR.

Las credenciales se pueden representar fácilmente en formato JSON, que además es el formato más comúnmente utilizado para ello. JSON-LD se presenta como una mejora sobre JSON, diseñado para facilitar la transición en sistemas que ya utilizan JSON. JSON-LD se codifica de manera similar a su predecesor, pero con el añadido de un campo "@context". Este campo permite extender la información de una credencial verificable en caso de que el emisor lo requiera. También se utiliza este formato para los esquemas de las credenciales, ya que el contexto permite definir nuevos esquemas basándose en otros ya existentes. Por último, se presentan los JWT, un formato que ha sido ampliamente implementado en procesos de autenticación y autorización. Se utilizan para realizar las transformaciones necesarias sobre una credencial con el fin de cumplir con la especificación de este tipo de tokens.

En un principio, decidimos emplear el formato de serialización de los "JSON Web Tokens" para transmitir credenciales verificables. Este formato se usa para codificar una serie de atributos o información, que irán firmados con una "Json Web Signature" (JWS) [21].

Como se puede observar en el listado 5.2, la credencial consiste en un objeto JSON con varios atributos, incluyendo un campo "proof" con una JWS. Para obtener un JWT a partir de una credencial, es necesario realizar ciertas transformaciones. Al final, el resultado debería tener una estructura similar a la mostrada en el listado 5.3.

```
0 {
1     "iss": "did:ethr:development:0x02a7b4984ecb19b7c5a2986735e
         5d51924a9767323f5d0e37516dd256e87420143",
2     "iat": 1689848861,
3     "exp": 1689850063,
4     "sub": "did:ethr:development:0x025862baf2cdb6937f9fef454da
         3da96fe925764b9b9362507f732d1bd09e9702f",
5     "grado": "Universitario",
6     "titulo": "Ingenieria Informatica",
7     "dni": "20085452C",
8     "curso": "2021-2022"
9 }
```

Listado 5.3: Set de atributos del JWT

Una vez se han definido los atributos que contendrá el JWT, se deberá codificar y firmar para obtener el JWT (Figura 5.1). En la figura de ejemplo se ha destacado en color las secciones que forman un token de este tipo. En un escenario real, estas secciones se separan mediante los puntos. La primera sección contiene unas cabeceras, como el tipo de algoritmo de digestión utilizado. La segunda sección, resaltada en rosa, contiene todos los atributos listados anteriormente. Finalmente, la última sección contiene la firma que el receptor del token deberá verificar.

Figura 5.1: Credencial del listado 5.3 en formato JWT.

Debido a la falta de documentación, se ha tenido que realizar un esfuerzo extra durante el proceso de aprendizaje de Veramo. Hemos tenido que analizar directamente el código y realizar muchas pruebas para comprender cómo funcionaba su proyecto. Du-

rante este proceso, se descubrió que su implementación de los JWT no era correcta. A la hora de generar una credencial con el plugin de Veramo, hay diferentes opciones para la firma de la credencial: "EthereumEip712Signature2021", "Linked Data Signature" o JWT. Realizando pruebas, se comprobó que la opción de JWT no devolvía un JWT, sino una credencial verificable que dentro del campo "proof" contenía un JWT. Después de revisar la especificación de los JWT (RFC7519) y la recomendación de W3C, se confirmó que ese no era un uso correcto. Por este motivo, se decidió ajustar el sistema que se estaba desarrollando para que soportara credenciales verificables en formato JSON-LD en lugar de JWT. Esta modificación permitió alinear el proyecto con los estándares y buenas prácticas recomendadas por W3C y garantizar una implementación adecuada y segura de las credenciales verificables.

5.2 CredentialFlow

El agente de Veramo permite enviar mensajes DIDComm, permitiendo así el uso de firmas basadas en las claves de los DID o la encriptación de mensajes para su comunicación. En este contexto, se ha desarrollado una clase que añade la funcionalidad de enviar los mensajes de los estándares de Aries para la emisión y la presentación de credenciales verificables. Esta nueva clase no sobrescribe la de Veramo para el envío de mensajes DIDComm, sino que la extiende para permitir el envío de dichos mensajes con el formato específico establecido en Aries.

En esta nueva clase, se han incorporado los tipos de mensajes que ambos estándares de Aries han especificado como iniciadores de la comunicación. Para la emisión de credenciales, los tipos de mensajes disponibles son "propose", "offer" y "request". Para la presentación de credenciales, también se incluyen los tipos "offer" y "request", pero no se considera el tipo "propose" ya que no existe en este contexto. Los mensajes finales de emisión de la credencial y de la presentación se realizan de forma automática cuando se reciben las peticiones correspondientes.

Las funciones encargadas de los mensajes reciben como atributos el DID del otro extremo de la comunicación y el tipo de credencial que se pretende manejar. A partir de esta información, la clase procede al formateo del mensaje con los campos correspondientes en función del tipo de mensaje seleccionado. De esta manera, se asegura que los mensajes enviados cumplan con los estándares y requisitos establecidos en Aries.

Para recibir las respuestas de los mensajes enviados, es necesario contar con el controlador correspondiente, tal como se detalla en la Sección 5.1. La implementación de este controlador permitirá gestionar y procesar las respuestas recibidas, completando así el flujo de comunicación entre los agentes.

La incorporación de esta clase de envío de mensajes Aries en el agente de Veramo fortalece su capacidad para interactuar con otros sistemas y agentes que siguen los estándares de Aries. La extensión de la clase existente en lugar de su reemplazo asegura la compatibilidad con el funcionamiento original del agente, lo que contribuye a un desarrollo más modular como promueve Veramo. Gracias a esta incorporación, el agente de Veramo puede participar de manera más efectiva en el ecosistema de identidad autosoberana y facilitar el intercambio seguro y confiable de credenciales verificables entre diferentes actores.

5.3 Contrato inteligente

Los contratos inteligentes son programas utilizados en las tecnologías blockchain para su implementación. Estos programas se almacenan en bloques y se ejecutan de manera automática en la red descentralizada cuando se cumplen las condiciones programadas. Esto asegura que las transacciones queden registradas de forma inmutable. En el contexto de los Identificadores Descentralizados (DID), la función del contrato será registrar la información relacionada con su DID, para posteriormente poder resolver un identificador en su documento.

En esta sección, primero se presenta el contrato que se ha decidido emplear para el sistema y cómo maneja el concepto de identidad. Luego se abordan los tipos de eventos de actualización que soporta el contrato, y finalmente se explican las dificultades que se han encontrado.

El contrato inteligente que vamos a utilizar es el "Ethereum DID Registry". El objetivo principal de este contrato es permitir la gestión de DIDs en la red de Ethereum, donde se puedan registrar, actualizar y buscar DIDs. El contrato permite a las direcciones de Ethereum presentar información criptográfica sobre ellas mismas sin necesidad de hacer un registro previo. Permite realizar rotaciones de claves públicas, y especificar diferentes claves y servicios. El contrato se encuentra desplegado tanto en la red principal de Ethereum como en otras 15, pero el código está disponible para poder ser desplegado en cualquier red privada compatible con la "Ethereum Virtual Machine". En nuestro caso lo utilizaremos para desplegar en una red de tipo Quorum. Este proceso se presentará en el capítulo 6.

En este contrato, cada cuenta dentro del namespace de Ethereum es considerada como un Identificador Descentralizado (DID), siguiendo el formato: "did:ethr:<identificador específico>". Este identificador específico consiste en la clave pública o dirección de cuenta de Ethereum representada en formato hexadecimal.

Para crear un DID de este tipo, es necesario generar primero un par de claves (clave pública y clave privada). El registro del DID en la red es implícito, ya que es imposible adivinar una clave privada a partir de su clave pública. Por este motivo, no se necesita interactuar con la red para el registro. El propietario de la clave privada será el dueño del DID.

En el momento de resolver un DID en su documento, el contrato devolverá toda la información relacionada con ese DID almacenada en la blockchain de Ethereum. En el caso de no encontrar nada, el proceso de resolución devolverá un documento predeterminado (Figura 5.4). Como se ha comentado previamente, los DIDs no necesitan un proceso de registro formal, y todos los DIDs se consideran activos a la espera de recibir actualizaciones.

Es importante tener en cuenta que la privacidad y seguridad de los DIDs en este contrato están respaldadas por la robustez y descentralización de la red de Ethereum. Cada DID está asociado con una cuenta específica en la red, y la propiedad del DID se basa en el control de la clave privada correspondiente. Esto asegura que solo el propietario legítimo pueda actualizar la información relacionada con su DID, y los demás actores solo puedan acceder a los datos públicos asociados con el DID a través de la resolución en la blockchain de Ethereum.

El contrato está basado en el ERC1056 ("Ethereum Request for Comment 1056") [32]. Este ERC define tres tipos de eventos disponibles: "DIDOwnerChanged", "DIDDelegateChanged" y "DIDAttributeChanged".

Si algún cambio se ha realizado sobre algún DID, el número del bloque de la transacción se almacena en el contrato. Esto permite identificar rápidamente el último cambio de un DID. Además, cada uno de los eventos listados antes contiene una referencia al bloque donde está registrado el cambio anterior. Esta función permite acceder directamente a la lista de bloques que contienen cambios, de otra manera habría que iterar sobre cada bloque de la cadena hacia atrás (hacia la creación de la cadena) buscando si ha habido algún evento para dicho DID. Una vez obtenida la lista de cambios para un DID habrá que ir aplicándolos sobre el documento predeterminado uno por uno en orden cronológico.

```
0  {
1    "@context": [
2      "https://www.w3.org/ns/did/v1",
3      "https://w3id.org/security/suites/secp256k1recovery-2020/v
         2"
4    ],
5    "id": "did:ethr:0xb9c5714089478a327f09197987f16f9e5d936e8a",
6    "verificationMethod": [
7      {
8        "id": "did:ethr:0xb9c5714089478a327f09197987f16f9e5d936e
           8a#controller",
9        "type": "EcdsaSecp256k1RecoveryMethod2020",
10       "controller": "did:ethr:0xb9c5714089478a327f09197987f16f
           9e5d936e8a",
11       "blockchainAccountId": "eip155:1:0xb9c5714089478a327f091
           97987f16f9e5d936e8a"
12     }
13   ],
14   "authentication": [
15     "did:ethr:0xb9c5714089478a327f09197987f16f9e5d936e8a#
         controller"
16   ],
17   "assertionMethod": [
18     "did:ethr:0xb9c5714089478a327f09197987f16f9e5d936e8a#
         controller"
19   ]
20 }
```

Listado 5.4: Documento DID predeterminado

Una identidad puede necesitar publicar cierta información que solo se necesita off-chain pero que todavía necesita los beneficios de la blockchain. Estos atributos suelen llamarse "non-Ethereum" ya que sirven para publicar endpoints y claves critográficas de un tipo diferente al que utiliza Ethereum.

