TRABAJO FIN DE GRADO

INGENIERÍA INFORMÁTICA

Implementación en Android de un agente de Identidad auto-Soberana (SSI) utilizando Hyperledger Aries

Autor
Iván Cortón da Silva

Director
Rafael Alejandro Rodríguez Gómez

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, septiembre de 2023

Implementación en Android de un agente de Identidad auto-Soberana (SSI) utilizando Hyperledger Aries

Autor
Iván Cortón da Silva

Director
Rafael Alejandro Rodríguez Gómez

Implementación en Android de un agente de Identidad auto-Soberana (SSI) utilizando Hyperledger Aries

Iván Cortón da Silva

Palabras clave: Hyperledger Aries, identidad auto-soberana (SSI), autenticación online, identidad digital, eIDAS2

Resumen

En los últimos años, el proceso de identificación en línea se ha convertido en una práctica común en nuestra vida diaria. Sin embargo, el uso frecuente de servicios de terceros, como Google, para autenticarnos, plantea preocupaciones fundamentales. Estos servicios centralizados no solo almacenan nuestros datos de identidad, sino que también ejercen un control significativo sobre ellos, creando una asimetría en la que la entidad que gestiona nuestra identidad tiene un poder desproporcionado sobre nosotros.

La Unión Europea ha reconocido estos desafíos y ha propuesto un Reglamento para abordarlos, reflejando una creciente conciencia sobre la necesidad de devolver el control de la identidad digital al usuario.

En este contexto, este Trabajo de Fin de Grado se centra en el estudio de la identidad auto-soberana propuesta por la *Linux Foundation*. El objetivo principal es comprender en profundidad su funcionamiento y, a partir de este conocimiento, desarrollar una aplicación Android. Esta aplicación permitirá a los usuarios gestionar sus propias conexiones y la emisión y recepción de credenciales, eliminando la dependencia de entidades externas en la autenticación en línea.

Además, se realiza una investigación exhaustiva sobre las soluciones existentes, tanto comerciales como de investigación, que abordan la problemática de la identificación en línea. Se lleva a cabo una comparativa detallada para destacar las diferencias y ventajas de la solución propuesta en este trabajo.

Para validar la solución desarrollada, se realizan pruebas exhaustivas que demuestran su eficacia y seguridad en la gestión de la identidad digital. Estas pruebas incluyen una serie de escenarios de uso real.

En última instancia, este trabajo contribuye al avance de la *SSI* como una solución prometedora para la gestión de la identidad digital, ofreciendo recomendaciones concretas para su implementación efectiva y segura en futuros sistemas de identificación digital. Su objetivo final es mejorar tanto

la privacidad como la seguridad de los usuarios, garantizando que sean los verdaderos poseedores y controladores de su propia identidad en línea.

La aplicación Android desarrollada durante este trabajo se encuentra disponible en el siguiente repositorio de Github: `https://github.com/IvanCorton/UGR-Credentify`.

Android Implementation of a Self-Sovereign Identity (SSI) Agent Using Hyperledger Aries

Iván Cortón da Silva

Keywords: Hyperledger Aries, self-sovereign identity (SSI), online authentication, digital identity, eIDAS2

Abstract

In recent years, the online identification process has become a common practice in our daily lives. However, the frequent use of third-party services, such as Google, to authenticate us raises fundamental concerns. These centralized services not only store our identity data, but also exercise significant control over it, creating an asymmetry in which the entity managing our identity has disproportionate power over us.

The European Union has recognized these challenges and has proposed a Regulation to address them, reflecting a growing awareness of the need to return control of digital identity to the user.

In this context, this Final Degree Project focuses on the study of the self-sovereign identity proposed by the *Linux Foundation*. The main objective is to understand in depth how it works and, from this knowledge, to develop an Android application. This application will allow users to manage their own connections and the issuance and receipt of credentials, eliminating the dependency on external entities in online authentication.