Aunque el contrato no haya sido desarrollo propio para este proyecto, se ha tenido que realizar un análisis y comprensión en profundidad del contrato. Esto ha supuesto aprender desde cero un lenguaje nuevo como es Solidity y realizar un periodo de pruebas para coger familiaridad con el entorno. Además, podemos argumentar que se ha conseguido demostrar un entendimiento exhaustivo del contrato y su funcionamiento por el hecho de haber sido capaz de identificar errores y que los desarrolladores los reconocieran.

CAPÍTULO 6
Implantación

En este capítulo, abordaremos la fase de implantación del Trabajo de Fin de Grado (TFG), en la cual se llevarán a cabo todas las acciones necesarias para poner en funcionamiento el proyecto desarrollado. En nuestro caso, el proceso de implantación implica inicialmente el despliegue de una red blockchain de pruebas y el despliegue del contrato correspondiente. Una vez completado este paso, se explicará cómo reemplazar los componentes originales de Veramo con los propios, así como los requisitos necesarios para comenzar a interactuar con otros agentes.

6.1 Despliegue de contratos inteligentes

Antes de desplegar los contratos, necesitamos una red blockchain sobre la cual trabajar. Ganache simplifica este proceso al ofrecerse como 'la blockchain en un clic'. Después de descargarlo, ejecutamos Ganache para comenzar a desplegar nuestra red. Al abrir Ganache por primera vez, podemos elegir la opción de inicio rápido o 'quickstart', ya que luego podremos configurarlo según nuestras necesidades. Por defecto, Ganache crea una cadena de bloques local con 10 cuentas pregeneradas y sus claves privadas. Además, proporciona información sobre la dirección y el puerto en los que ha levantado su endpoint para interactuar con la red.

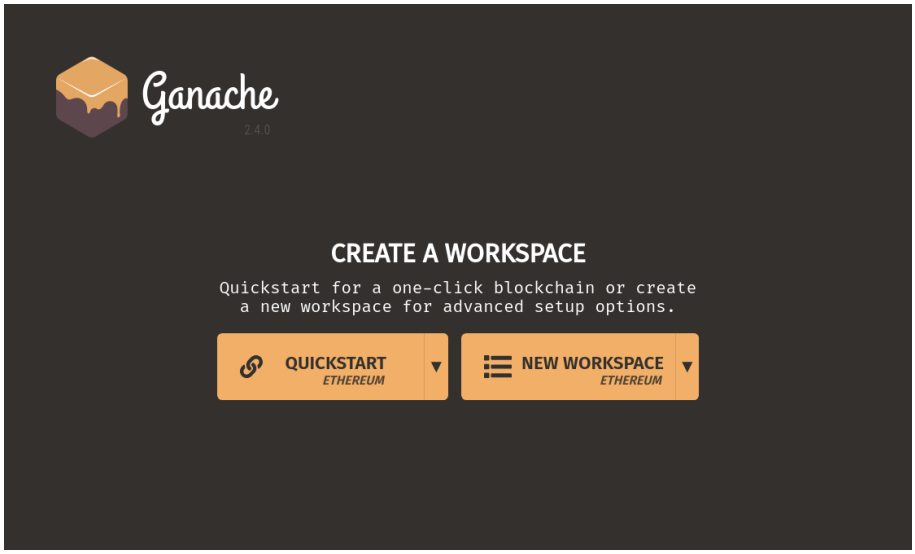

Figura 6.1: Pantalla de inicio Ganache

Durante las pruebas de este proyecto no usaremos las cuentas pregeneradas que proporciona Ganache ya que nuestro objetivo será ir generando DID a partir de cualquier cuenta dentro del namespace de Ethereum. Por este motivo para evitar tener que asignar cierta cantidad de balance a cada DID que vayamos generando para las pruebas hemos preferido establecer que el coste de transacción sea cero. Haciendo así una red de tipo Quorum.

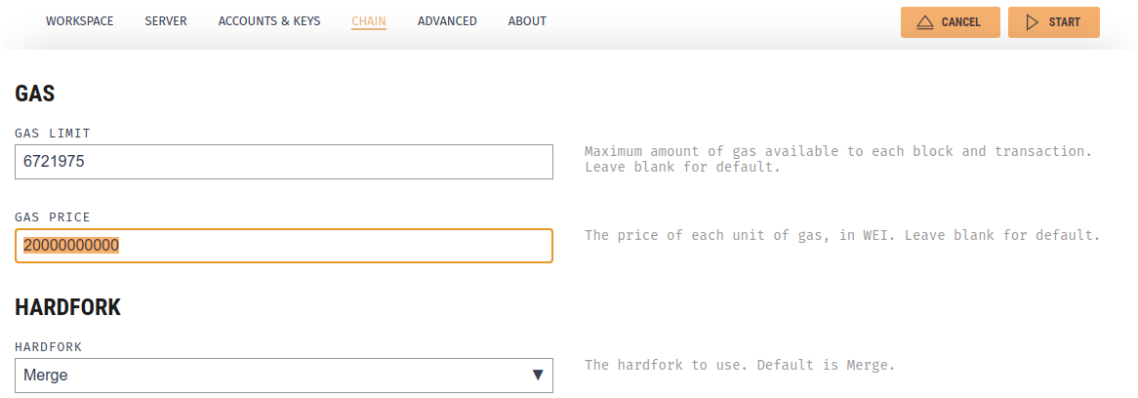

Figura 6.2: Ajustes en Ganache

Una vez tenemos la red, ahora lo que necesitamos es instalar Truffle e iniciar un proyecto. Un proyecto de Truffle está formado por un archivo 'truffle-config.js'. En este archivo se definen los contratos y las redes donde se van a desplegar. En nuestro caso usaremos una red local en el puerto predeterminado de tipo Quorum. Al utilizar quorum estamos indicando como hemos hecho en Ganache que la red no tendrá coste por transacción.

```
module.exports = {
  contracts_build_directory: "../client/src/contracts",
  networks: {
    development: {
      host: "127.0.0.1",
      port: 8545,
      network_id: "*",
      type: "quorum",
    },
  },
```

Figura 6.3: Contenido truffle-config.js.

Una vez definida la red a utilizar solo tendremos que ejecutar desde un terminal 'truffle compile'. Esto compilará los contratos especificados y generará los archivos de artefactos en el directorio 'build/contracts'. Estos archivos contienen información importante sobre los contratos que se desplegarán en la red.

Los archivos de artefactos generalmente están en formato JSON y contienen detalles sobre los contratos, como la dirección del contrato desplegado, las funciones y eventos disponibles, así como otros metadatos relevantes. Estos archivos se utilizan para interactuar con los contratos desde otras aplicaciones o scripts.

Finalmente, para migrar los contratos a la red Quorum local utilizaremos el comando 'truffle migrate'. Truffle ejecutará las migraciones necesarias y desplegará tus contratos

en la red local. Los detalles del despliegue, como las direcciones de contrato y las transacciones, se mostrarán en la terminal.

Una vez realizados estos pasos ya podremos comenzar a emitir transacciones y eventos dentro de la red. Una de las mejores funcionalidades de Ganache es el explorador de bloques que tiene integrado, con él se pueden ver desde la interfaz los eventos que van generándose y las transacciones que se registran. Como Ganache no tiene acceso a los artefactos del contrato no es capaz de descifrarlos, para solucionarlo solo tendremos que sincronizar el proyecto Truffle que habíamos realizado con Ganache desde los ajustes de este.

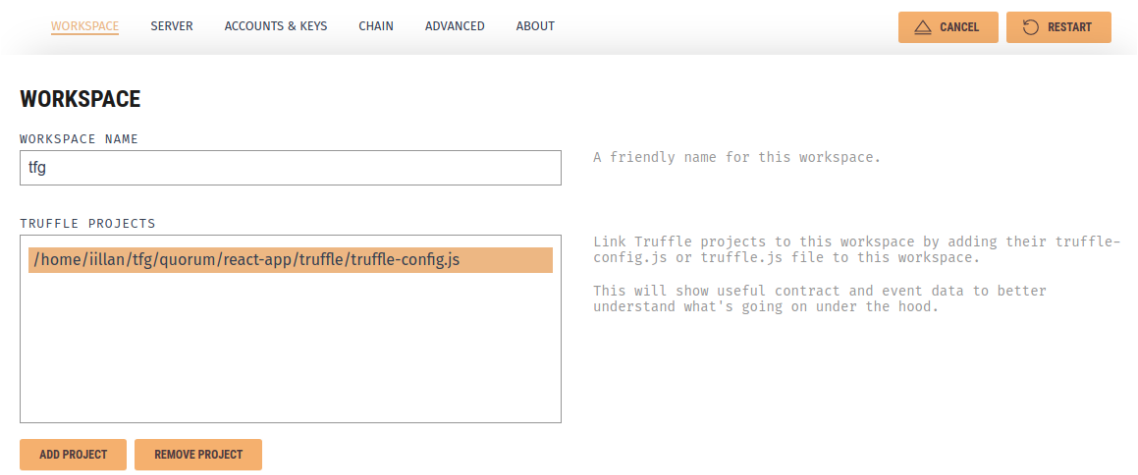

Figura 6.4: Sincronizar proyecto Truffle

6.2 Configuración con Veramo

Al estar el código desarrollado pensado para utilizarse como plugin para Veramo, la integración es realmente sencilla. Veramo utiliza un fichero para la configuración de los agentes dentro de un proyecto. El proyecto desarrollado exporta las siguientes clases: CredentialFlow, DataStore, DataStoreORM, IssueCredentialHandle y PresentProofHandler. Para integrar el plugin en Veramo, solo habrá que añadir las nuevas clases al archivo de configuración de Veramo y sustituir las ya existentes de Veramo con el mismo nombre.

```
0  export const veramoAgent = createAgent<VeramoAgent>({
1    plugins: [
2      new MessageHandler({
3        messageHandlers: [
4          new DIDCommMessageHandler(),
5          new IssueCredentialHandler(),
6          new PresentProofHandler(),
7        ],
8      }),
9      new DataStore(dbConnection),
10     new DataStoreORM(dbConnection),
11     new CredentialFlow(),
12   ],
13 });
```

Listado 6.1: Archivo de configuración de Veramo

CAPÍTULO 7
Pruebas

En este capítulo se probará el código desarrollado a través de un demostrador que en las siguientes secciones se presentará en formato de guía para que el lector que lo necesite pueda replicar los pasos. La demostración consistirá en unos scripts que mostrarán cómo funciona el sistema implementado y cómo iniciar la comunicación.

La demostración se despliega localmente y utiliza la blockchain de Ganache como registro de datos verificable. Por lo tanto, es necesario iniciar un nodo de Ganache en el dispositivo local. El agente de demostración se conecta a ws://localhost:8545, el cual es uno de los endpoints predeterminados cuando se inicia una red de Ganache.