In addition, a thorough investigation of existing solutions, both commercial and research, that address the problem of online identification is carried out. A detailed comparison is carried out to highlight the differences and advantages of the solution proposed in this paper.

To validate the developed solution, extensive tests are conducted to demonstrate its effectiveness and security in digital identity management. These tests include a series of real use scenarios.

Ultimately, this work contributes to the advancement of *SSI* as a promising solution for digital identity management, offering concrete recommendations for its effective and secure implementation in future digital identification systems. Its ultimate goal is to enhance both the privacy and security of users, ensuring that they are the true possessors and controllers of their own online identity.

The Android application developed during this work is available in the following Github repository: `https://github.com/IvanCorton/UGR-Credentify`.

Yo, **Iván Cortón da Silva**, alumno de la titulación INGENIERÍA INFORMÁTICA de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 15426623V, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Iván Cortón da Silva

Granada a 1 de septiembre de 2023.

D. **Rafael Alejandro Rodríguez Gómez**, Profesor del Área de Telemática del Departamento Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada.

Informa:

Que el presente trabajo, titulado ***Implementación en Android de un agente de Identidad auto-Soberana (SSI) utilizando Hyperledger Aries***, ha sido realizado bajo su supervisión por **Iván Cortón da Silva**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 1 de septiembre de 2023.

El director:

Rafael Alejandro Rodríguez Gómez

Agradecimientos

En primer lugar agradecer enormemente el trabajo, esfuerzo, tiempo y dedicación que ha tomado mi tutor, Rafael, por haberme dado la oportunidad de realizar un trabajo de investigación de esta envergadura.

Me gustaría destacar especialmente la actitud positiva y todo lo que ha hecho por mi durante el desarrollo de este trabajo, sin él, este trabajo no habría sido posible.

Agradecer a mi familia; mi madre, mi abuela, mi hermana y mi tía, porque sé que no ha sido fácil para ellas acompañarme en este camino lleno de adversidades donde nunca me han fallado. Estas me han forjado como persona, y es por eso que todo lo que soy se lo debo a ellas, sin vosotras esto no hubiera tenido sentido, gracias de corazón.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación y contexto del proyecto

El informe anual *Global Networking Trends* de Cisco en 2023 [1] destaca la creciente importancia de la seguridad en la gestión de redes.

Según este informe, el 56 % de los profesionales de TI encuestados identifican la seguridad como el principal desafío en este ámbito. Además, se ha producido un cambio significativo en las prioridades de TI, donde la agilidad y el rendimiento empresarial han superado a las preocupaciones tradicionales sobre costos y gestión de redes. Este cambio refleja la creciente demanda de soluciones que proporcionen visibilidad completa, automatización y acceso a modelos operativos basados en la nube.

Este contexto de transformación tecnológica y digitalización subraya la necesidad de abordar los desafíos relacionados con la identificación en línea. En respuesta a esta necesidad, la *Linux Foundation* [2] ha propuesto un marco regulativo para la creación de una Identidad Digital Europea autosoberana (SSI) que sea segura y compatible en toda la Unión Europea.

En las últimas décadas, hemos sido testigos de un crecimiento exponencial de Internet impulsado por avances tecnológicos. Este crecimiento se refleja en el aumento constante de dispositivos conectados, servicios disponibles y hogares con acceso a Internet.

Sin embargo, este crecimiento vertiginoso ha llevado a una avalancha de comunicaciones en línea, que incluyen interacciones entre usuarios y comunicaciones máquina a máquina (M2M). A pesar de esta conectividad, la identificación en línea se enfrenta a desafíos significativos. Las regulaciones actuales, como el Reglamento (UE) nº 910/2014 [3], junto con los métodos de autenticación tradicionales, no proporcionan el nivel deseado de confianza, privacidad ni usabilidad.