Este demostrador genera y utiliza identificadores "did:ethr". El método DID de Ethereum agrega un segmento adicional al identificador para especificar la red en la que está desplegado el contrato inteligente. En el caso de trabajar en una red de pruebas local, los identificadores comenzarán con: "did:ethr:development".

Primero, se va a demostrar cómo un usuario puede adquirir una credencial, y luego se explicará el proceso de presentar y verificar esa credencial. Para los siguientes escenarios, serán necesarios dos agentes diferentes para cumplir con diferentes roles. Esto puede ser tan simple como clonar el repositorio y luego copiar el directorio generado en uno nuevo. Preste atención del directorio de trabajo donde se deben ejecutar los scripts. En cualquier momento, si se desea verificar la estructura completa del mensaje, se dispone de un script npm llamado "getMessage". Este script recibe el ID del mensaje como argumento y lo recupera de la base de datos SQLite utilizada por Veramo.

Finalmente, se presentarán una serie de escenarios adicionales que servirán para mostrar los mecanismos de seguridad que presenta el sistema desarrollado, basado en identidad autosoberana.

Para los siguientes escenarios se utilizarán los siguientes agentes: una universidad (U), tres estudiantes de la universidad U (A, B y C) y una empresa externa (E).

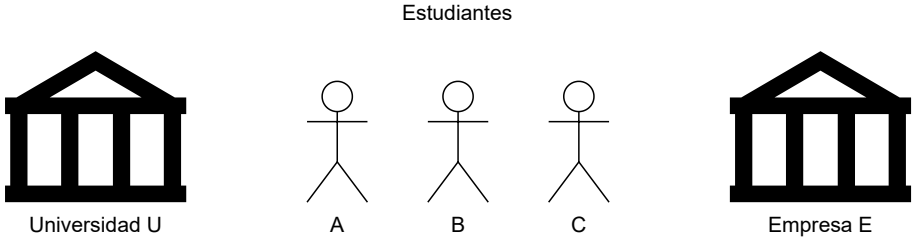

Figura 7.1: Agentes de los escenarios.

7.1 Escenario 1. Emisión de credenciales

El primer escenario que se va a presentar es el proceso que debe seguir el estudiante A para solicitar una credencial verificable de título universitario a la universidad U.

En esta sección se va a explicar primero los pasos necesarios para iniciar el proceso de comunicación entre dos agentes para el proceso de emisión de credenciales. Esta preparación no es específica para dicha interacción, sino que es un elemento común para cualquier agente que desee intercambiar cualquier tipo de mensaje con otro. Por este motivo solo se explicará una vez, aunque para el escenario de presentación de credenciales haya que repetir los pasos.

7.1.1. Preparación

Para enviar y recibir mensajes DIDComm, es necesario cumplir con los siguientes requisitos. El agente debe tener:

1. Un identificador descentralizado (DID).
2. Un enpoint de mensajería DIDComm definido en el documento de dicho DID.
3. Un enpoint funcional escuchando en la dirección especificada en el documento.

Para satisfacer las necesidades enumeradas anteriormente, habrá que seguir las siguientes secciones. Para cada una de estas tareas se ha preparado un script para poder explicar paso a paso. Aunque también se ha dejado el script "prepareDemo", el cual realiza los tres siguientes pasos de una vez, recibiendo como parámetro el puerto donde se desea publicar y levanta el endpoint.

Generar DID

Veramo permite registrar nuevos identificadores con un alias. En este caso, cada nuevo identificador se establece como predeterminado y el alias del DID predeterminado anterior se cambia a un UUID aleatorio. En cualquier momento se puede cambiar el DID predeterminado por otro, esto permite al usuario realizar una rotación de identificadores y poder utilizar diferentes DID para diferentes interacciones.

```
0 iillan@iillan-lap:~/code/veramo/credential-flow-plugin$ npm run addIdentifier
1
2 > veramo-agent@1.0.0 addIdentifier
3 > node dist/__tests__/addIdentifier.js
4
5 New identifier created: did:ethr:development:0
      x03fd13a4619968dbcef148d7c39de98773dfe7439d800eaa7e72dec85327493d68
```

Listado 7.1: Generación de un nuevo DID

Añadir el servicio de mensajería al documento DID

Como se ha mencionado antes para esta demostración, los endpoints se despliegan en la máquina local, por lo que el usuario solo deberá especificar el puerto deseado. El sistema funcionaría igual si se definieran URL públicas, pero para la realización de pruebas el despliegue actual es suficiente.

```
iillan@iillan-lap:~/code/veramo/credential-flow-plugin$ npm run addMessagingSvc
    9999

> veramo-agent@1.0.0 addMessagingSvc
> node dist/__tests__/addMessagingSvc.js "9999"

Added DIDCommMessaging Service on port: 9999
txHash:0xe7f46db78d44f26d599bed12ce7399d9b32353164ca17179b70e8fb786436498
```

Listado 7.2: Adición de un nuevo endpoint al documento DID

Iniciar la escucha del endpoint

Ahora es el momento de iniciar el endpoint y esperar nuevos mensajes. Para reproducir la demostración, se necesitarán dos agentes, lo que implica tener dos terminales para los endpoints, así como otros dos terminales adicionales para ejecutar los scripts.

```
iillan@iillan-lap:~/code/veramo/credential-flow-plugin$ npm run
    startMessagingSvc

> veramo-agent@1.0.0 startMessagingSvc
> node dist/__tests__/startMessagingSvc.js

DIDCommMessaging Service listening on port: 9999
```

Listado 7.3: Inicio del servicio de mensajería

7.1.2. Mensajes

Una vez se han cumplimentado los requisitos necesarios para el establecimiento de la comunicación ya se pueden iniciar las pruebas de los diferentes mensajes. Como ya se explicó en anteriores capítulos el estándar para la emisión de credenciales de Aries define que la comunicación puede comenzar con tres mensajes diferentes. Para la demostración se va a asumir la interacción más larga, lo que significa comenzar con una propuesta del futuro propietario (el estudiante A), al emisor de la credencial (la universidad U).

Propuesta

El estudiante puede que no sepa exactamente el nombre del esquema de la credencial de título universitario, por lo que genera una "credential_preview" con los atributos y valores deseados que le gustaría recibir, en este caso el título universitario y su DNI.

```
iillan@iillan-lap:~/holder$ npm run sendProposeVC did:ethr:development:0
    x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143

> veramo-agent@1.0.0 sendProposeVC
> node dist/__tests__/sendProposeCred.js "did:ethr:development:0
    x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143"

Sent Propose Credential: bb526ae7-f005-4350-9f8d-045649d008a2
```

Listado 7.4: Envío del mensaje propuesta de emisión

Oferta

Una vez que el estudiante envía el mensaje de propuesta, el terminal de la universidad devolverá una salida parecida al listado 7.5. La universidad verifica si puede emitir alguna credencial con los valores propuestos. El estudiante recibe una oferta con las posibles credenciales que podría solicitar a la universidad. En este ejemplo, el estudiante propuso atributos que coinciden con el esquema de credencial: "UniversityDegreeCredential". El estudiante recibe la oferta de credencial (Listado 7.6), esta aún no es una credencial verificable porque se envía sin el atributo de prueba ("proof"). Aunque no presente firma, la oferta especificará con qué tipo de algoritmo se firmará la futura credencial emitida.

```
// Issuer terminal
Received Message from: did:ethr:development:0
    x025862baf2cdb6937f9fef454da3da96fe925764b9b9362507f732d1bd09e9702f
Message type: https://didcomm.org/issue-credential/2.0/propose-credential
Propose Credential ID: bb526ae7-f005-4350-9f8d-045649d008a2
Proposed credential type: VerifiableCredential
Attributes from credential_preview: grado, dni
Compatible credentialSchemas: [{"type":"UniversityDegreeCredential","attributes
    ":"grado,dni"}]
Saved Message: 9dd200dc-3333-4306-bef6-1559f6c645b7
Sent Offer Credential: 9dd200dc-3333-4306-bef6-1559f6c645b7
```

Listado 7.5: Recepción del mensaje de propuesta de emisión

```
// Holder terminal
Received Message from: did:ethr:development:0
    x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143
Message type: https://didcomm.org/issue-credential/2.0/offer-credential
Offer Credential ID: 9dd200dc-3333-4306-bef6-1559f6c645b7
Credential offers~attach list: {
  "credential": {
    "issuer": "did:ethr:development:0
        x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143",
    "type": [
      "VerifiableCredential",
      "UniversityDegreeCredential"
    ],
    "credentialSubject": {
      "id": "did:ethr:development:0
          x025862baf2cdb6937f9fef454da3da96fe925764b9b9362507f732d1bd09e9702f",
      "grado": "gradoDefaultValue",
      "titulo": "tituloDefaultValue"
    }
  },
  "options": {
    "proofType": "EcdsaSecp256k1RecoverySignature2020"
  }
}
```

Listado 7.6: Recepción del mensaje de oferta de emisión

Solicitud

Ahora que el estudiante sabe cuál es el tipo credencial que necesita y lo puede solicitar. Cuando la universidad recibe el mensaje de solicitud de credencial, verifica si conoce el tipo de credencial. Esto puede parecer innecesario, ya que el emisor ya lo hizo al generar la oferta, pero debemos recordar que esto no siempre es así; el estudiante puede

iniciar el flujo de comunicación directamente con esta solicitud si ya conoce el tipo de credencial que necesita.

```
iillan@iillan-lap:~/holder$ npm run sendRequestVC did:ethr:development:0
    x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143
    UniversityDegreeCredential

> veramo-agent@1.0.0 sendRequestVC
> node dist/__tests__/sendRequestCred.js "did:ethr:development:0
    x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143" "
    UniversityDegreeCredential"

Sent Request Credential: 11f027ca-d0bc-4fb2-bb3a-6c5269cd6949
```

Listado 7.7: Envío del mensaje de solicitud de credencial

Emisión

Cuando la universidad recibe la solicitud de emisión del estudiante, además de verificar si soporta el tipo solicitado, deberá consultar en sus propios registros si el estudiante efectivamente se graduó. La universidad también debe consultar si ha emitido previamente esa credencial para dicho estudiante. Una vez que el estudiante ha recibido la credencial verificable solicitada, puede recuperarla del mensaje y verificarla. La credencial se almacena en la base de datos SQLite para su futuro uso.

```
// Holder terminal
Received Message from: did:ethr:development:0
    x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143
Message type: https://didcomm.org/issue-credential/2.0/issue-credential
Issue Credential ID: fb7c1e52-22e0-463b-af42-70d71dc16225
Verifiable Credential: ...
```

Listado 7.8: Recepción del mensaje de emisión de credencial

7.2 Escenario 2. Presentación de credenciales

Una vez obtenida una credencial, podemos continuar con el siguiente escenario. Ahora que el estudiante A ha obtenido la credencial verificable de título universitario y la tiene almacenada en su cartera, la presentará a la empresa E para demostrar que posee el grado. Este escenario será muy común en el futuro para procesos de solicitud de empleo o contratación de servicios.

Durante la presentación, se evaluará la validez de la credencial y se verificará la autenticidad del titular. Esto se realiza mediante el análisis de las firmas digitales y la verificación del esquema de la credencial. Si todo es correcto, el verificador (la empresa E) aceptará la presentación y considerará al propietario como legítimo para acceder a los servicios o recursos solicitados.

7.2.1. Preparación

Para este escenario, no es necesario preparar otro agente para la empresa ya que eso significaría volver a duplicar el repositorio e iniciar otro par de terminales. Aunque haya

tres roles diferentes, cada agente admite todos ellos, por lo que podemos reutilizar el despliegue del escenario anterior y utilizar los terminales de la universidad como si fueran la empresa E.

7.2.2. Mensajes

Igual que en el escenario para la emisión de las credenciales se va a asumir la interacción más larga para comprobar todos los mensajes. Aunque de normal será el verificador el encargado de iniciar la comunicación, el estándar soporta un mensaje de propuesta del propietario al verificador permitiéndole iniciar la comunicación. En caso de que el estudiante recibiera una solicitud de presentación y no estuviera concorde con esta, podría responder con un mensaje de propuesta para iniciar discusión entre ambos agentes.

Propuesta

De manera similar al paso de propuesta en el escenario de emisión, el estudiante no necesita saber qué esquemas admite el verificador para iniciar la comunicación. El propietario puede indicar el tipo de credencial que le gustaría utilizar sin conocimiento previo de los esquemas admitidos por el verificador.

```
0 iillan@iillan-lap:~/holder$ npm run sendProposeVP did:ethr:development:0
      x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143
      UniversityDegreeCredential
1 > veramo-agent@1.0.0 sendProposeVP
2 > node dist/__tests__/sendProposeVP.js "did:ethr:development:0
      x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143" "
      UniversityDegreeCredential"
3 Sent Request Presentation: 5bbc054b-db3c-44fc-928a-bb8e86980b7d
```

Listado 7.9: Envío del mensaje de propuesta de presentación

```
0 // Verifier terminal
1 Received Message from: did:ethr:development:0
      x025862baf2cdb6937f9fef454da3da96fe925764b9b9362507f732d1bd09e9702f
2 Message type: https://didcomm.org/present-proof/2.0/propose-presentation
3 Propose Presentation: 5bbc054b-db3c-44fc-928a-bb8e86980b7d
4 Credential type: UniversityDegreeCredential supported
5 Saved Message: 4c50d784-c9d9-4151-99ef-a8cc4075d441
6 Sent Request Presentation: 4c50d784-c9d9-4151-99ef-a8cc4075d441
```

Listado 7.10: Recepción del mensaje de propuesta de presentación

Solicitud

Una vez recibida la propuesta del estudiante, la empresa puede enviar un mensaje de solicitud a este. En dicha solicitud la empresa especificará un reto o "challenge", el cual el estudiante deberá utilizar para firmar la credencial verificable.

Presentación

El estudiante recibe la solicitud y genera la presentación verificable. Una vez generada, la envía a la empresa. Finalmente, el verificador inicia el proceso de verificación en el

```
// Holder terminal
Received Message from: did:ethr:development:0
    x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143
Message type: https://didcomm.org/present-proof/2.0/request-presentation
Request Presentation: 4c50d784-c9d9-4151-99ef-a8cc4075d441
Got credential for presentation
Generating verifiable presentation
Saved Message: 4a98be03-f5eb-45c1-a60b-e5a05b0ac17a
Sent Presentation: 4a98be03-f5eb-45c1-a60b-e5a05b0ac17a
```

Listado 7.11: Recepción del mensaje de solicitud de presentación

```
// Verifier terminal
Received Message from: did:ethr:development:0
    x025862baf2cdb6937f9fef454da3da96fe925764b9b9362507f732d1bd09e9702f
Message type: https://didcomm.org/present-proof/2.0/presentation
Presentation: 4a98be03-f5eb-45c1-a60b-e5a05b0ac17a
Found previous Request Presentation
Challenge: test_challenge
Verified Presentation: true
```

Listado 7.12: Recepción del mensaje de presentación

que debe comprobar tanto las firmas digitales (de la universidad y del estudiante) como el esquema de la credencial (Listado 7.12).

Reconocimiento

Tras esta etapa en la que se gestiona el mensaje "presentation" enviado por el estudiante y recibido por la empresa, esta debe responder al propietario. El mensaje a gestionar será un ack-presentation si la verificación ha tenido éxito o un report-problem si no ha sido así.

7.3 Escenario 3. Estudiante modifica la credencial

Ahora supongamos que el estudiante B está cursando un grado en la universidad y, por lo tanto, aún no puede solicitar su credencial de título universitario. La única credencial que podría solicitar en este momento sería una credencial de alumno, certificando que está cursando ciertas asignaturas del grado. Sin embargo, este estudiante quiere hacerse con una credencial de título universitario sin haber completado los requisitos para obtenerla.

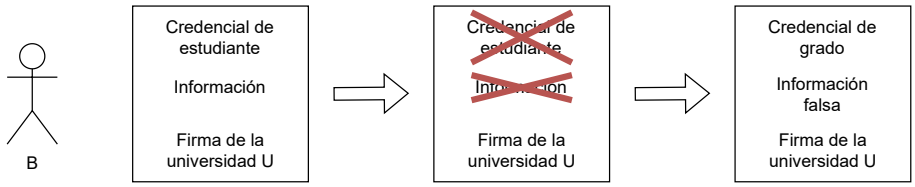

Figura 7.2: Estudiante B falsifica la información de la credencial.

Por este motivo, el estudiante B modifica su credencial de alumno para falsificar un título de una carrera que aún no ha obtenido. En este caso, el estudiante malintencionado cambia el esquema de la credencial y luego ajusta los atributos de la credencial para que coincidan con el esquema modificado. Sin embargo, este intento no tendría éxito, ya que

la firma de la credencial está vinculada al contenido original. Por lo tanto, cuando el estudiante B presente la credencial falsificada a una empresa, esta podrá comprobar que la firma que añadió la universidad a la credencial no se corresponde con el contenido modificado de la misma.

Incluso si el estudiante intentara generar otra firma para la credencial, sobrescribiendo así la de la universidad, seguiría sin tener éxito. Esto se debe a que cuando la empresa intentara verificar la firma usando alguna de las claves públicas de la universidad, no tendría éxito en la verificación.

7.4 Escenario 4. Estudiante suplanta a la universidad

En este escenario, el estudiante B quiere seguir intentando engañar el sistema. Por este motivo, su compañero, el estudiante C, se hace pasar por la universidad y le genera una credencial idéntica a la que esta generaría.

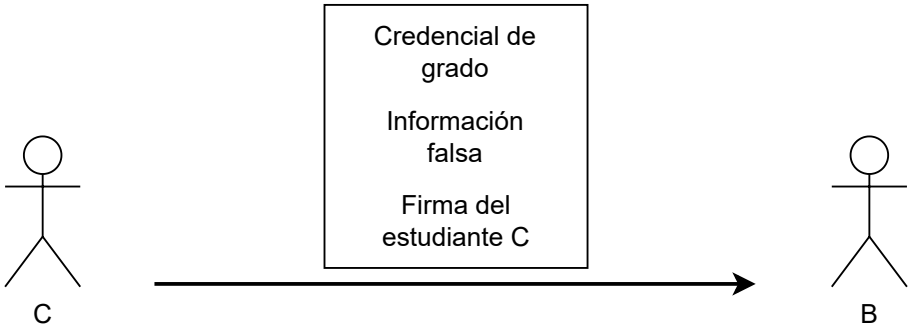

Figura 7.3: Estudiante C se hace pasar por una universidad.

Con esta nueva credencial, el estudiante B genera una presentación y se la envía a la empresa. En esta ocasión, cuando la empresa intente verificarla, no habrá ningún problema y el proceso resultará en que la información es válida y la firma es correcta.

Para solucionar este problema, la empresa tendría que consultar alguna lista de confianza, mantenida por alguna entidad con cierta autoridad, para comprobar que el DID que aparece como emisor dentro de la credencial es efectivamente de una universidad legítima y no un impostor.

7.5 Escenario 5. Estudiante intercepta una credencial

El siguiente intento del estudiante B para conseguir la credencial es un poco más complejo. En este escenario, el estudiante malicioso ha conseguido establecer un ataque de tipo man-in-the-middle entre algún otro estudiante y la universidad con el objetivo de interceptar algún mensaje de emisión de credencial.

Eventualmente otro estudiante inicia el proceso de solicitud y el estudiante B intercepta los mensajes. En el demostrador realizado los mensajes estaban firmados por los extremos, impidiendo que algún atacante modifique el mensaje. Como medida extra de seguridad, el estándar "DIDComm Messaging" permite además cifrar los mensajes con la clave pública de los receptores. Esto supondría que el estudiante B, aún que interceptara los mensajes no sería capaz de hacer nada con ellos.

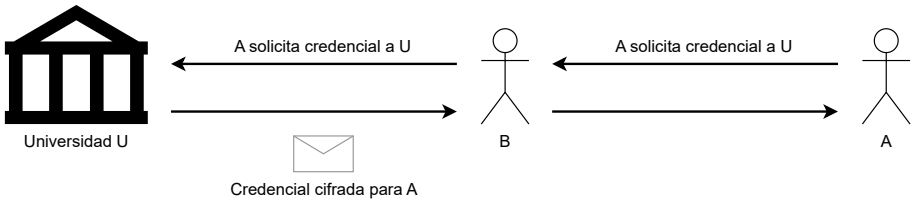

Figura 7.4: Estudiante B intercepta la credencial cifrada.

Aparte, en el improbable caso de que estos mensajes se compartieran en texto plano y que por lo tanto el estudiante malicioso obtuviera la credencial de otro estudiante estaría en la misma situación del escenario tres.

Ahora supongamos que, además de la credencial, el estudiante malintencionado ha interceptado una presentación firmada por otro estudiante legítimo. En un principio, podríamos pensar que el estudiante malicioso podría intentar suplantar la identidad de ese otro estudiante, utilizando dicha presentación para diversos procesos. Para evitar este problema en el proceso de solicitud de una presentación verificable, el verificador especificará diferentes desafíos o retos para cada ocasión. Esto implicará que se tengan que regenerar las presentaciones, pero nos asegurará la seguridad y evitará que se pueda suplantar la identidad del estudiante legítimo.

7.6 Escenario 6. Cartera de credenciales comprometida

En sus intentos por conseguir una credencial que no tiene, el estudiante B finalmente compromete la seguridad de la cartera digital del estudiante A y obtiene acceso a todos sus datos. Esto puede presentar dos casos diferentes.

Si la cartera solo almacenaba credenciales verificables, no habría un problema grave. El estudiante legítimo podría conseguir una nueva cartera y volver a solicitar sus credenciales. El mayor problema surge si el estudiante malicioso ha obtenido tanto las credenciales como los pares de claves públicas-privadas. En este caso, el atacante podría suplantar completamente la identidad de la víctima, no solo en el ámbito académico con la credencial de título universitario, sino también en otros aspectos de su vida donde utilice credenciales verificables.

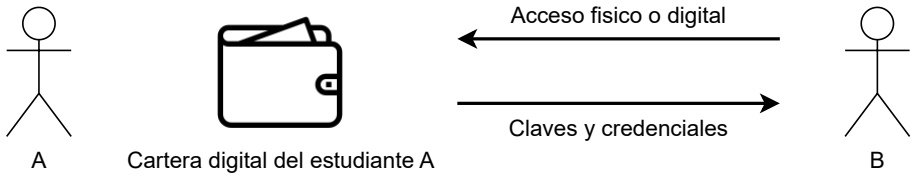

Figura 7.5: Estudiante B consigue acceso a toda la información confidencial de A.

Este último escenario es el más peligroso que se podría enfrentar. Por este motivo, las carteras digitales se convierten en uno de los principales pilares para la seguridad de los usuarios. Para estos casos, sería necesario establecer mecanismos para la revocación de los DID y las credenciales comprometidas. Esto permitiría invalidar las credenciales afectadas y proteger la identidad del usuario legítimo.

Es importante que los sistemas de identidad autosoberana incluyan mecanismos sólidos de seguridad, como el uso de contraseñas fuertes, autenticación de dos factores y cifrado de datos, para evitar que los atacantes comprometan la seguridad de los usua-

rios y proteger la integridad de las credenciales verificables almacenadas en las carteras digitales.

7.7 Escenario 7. Uso de los servicios de la universidad

En este último escenario el estudiante A todavía está cursando asignaturas en alguna de las titulaciones de la unviersidad U. Se asumirá que el uso de identidad autosoberana ya es común y que para utilizar la mayoría de los servicios informáticos pueden utilizarse DID y VC. Ese escenario será posible en un futuro cercano, pero no lo es actualmente.

Al matricularse A en algunas asignaturas de la titulación T en la universidad U, obtendrá las credenciales pertinentes que acrediten ese hecho para el curso actual. Para acceder a la intranet de U, el agente verificador pertinente solo solicitará alguna credencial que demuestre la vinculación del usuario con U. A tiene ese tipo de credencial, por lo que podrá utilizarla para superar esa comprobación y acceder a la intranet. Algo similar utilizarán los verificadores para gestionar el acceso a los laboratorios de las diferentes asignaturas en las que se ha matriculado A. Con ello, ya no será necesario que A recuerde y proporcione su identificador y contraseña cada vez que quiera acceder a la intranet o los laboratorios de U. En su lugar, bastará con utilizar las credenciales pertinentes desde su cartera digital. Esto no es solo conveniente para A, sino que también simplificará la administración de los laboratorios docentes en U.

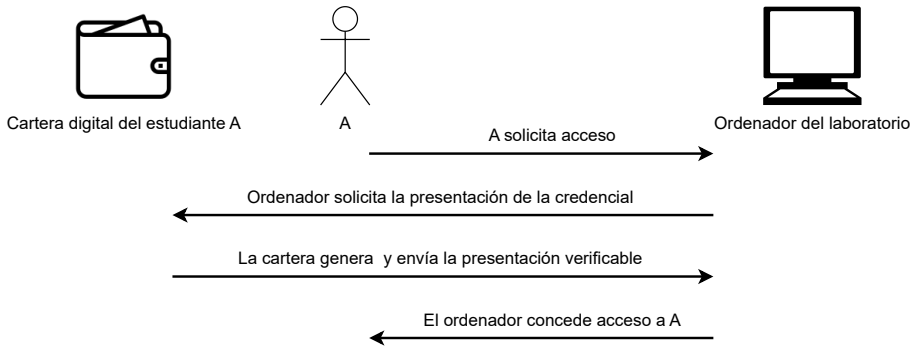

Figura 7.6: A accede a los ordenadores del laboratorio de una asignatura.

Por otra parte, con esta gestión U podrá controlar el acceso a sus laboratorios con una granularidad de asignatura. Si A está matriculado en la asignatura X, pero no en la Y, no podrá acceder presencialmente al laboratorio donde se estén impartiendo prácticas de Y, pues no poseería la credencial necesaria para ello. Ese tipo de control no resulta sencillo actualmente, con sistemas de gestión de identidad federada.

7.8 Conclusión

En esta sección de pruebas, se ha demostrado el flujo completo de interacciones para la emisión y presentación de credenciales verificables utilizando la implementación realizada de los estándares de Aries sobre el framework de Veramo. Se ha expuesto cómo un agente puede adquirir una credencial, almacenarla en su billetera y luego presentarla a un verificador.

Si bien hemos realizado pruebas exitosas en un entorno local, es importante mencionar que en lugar de utilizar identificadores en una red local también se podría realizar la

integración con redes y soluciones de blockchain públicas, lo que amplía sus capacidades para escenarios de producción y despliegues más amplios.

Finalmente, se han presentado una serie de escenarios que demuestran los mecanismos de seguridad presentes en las credenciales verificables y los DID frente a diversos tipos de ataques, además de resaltar los beneficios que ofrecen en el contexto de la autenticación de estudiantes en su universidad.

CAPÍTULO 8
Conclusiones

Este trabajo reúne una investigación de varios meses sobre el estado actual de la identidad autosoberana, presentando y explicando las diferentes organizaciones involucradas y sus estándares, así como los proyectos más relevantes de la comunidad. Se ha llevado a cabo un análisis y una comparación exhaustiva de cada uno de los componentes de la identidad autosoberana, centrándose en los diferentes protocolos para la emisión y presentación de las credenciales verificables. Este análisis abarca todos los intentos de estandarización de estos procesos por parte de las diferentes organizaciones que trabajan en el tema, y es valioso considerando que es el único realizado hasta la fecha.

Al comenzar este TFG, no se esperaba tener que consultar y leer una lista tan numerosa de especificaciones. Aunque esta tarea podría considerarse tediosa, ha sido de gran utilidad para aprender sobre una amplia variedad de campos, como criptografía, sistemas distribuidos y seguridad. Durante este trabajo, se ha valorado la importancia de realizar una investigación seria antes de cualquier desarrollo ya que, sin conocer y comparar las soluciones en desarrollo, es imposible proponer algo nuevo.

El trabajo realizado ha servido para comprender el estado actual y el futuro de la identidad autosoberana. Se ha observado el creciente apoyo que ha surgido en los últimos años, así como la relevancia que este nuevo paradigma está adquiriendo para la gestión de identidades. Tal es su impacto que incluso la Comisión Europea ha decidido utilizarla.

Con respecto a los estándares que respaldan esta nueva idea, aún les falta alcanzar cierto grado de madurez. Actualmente, la mayoría de los estándares se encuentran en desarrollo o en borradores de su primera versión. Hasta que no alcancen una versión estable, no podrán aparecer implementaciones relevantes. Y hasta que ese momento llegue, no se podrán plantear verdaderas soluciones de interoperabilidad entre las diferentes soluciones que existirán en el futuro.

Después de trabajar durante un tiempo con el framework de Veramo, se puede apreciar el valor que aporta. Veramo proporciona una base sobre la cual se pueden implementar agentes más específicos o complejos. Las funcionalidades que ya están implementadas tienen un gran valor, ya que desarrollar todo desde cero requeriría el trabajo de un equipo durante años. Sin embargo, una de las principales desventajas de Veramo es su falta de documentación. Aunque hay algunas guías sobre las funciones más básicas, son insuficientes para un desarrollo más complejo.

Sin embargo, esto no ha sido algo negativo al final, ya que ha obligado a investigar en diferentes repositorios y consultar directamente el código cada vez que surgía una duda o problema. Esto ha permitido obtener un conocimiento en profundidad del framework, hasta el punto de ser capaz de detectar varios fallos y avisar sobre ellos.

Por otro lado, al decidir implementar uno de los estándares existentes en la comunidad, se ha llevado a cabo un trabajo completamente diferente al resto de proyectos de grado y máster relacionados. En lugar de utilizar una solución de identidad autosoberana ya existente con una aplicación web o un demostrador con tecnologías de la comunidad, se ha creado una solución que previamente no existía en la comunidad. Esta solución implementada podrá servir al resto de desarrolladores del proyecto de Aries como una referencia única. Además, ha permitido obtener una mayor familiaridad con las tecnologías, ya que no es lo mismo utilizarlas que desarrollarlas.

8.1 Relación con los estudios cursados

Ahora se comentarán las asignaturas del grado que han servido como base para el desarrollo de este trabajo. Una de las tareas más importantes en la realización de un trabajo de este calibre es establecer un plan de trabajo adecuado, en el cual se especifiquen sus etapas y la respectiva división en tareas, así como los plazos para cada una de ellas. Luego, se procede a calcular el coste total del trabajo basándose en las horas-hombre dedicadas. Estas tareas fueron explicadas en la asignatura de Gestión de Proyectos (GPR), cursada en el tercer curso.

Durante la etapa de implementación de este TFG, surgieron algunos problemas en el manejo de mensajes asíncronos. Por este motivo, fue necesario revisar la teoría de algunas asignaturas como "Tecnología de Sistemas de Información en la Red" (TSR), también estudiada en el tercer curso.

Una de las asignaturas que ha tenido mayor relación con este proyecto ha sido la optativa de Criptografía del cuarto curso. Al principio de esta asignatura, se abordaron los conceptos de sistemas basados en claves secretas, públicas y mixtas, así como los algoritmos de digestión o hash. Todos estos conceptos se han profundizado al trabajar directamente con ellos, especialmente al manejar las diferentes claves en la cartera digital.

Otra asignatura que ha sido de gran utilidad ha sido "Ingeniería del Software" de tercero para la etapa de diseño, para la cual se ha tenido que realizar el diagrama de clases. También se ha tenido que recordar cierta teoría durante el desarrollo sobre la división del código en diferentes capas en función de sus objetivos.

CAPÍTULO 9
Trabajo futuro

Ahora vamos a comentar los posibles caminos que se podrían seguir para mejorar este proyecto. Como se ha visto, un sistema de identidad autosoberana completo tiene muchos aspectos a tener en cuenta. Nos hemos centrado en los mecanismos de comunicación entre agentes para la emisión y presentación de credenciales, pero hay otras áreas que podrían ser consideradas.

En cuanto a los esquemas de las credenciales, es un asunto relevante pero aún no ha habido avances significativos en la comunidad. En este trabajo, hemos utilizado la wallet como registro para los esquemas, lo que significa que cada agente almacena y conoce los esquemas que soporta localmente. Sin embargo, sería útil que un propietario pudiera consultar qué esquemas son soportados tanto por el emisor como por el verificador sin tener que comunicarse directamente con ellos. Una opción sería que cada entidad publique una lista en su página web. Sin embargo, esta solución no es suficiente, ya que las páginas web son vulnerables a ataques. Por lo tanto, la opción más acertada sería registrar los esquemas y sus actualizaciones en la propia blockchain, junto con los identificadores y sus documentos. Esto permitiría tener un listado inalterable y altamente disponible.

Para lograr esta funcionalidad, se debería desarrollar un nuevo contrato inteligente que permita las operaciones CRUD para los esquemas. Además, se debería considerar qué formatos se utilizarán para dichos esquemas. Actualmente, el W3C está en proceso de generar una recomendación, pero por ahora es solo un borrador temprano. EBSI ya utiliza ejemplos de esquemas de credenciales en sus pilotos, los cuales están publicados en su repositorio. También tienen documentación sobre la API del contrato inteligente para los esquemas. En sus ejemplos, utilizan credenciales en formato JSON-LD, que parece ser el formato más compatible. Sin embargo, es importante destacar que el código de este contrato no está disponible públicamente a fecha de julio de 2023.

Otra opción es agregar más mecanismos de control a las credenciales verificables. Algunas credenciales no necesitan una relación constante entre el emisor y el propietario, como un certificado de graduación. Una vez emitida dicha credencial, no debería cambiar a lo largo de los años. Sin embargo, las credenciales también pueden representar membresías, y en este caso, el emisor debe poder revocar la credencial emitida si el propietario se da de baja o es expulsado. En este aspecto, se plantea agregar otro contrato inteligente al sistema donde se puedan registrar las credenciales emitidas para revocarlas si fuera necesario. Esto podría aumentar el tiempo necesario para verificar las credenciales.

Además, se podría considerar mejorar la seguridad de la wallet. Actualmente, la cartera está implementada con una base de datos SQLite local que almacena toda la información necesaria para el usuario en tablas. Sin embargo, para que pueda ser realmente una cartera digital segura, se deben implementar mecanismos de seguridad adicionales para evitar posibles ataques de suplantación de identidad o filtración de información.

Bibliografía

[1] Alastria Digital Identity. An ongoing project. Alastria Presentation, septiembre 2019. Disponible en: https://portal.r2docuo.com/alastria/document?L3110FC15F. Accedido en agosto de 2023.

[2] Alastria DID Method Specification. Alastria GitHub Repository, marzo 2023. Disponible en: https://github.com/alastria/alastria-identity/wiki/Alastria-DID-Method-Specification. Accedido en agosto de 2023.

[3] AEPD. Derecho de supresión "al olvido": buscadores de internet, 2022. Disponible en: https://www.aepd.es/es/areas-de-actuacion/internet-y-redes-sociales/derecho-al-olvido, Accedido mayo de 2023.

[4] Christopher Allen. The path to self-sovereign identity. Life with Alacrity blog, abril 2016. Disponible en: http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html. Accedido en mayo de 2023.

[5] BBC. Facebook's data-sharing deals exposed. https://www.bbc.com/news/technology-46618582, 2018. Accedido 31-01-2023.

[6] Daniel Buchner, Brent Zundel, Jace Hensley, Daniel McGrogan, Gabe Cohen, and Kim Hamilton Duffy. Credential Manifest 1.x Editor's Draft. Decentralized Identity Foundation Working-Group Approved Draft, mayo 2023. Disponible en: https://identity.foundation/credential-manifest/. Accedido en julio de 2023.

[7] Daniel Buchner, Brent Zundel, Martin Riedel, and Kim Hamilton Duffy. Presentation Exchange 2.0.0. Decentralized Identity Foundation Ratified Specification, marzo 2023. Disponible en: https://identity.foundation/presentation-exchange/spec/v2.0.0/. Accedido en julio de 2023.

[8] Daniel Buchner, Brent Zundel, Martin Riedel, and Kim Hamilton Duffy. Presentation Exchange 2.X.X. Decentralized Identity Foundation Pre-Draft, marzo 2023. Disponible en: https://identity.foundation/presentation-exchange/. Accedido en julio de 2023.

[9] Kim Cameron. The laws of identity. *Microsoft Corp*, 12:8–11, 2005.

[10] Chadwick, Kenichi Nakamura, Vercammen, and Jo. OpenID for Verifiable Credentials. *The OpenID Foundation*, junio 2022. Disponible en: https://openid.net/wordpress-content/uploads/2022/06/OIDF-Whitepaper_OpenID-for-Verifiable-Credentials-V2_2022-06-23.pdf. Accedido en julio de 2023.

[11] Stephen Curran, Artur Philipp, Hakan Yildiz, Sam Curren, and Victor Martinez Jurado. AnonCreds Specification v1.0. Hyperledger Foundation Draft, 2022. Disponible en: https://hyperledger.github.io/anoncreds-spec/. Accedido en junio de 2023.

[12] Sam Curren, Tobias Looker, and Oliver Terbu. DIDComm Messaging v2.x Editor's Draft. Disponible en: https://identity.foundation/didcomm-messaging/spec/. Accedido junio 2023, 2023.

[13] Descentralized Identity Foundation (DIF). Universal Resolver - resolve practically any DID, septiembre 2022. Disponible en: https://blog.identity.foundation/uni-resolver/. Accedido en julio de 2023.

[14] DIF. Decentralized Identity Foundation, mayo 2017. Disponible en: https://identity.foundation/. Accedido en mayo de 2023.

[15] EBSI. EBSI DID Method. https://ec.europa.eu/digital-building-blocks/wikis/display/EBSIDOC/EBSI+DID+Method, 2021.

[16] European Parliament and Council of the European Union. On the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). Regulation (EU) 2016/679 of the European Parliament and of the Council, abril 2016. Disponible en: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679. Accedido en junio de 2023.

[17] Jacobo Fiaño Rodríguez. Sistema de Identidad Digital Soberana y Descentralizada basada en Blockchain. TFG. Universidade da Coruña. Facultade de Informática, 2022. Disponible en: http://hdl.handle.net/2183/32059. Accedido en julio de 2023.

[18] George Aristy . Aries RFC 0510: Presentation-Exchange Attachment format for requesting and presenting proofs. Hyperledger Aries Request for Comment, julio 2020. Disponible en: https://github.com/hyperledger/aries-rfcs/blob/main/features/0510-dif-pres-exch-attach/README.md#propose-presentation-attachment-format. Accedido en julio de 2023.

[19] Daniel Hardman. Aries RFC 0005: DID Communication. https://github.com/hyperledger/aries-rfcs/tree/main/concepts/0005-didcomm, 2019.

[20] Martin E Hellman. An overview of public key cryptography. *IEEE Communications Magazine*, 40(5):42–49, 2002. Disponible en: https://netlab.ulusofona.pt/im/teoricas/OverviewPublicKeyCryptography.pdf. Accedido en agosto de 2023.

[21] Michael B. Jones, John Bradley, and Nat Sakimura. JSON Web Signature (JWS). RFC 7515, May 2015. Disponible en: https://datatracker.ietf.org/doc/html/rfc7515. Accedido en agosto de 2023.

[22] Michael B. Jones, John Bradley, and Nat Sakimura. JSON Web Token (JWT). RFC 7519, May 2015. Disponible en: https://datatracker.ietf.org/doc/html/rfc7519. Accedido en agosto de 2023.

[23] Xuejia Lai and James L. Massaey. A Proposal for a New Block Encryption Standard. *Advances in Cryptology — EUROCRYPT '90. Lecture Notes in Computer Science. Vol. 473. pp. 389–404*, 1991. ISBN 978-3-540-53587-4.

[24] Nan Li. Research on Diffie-Hellman key exchange protocol. *2010 2nd International Conference on Computer Engineering and Technology*, 4:V4–634. Disponible en: https://ieeexplore.ieee.org/abstract/document/5485276?casa_token=0yUiwVvR7CYAAAAA:eGeO85WhKl7gw8t1w_OnNJPJ1ptvHSBcU2aH0rXyZUHtmexnNEDUJ0E-L4CaG61P1Nr39yOomVg. Accedido en agosto de 2023.

[25] T. Lodderstedt, K. Yasuda, and T. Looker. OpenID Connect for Verifiable Credential Issuance, abril 2022. Disponible en: https://openid.net/specs/openid-connect-4-verifiable-credential-issuance-1_0-05.html. Accedido en agosto de 2023.

[26] George Loladze. Notarización de documentos con Ethereum. TFG. Universitat Politècnica de València, 2020. Disponible en: https://riunet.upv.es/bitstream/handle/10251/151565/Loladze%20-%20Notarizaci%C3%B3n%20de%20documentos%20con%20Ethereum.pdf?sequence=1. Accedido en julio de 2023.

[27] Christian Lundkvist, Rouven Heck, Joel Torstensson, Zac Mitton, and Michael Sena. Uport: A platform for self-sovereign identity. 2017. Disponible en: https://whitepaper.uport.me/uPort_whitepaper_DRAFT20170221. Accedido junio 2023.

[28] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008. Disponible en: https://bitcoin.org/bitcoin.pdf. Accedido junio 2023.

[29] Nikita Khateev, Stephen Curran. Aries RFC 0454: Present Proof Protocol 2.0. Hyperledger Aries Request for Comment, abril 2020. Disponible en: https://github.com/hyperledger/aries-rfcs/tree/main/features/0454-present-proof-v2. Accedido en julio de 2023.

[30] Nikita Khateev, Stephen Klump, Stephen Curran. Aries RFC 0453: Issue Credential Protocol 2.0. Hyperledger Aries Request for Comment, marzo 2020. Disponible en: https://github.com/hyperledger/aries-rfcs/tree/main/features/0453-issue-credential-v2. Accedido en julio de 2023.

[31] National Institute of Standards and Technology. Advanced Encryption Standard (AES). *Federal Information Processing Standards Publication 197, Department of Commerce*, noviembre 2001. Disponible en: https://csrc.nist.gov/pubs/fips/197/final. Accedido en agosto de 2023.

[32] Pelle Braendgaard, Joel Torstensson. ERC-1056: Ethereum Lightweight Identity. Ethereum Improvement Proposals, mayo 2018. Disponible en: https://eips.ethereum.org/EIPS/eip-1056. Accedido en julio de 2023.

[33] F. Javier Fernández-Bravo Peñuela, Jordi Arjona Aroca, Francesc D. Muñoz-Escoí, Yuriy Yatsyk Gravrylyak, Ismael Illán García, and José M. Bernabéu-Aubán. DELTA: DLT-Database Synchronization. In *2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–9, 2023.

[34] Bart Preneel. Cryptographic hash functions. *European Transactions on Telecommunications*, 5(4):431–448, 1994. Disponible en: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.4460050406. Accedido en agosto de 2023.

[35] Antonio Ruiz García. Desarrollo de una aplicación web del modelo de identidad soberana Alastria ID. TFG. ETS de Ingeniería Informática de la UPM, junio 2022. Disponible en: https://oa.upm.es/71321/. Accedido en agosto de 2023.

[36] María Ruiz Molina. Identidad descentralizada aplicada. TFM. Universitat Oberta de Catalunya (UOC), 2023. Disponible en: https://openaccess.uoc.edu/bitstream/10609/148138/1/mruizmolina0TFM0623memoria.pdf. Accedido en julio de 2023.

[37] Natsuhiko Sakimura, John Bradley, Mike Jones, Breno De Medeiros, and Chuck Mortimore. OpenID Connect Core 1.0. *The OpenID Foundation*, 2014. Disponible en: https://openid.net/specs/openid-connect-core-1_0-final.html. Accedido en julio de 2023.

[38] Pablo Santos-Cabaleiro. Análisis y prototipado de Identidad Digital Descentralizada basada en Blockchain. TFG. Universidade da Coruña. Facultade de Informática, 2022. Disponible en: http://hdl.handle.net/2183/32090. Accedido en julio de 2023.

[39] Guillermo Sanz González. Diseño e implementación de un sistema de identidad digital descentralizada para ciudadanos de la Unión Europea en el ámbito sanitario. TFM. Universidad Politécnica de Madrid. ETSI Telecomunicacion, 2023. Disponible en: https://oa.upm.es/72965/. Accedido en julio de 2023.

[40] Selvanathan, Nirojan, Jayakody, Dileepa, Damjanovic-Behrendt, and Violeta. Federated identity management and interoperability for heterogeneous cloud platform ecosystems. Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES), 2019.

[41] Simon SY Shim, Geetanjali Bhalla, and Vishnu Pendyala. Federated identity management. *Computer*, 38(12):120–122, 2005.

[42] Manu Sporny, Dave Longley, and David Chadwick. Verifiable Credentials Data Model 1.0. Expressing verifiable information on the Web. World Wide Web Consortium Recommendation, noviembre 2019. Disponible en: https://www.w3.org/TR/2019/REC-vc-data-model-20191119/. Accedido en mayo de 2023.

[43] Manu Sporny, Dave Longley, and David Chadwick. Verifiable credentials data model v1.1. World Wide Web Consortium Recommendation, marzo 2022. Disponible en: https://www.w3.org/TR/2022/REC-vc-data-model-20220303/. Accedido en mayo de 2023.

[44] Manu Sporny, Dave Longley, and David Chadwick. Verifiable credentials data model v2.0. World Wide Web Consortium Working Draft, mayo 2023. Disponible en: https://www.w3.org/TR/vc-data-model-2.0/. Accedido en mayo de 2023.

[45] Manu Sporny, Dave Longley, Markus Sabadello, Drummond Reed, Orie Steele, and Christopher Allen. Decentralized Identifiers (DIDs) v1.0. Core architecture, data model, and representations. World Wide Web Consortium Recommendation, julio 2022. Disponible en: https://www.w3.org/TR/did-core/. Accedido en mayo de 2023.

[46] Manu Sporny and Orie Steele. DID Specification Registries. World Wide Web Consortium Group Note, mayo 2023. Disponible en: https://www.w3.org/TR/did-spec-registries/#did-methods. Accedido en junio de 2023.

[47] Phillip J. Windley. *Learning Digital Identity. Design, Deploy, and Manage Identity Architectures*. O'Reilly Media, Inc., enero 2023. ISBN 978-1-098-11769-6.

[48] Dylan Yaga, Peter Mell, Nik Roby, and Karen Scarfone. Blockchain technology overview. *arXiv preprint arXiv:1906.11078*, 2019. Disponible en: https://arxiv.org/abs/1906.11078. Accedido en agosto de 2023.

[49] Hakan Yildiz, Axel Küpper, Dirk Thatmann, Sebastian Göndör, and Patrick Herbke. A Tutorial on the Interoperability of Self-sovereign Identities, 2022. Disponible en: https://doi.org/10.48550/arXiv.2208.04692. Accedido en mayo de 2023.

Glosario

DID efímero Tipo de identificador que pretende ser utilizado en relaciones de breve duración. 32

endpoint Dirección de una API, o bien un backend que se encarga de dar respuesta a una petición. 27

framework Entorno o marco de trabajo, un conjunto de prácticas, conceptos y criterios a seguir estandarizados. 39

identidad autosoberana Concepto de una identidad digital que un individuo gestiona de forma descentralizada, sin que terceros almacenen sus datos. 1

inicio de sesión único Esquema de autenticación que permite a los usuarios iniciar sesión una vez, y obtener acceso seguro a varios servicios y aplicaciones relacionadas durante esa sesión sin necesidad de registrarse otra vez. 1

man-in-the-middle Ataque de intermediario. Ataque en el que se adquiere la capacidad de leer, insertar y modificar a voluntad. 68

nonce Número arbitrario que se puede usar una única vez en una comunicación criptográfica. A menudo es un número aleatorio. 18

Acrónimos

AIJU Instituto Tecnológico de Producto Infantil y Ocio. 2

API Application Programming Interface - Interfaz de Programación de Aplicaciones. 34

DID Decentralized Identifier - Identificador descentralizado. 2

DIF Decentralized Identity Foundation. 12, 30

DNI Documento Nacional de Identidad. 13

FEDER Fondo Europeo de Desarrollo Regional. 2

HTML HyperText Markup Language - Lenguaje de Etiquetas de Hipertexto. 20

ISO International Organization for Standardization - Organización Internacional de Normalización. 20

ITI Instituto Tecnológico de Informática. 2

IVACE Instituto Valenciano de Competitividad Empresarial. 2

P2P Peer-to-peer. 12

SDK Software Development Kit - Kit de Desarrollo de Software. 24

SSI Self-sovereign identity - Identidad Autosoberana. 8

TCP/IP Transmission Control Protocol/Internet Protocol - Protocolo de Control de Transmisión/Protocolo de Internet. 5

VC Verifiable Credential - Credencial Verificable. 13

VP Verifiable Presentation - Presentación Verificable. 14

W3C World Wide Web Consortium - Consorcio WWW. 2

APÉNDICE A
Ejemplos de mensajes

```
{
  "id": "fb7c1e52-22e0-463b-af42-70d71dc16225",
  "type": "https://didcomm.org/issue-credential/2.0/issue-credential",
  "raw": ... ,
  "data": {
    "@type": "https://didcomm.org/issue-credential/2.0/issue-credential",
    "@id": "fb7c1e52-22e0-463b-af42-70d71dc16225",
    "comment": "This is the credential you requested",
    "replacement_id": "abc589uy-f0cb-44bd-89fd-34333j660b12"
    "formats": [
      {
        "attach_id": "eaf9cf8b-f0cb-44bd-89fd-23222f559a01",
        "format": "aries/ld-proof-vc@v1.0"
      }
    ],
    "credentials~attach": [
      {
        "@id": "eaf9cf8b-f0cb-44bd-89fd-23222f559a01",
        "mime-type": "application/ld+json",
        "data": { verificable credential }
      }
    ]
  },
  "from": "did:ethr:development:0x02a7b4984ecb19b7c5a2986735e5d51924a9767323f5d0e37516dd256e87420143",
  "to": "did:ethr:development:0x025862baf2cdb6937f9fef454da3da96fe925764b9b9362507f732d1bd09e9702f"
}
```

Listado A.1: Contenido de un mensaje "issue-credential"

```
{
  "id": "bb526ae7-f005-4350-9f8d-045649d008a2",
  "type": "https://didcomm.org/issue-credential/2.0/propose-
     credential",
  "raw": ... ,
  "data": {
    "@type": "https://didcomm.org/issue-credential/2.0/propose
       -credential",
    "@id": "bb526ae7-f005-4350-9f8d-045649d008a2",
    "credential_preview": ... ,
    "formats": [
      {
        "attach_id": "1d7c61e8-ec5c-43ad-b229-a7fe611f5f60",
        "format": "aries/ld-proof-vc-detail@v1.0"
      }
    ],
    "filters~attach": [
      {
        "@id": "1d7c61e8-ec5c-43ad-b229-a7fe611f5f60",
        "mime-type": "application/json",
        "data": {
          "credential": {
            "type": ["VerifiableCredential"],
            "issuer": "did:ethr:development:0x02a7b4984ecb19b7
               c5a2986735e5d51924a9767323f5d0e37516dd256e87420
               143",
            "credentialSubject": {
              "id": "did:ethr:development:0x025862baf2cdb6937f
                 9fef454da3da96fe925764b9b9362507f732d1bd09e97
                 02f"
            }
          },
          "options": {
            "proofType": "EcdsaSecp256k1RecoverySignature2020"
          }
        }
      }
    ]
  },
  "from": "did:ethr:development:0x025862baf2cdb6937f9fef454da3
     da96fe925764b9b9362507f732d1bd09e9702f",
  "to": "did:ethr:development:0x02a7b4984ecb19b7c5a2986735e5d5
     1924a9767323f5d0e37516dd256e87420143"
}
```

Listado A.2: Contenido de un mensaje "propose-credential"

```
[{
  "@id": "3b800dc8-d8a3-48e8-8f48-4f1b2bc7aca8",
  "mime-type": "application/json",
  "data": {
    "credential": {
      "issuer": "did:ethr:development:0x02a7b4984ecb19b7c5a298
         6735e5d51924a9767323f5d0e37516dd256e87420143",
      "type": [
      "VerifiableCredential",
      "UniversityDegreeCredential"
      ],
      "credentialSubject": {
        "id": "did:ethr:development:0x025862baf2cdb6937f9fef45
           4da3da96fe925764b9b9362507f732d1bd09e9702f",
        "grado": "Universitario",
        "titulo": "Ingenieria Informatica",
        "dni": "20085452C",
        "curso": "2021-2022"
      }
    },
    "options": {
      "proofType": "EcdsaSecp256k1RecoverySignature2020"
    }
  }
}
{
  "@id": "4c911dc8-d8a3-48e8-8f48-4f1b2bc8bdb9",
  "mime-type": "application/json",
  "data": {
    "credential": {
      "issuer": "did:ethr:development:0x02a7b4984ecb19b7c5a298
         6735e5d51924a9767323f5d0e37516dd256e87420143",
      "type": [
      "VerifiableCredential",
      "UniversityStudentCredential"
      ],
      "credentialSubject": {
        "id": "did:ethr:development:0x025862baf2cdb6937f9fef45
           4da3da96fe925764b9b9362507f732d1bd09e9702f",
        "grado": "Universitario",
        "titulo": "Ingenieria Electronica",
        "dni": "20085452C"
      }
    },
    "options": {
      "proofType": "EcdsaSecp256k1RecoverySignature2020"
    }
  }
}]
```

Listado A.3: Fragmento de un mensaje "offer-credential"

```
{
  "id": "11f027ca-d0bc-4fb2-bb3a-6c5269cd6949",
  "type": "https://didcomm.org/issue-credential/2.0/request-
     credential",
  "raw": ... ,
  "data": {
    "@type": "https://didcomm.org/issue-credential/2.0/request
       -credential",
    "@id": "11f027ca-d0bc-4fb2-bb3a-6c5269cd6949",
    "formats": [
      {
        "attach_id": "31e6ee93-c3e1-4d73-b85f-a2c656a084c4",
        "format": "aries/ld-proof-vc-detail@v1.0"
      }
    ],
    "requests~attach": [
      {
        "@id": "31e6ee93-c3e1-4d73-b85f-a2c656a084c4",
        "mime-type": "application/json",
        "data": {
          "credential": {
            "type": ["VerifiableCredential", "
               UniversityDegreeCredential"],
            "issuer": "did:ethr:development:0x02a7b4984ecb19b7
               c5a2986735e5d51924a9767323f5d0e37516dd256e87420
               143",
            "credentialSubject": {
              "id": "did:ethr:development:0x025862baf2cdb6937f
                 9fef454da3da96fe925764b9b9362507f732d1bd09e97
                 02f"
            }
          },
          "options": {
            "proofType": "EcdsaSecp256k1RecoverySignature2020"
          }
        }
      }
    ]
  },
  "from": "did:ethr:development:0x025862baf2cdb6937f9fef454da3
     da96fe925764b9b9362507f732d1bd09e9702f",
  "to": "did:ethr:development:0x02a7b4984ecb19b7c5a2986735e5d5
     1924a9767323f5d0e37516dd256e87420143"
}
```

Listado A.4: Contenido de un mensaje "request-credential"

```
{
  "id": "5bbc054b-db3c-44fc-928a-bb8e86980b7d",
  "type": "https://didcomm.org/present-proof/2.0/propose-
     presentation",
  "raw": ... ,
  "data": {
    "@type": "https://didcomm.org/present-proof/2.0/propose-
       presentation",
    "@id": "5bbc054b-db3c-44fc-928a-bb8e86980b7d",
    "formats": [
      {
        "attach_id": "53cc80e6-a39f-4c6c-a307-e8c1e8e5ea63",
        "format": "dif/presentation-exchange/definitions@v1.0"
      }
    ],
    "proposals~attach": [
      {
        "@id": "53cc80e6-a39f-4c6c-a307-e8c1e8e5ea63",
        "mime-type": "application/json",
        "data": {
          "input_descriptors": {
            "name": "UniversityDegreeCredential",
            "schema": [{
                "uri": "hub://example.com/degree.json"
            }]
          }
        }
      }
    ]
  },
  "metaData": [
    {
      "type": "DIDCommMessaging",
      "value": "credential-flow-demo"
    },
    {
      "type": "didCommMetaData",
      "value": "{"packing":"jws"}"
    }
  ],
  "from": "did:ethr:development:0x025862baf2cdb6937f9fef454da3
     da96fe925764b9b9362507f732d1bd09e9702f",
  "to": "did:ethr:development:0x02a7b4984ecb19b7c5a2986735e5d5
     1924a9767323f5d0e37516dd256e87420143"
}
```

Listado A.5: Contenido de un mensaje "propose-presentation"

```
{
  "id": "4c50d784-c9d9-4151-99ef-a8cc4075d441",
  "type": "https://didcomm.org/present-proof/2.0/request-
     presentation",
  "raw": ... ,
  "data": {
    "@type": "https://didcomm.org/present-proof/2.0/request-
       presentation",
    "@id": "4c50d784-c9d9-4151-99ef-a8cc4075d441",
    "formats": [
      {
        "attach_id": "13e0835e-7ee3-4b27-88da-17817189b68e",
        "format": "dif/presentation-exchange/definitions@v1.0"
      }
    ],
    "request_presentations~attach": [
      {
        "@id": "13e0835e-7ee3-4b27-88da-17817189b68e",
        "mime-type": "application/json",
        "data": {
          "options": {
            "challenge": "test_challenge",
            "domain": "defaultDomain"
          },
          "presentation_definition": {
            "input_descriptors": {
              "name": "UniversityDegreeCredential",
              "schema": [{
                "uri": "hub://example.com/degree.json"
              }]
            },
            "format": {
              "ldp_vp": {
                "proof_type": ["Ed25519Signature2018"]
              }
            }
          }
        }
      }
    ]
  }
  "from": "did:ethr:development:0x02a7b4984ecb19b7c5a2986735e5
     d51924a9767323f5d0e37516dd256e87420143",
  "to": "did:ethr:development:0x025862baf2cdb6937f9fef454da3da
     96fe925764b9b9362507f732d1bd09e9702f"
}
```

Listado A.6: Contenido de un mensaje "request-presentation"

```
{
  "id": "4a98be03-f5eb-45c1-a60b-e5a05b0ac17a",
  "type": "https://didcomm.org/present-proof/2.0/presentation",
  "raw": ... ,
  "data": {
    "@type": "https://didcomm.org/present-proof/2.0/presentation",
    "@id": "4a98be03-f5eb-45c1-a60b-e5a05b0ac17a",
    "comment": "Here you have the presentation requested",
    "formats": [
      {
        "attach_id": "12be72e2-f412-4c1e-9dbb-91e41dff5829",
        "format": "dif/presentation-exchange/submission@v1.0"
      }
    ],
    "presentations~attach": [
      {
        "@id": "12be72e2-f412-4c1e-9dbb-91e41dff5829",
        "mime-type": "application/ld+json",
        "data": {
          "@context": ... ,
          "holder": "did:ethr:development:0x025862baf2cdb6937f9fef454da
             3da96fe925764b9b9362507f732d1bd09e9702f",
          "verifiableCredential": { verifiable credential },
          "verifier": [
            "did:ethr:development:0x02a7b4984ecb19b7c5a2986735e5d51924a
               9767323f5d0e37516dd256e87420143"
          ],
          "type": [
            "VerifiablePresentation"
          ],
          "proof": {
            "type": "EcdsaSecp256k1RecoverySignature2020",
            "created": "2023-07-04T06:58:18Z",
            "verificationMethod": "did:ethr:development:0x025862baf2cdb
               6937f9fef454da3da96fe925764b9b9362507f732d1bd09e9702f#
               controller",
            "proofPurpose": "authentication",
            "challenge": "test_challenge",
            "jws": "eyJhbGciOiJFUzI1NkstUiIsImI2NCI6ZmFsc2
               UsImNyaXQiOlsiYjY0Il19..Sjo23mp1o3CfpRUVev4xTnIGH_WOEk-
               B8eMvwcJvTgweMKlTFe54p7K_WapbIt3M2WgOhlCckOJ4hbmfOWcd0
               gA"
          }
        }
      }
    ]
  },
  "from": "did:ethr:development:0x025862baf2cdb6937f9fef454da3da96fe925
     764b9b9362507f732d1bd09e9702f",
  "to": "did:ethr:development:0x02a7b4984ecb19b7c5a2986735e5d51924a9767
     323f5d0e37516dd256e87420143",
}
```


Listado A.7: Contenido de un mensaje "presentation"

APÉNDICE B
Objetivos de desarrollo sostenible

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenible	**Alto**	**Medio**	**Bajo**	**No procede**
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**		X		
ODS 4. **Educación de calidad.**		X		
ODS 5. **Igualdad de género.**				X
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**		X		
ODS 9. **Industria, innovación e infraestructuras.**	X			
ODS 10. **Reducción de las desigualdades.**		X		
ODS 11. **Ciudades y comunidades sostenibles.**				X
ODS 12. **Producción y consumo responsables.**				X
ODS 13. **Acción por el clima.**				X
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**		X		
ODS 17. **Alianzas para lograr objetivos.**				X

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

La identidad autosoberana se refiere a la capacidad de una persona para tener control y potestad sobre su propia identidad y datos personales en el mundo digital. Esto implica el derecho de las personas a decidir cómo y cuándo comparten su información personal, quién puede acceder a ella y con qué propósito. En un contexto de creciente digitalización, donde los datos personales se utilizan para diversos fines, desde la prestación de servicios hasta la toma de decisiones gubernamentales, la identidad digital se convierte en un tema crítico para proteger la privacidad y la seguridad de las personas.

Uno de los Objetivos de Desarrollo Sostenible relacionado con la identidad autónoma es el ODS 16: Paz, Justicia e Instituciones Sólidas. Garantizar que las personas tengan control sobre sus datos personales y puedan proteger su privacidad y seguridad digital es fundamental para construir sociedades más justas y transparentes. La protección de la privacidad y la prevención del mal uso de datos personales contribuyen a la creación de mejores organizaciones, fortaleciendo la confianza de la ciudadanía en los sistemas gubernamentales y privados.

La identidad autosoberana también puede fomentar la tecnología responsable, alineada con el ODS 9: Industria, Innovación e Infraestructura. Al promover sistemas de manejo de datos centrados en el individuo, se incentiva la innovación tecnológica que respeta la privacidad y la seguridad, evitando prácticas invasivas y dañinas para la sociedad. También está vinculada con el ODS 10: Reducción de las Desigualdades. Permitir que personas vuelvan a tener control sobre sus identidades puede ser especialmente útil en situaciones de riesgo como los refugiados de guerra.

Además, la adopción de los sistemas basados en identidad autosoberana puede facilitar el acceso a servicios digitales, lo que tiene relación con varios ODS. Por ejemplo, en el ODS 3: Salud y Bienestar, la identidad autónoma permite a las personas acceder a sus registros médicos de manera segura y contribuye a una protección sobre datos médicos privados. En el ODS 4: Educación de Calidad, puede facilitar el acceso a los sistemas educativos de otros países. En el ODS 8: Trabajo Decente y Crecimiento Económico, la identidad autónoma facilita la inclusión financiera, permitiendo que las personas sin cuentas bancarias tradicionales accedan a servicios financieros digitales.

Como se puede ver la identidad autosoberana y los ODS están relacionados en varios aspectos. Garantizar la protección de la privacidad y la seguridad digital, empoderar a diversos grupos y fomentar la tecnología responsable son fundamentales para lograr un desarrollo sostenible en la era digital.