Para abordar estos desafíos, surgió la Propuesta de Reglamento eIDAS 2 [4], presentada por el Parlamento Europeo y el Consejo en junio de 2021. Este proyecto tiene como objetivo establecer un Marco para una Identidad Digital Europea (SSI) que se adapte a la era digital actual. El propósito es crear un sistema de identidad auto-soberana (SSI) altamente seguro y confiable que sea compatible en toda la Unión Europea.

La *Linux Foundation* ha desarrollado tres proyectos clave, Indy, Ursa y Aries, que se basan en tecnología Blockchain. Estos proyectos, en conjunto, crean un sistema SSI distribuido, descentralizado e inmutable que ofrece autenticación e integridad en los registros. Además, la introducción de una cartera de identidad digital otorga a los usuarios un control absoluto sobre sus datos.

Se espera que esta iniciativa tenga un amplio alcance y penetración en los próximos años, transformando la forma en que gestionamos la identidad en línea. La combinación de un entorno altamente conectado y regulaciones actualizadas promete un futuro emocionante para la identidad digital en Europa.

La aplicación Android desarrollada durante este trabajo se encuentra disponible en el siguiente repositorio de Github: `https://github.com/IvanCorton/UGR-Credentify`.

1.2. Objetivos del proyecto y solución propuesta

El proyecto se enfoca en el estudio detallado del ecosistema ARIES, con el propósito central de diseñar e implementar una aplicación Android que funcione como cartera o *wallet* de credenciales verificables. Esta aplicación ofrecerá a los usuarios la capacidad de generar y almacenar credenciales válidas, brindando flexibilidad en su operación.

Para la consecución de este objetivo general es necesario cumplir con las siguientes tareas:

1. **Revisión de Literatura y Documentación**: Realizar una revisión exhaustiva de la literatura y documentación existente relacionada con el ecosistema ARIES y el proyecto Hyperledger de la *Linux Foundation*. Esta revisión proporcionará una base sólida para el desarrollo posterior.

2. **Diseño y Arquitectura de la Aplicación**: Definir la arquitectura de la aplicación Android, detallando las funcionalidades que se incorporarán. Este paso es fundamental para la planificación efectiva de la implementación.

3. **Implementación Tecnológica**: Llevar a cabo la implementación de la aplicación, integrando de manera efectiva la tecnología ARIES. Esto implica traducir el diseño conceptual en una aplicación funcional.

1.3. Estructura de la memoria

En el Capítulo 2 (Contexto y estado del Arte) se explican los contenidos teóricos que sirven como base para comprender el funcionamiento del proyecto Hyperledger y su relación con la identidad auto-soberana. Además se realiza una comparativa entre diferentes aplicaciones estudiadas para aportar una motivación real del objetivo de la implementación de la aplicación que se trata en este proyecto.

A continuación, en el Capítulo 3 (Objetivos y planificación) se definen las fases en las que se desarrolla el proyecto, así como el tiempo empleado en la realización junto a un cálculo de referencia sobre el presupuesto del trabajo realizado.

En el Capítulo 4 (Diseño) se exploran las decisiones de diseño cruciales, como la elección de una metodología de desarrollo y la arquitectura de los agentes, y se presentan los requisitos funcionales y no funcionales junto con casos de uso. Seguidamente se encuentra el Capítulo 5 (Implementación), donde se especifican los elementos usados para realizar pruebas sobre diferentes escenarios de la aplicación.

Se detalla en el Capítulo 6 (Validación y pruebas) todo el procedimiento seguido para la ejecución de pruebas sobre la aplicación y las diferentes interfaces de usuario desarrolladas permitiendo la interacción.

Por último, el Capítulo 7 (Conclusiones y Líneas de Trabajo Futuro) incluye unas breves conclusiones sobre el proyecto realizado y unas posibles líneas de trabajo futuro.

Capítulo 2

Contexto y estado del Arte

Este capítulo se enfoca en explorar el marco teórico de las cartera de identidad digital y su relevancia en la era actual. Se explorará el estado del arte de esta tecnología emergente y cómo se compara con las alternativas tradicionales.

2.1. Entendiendo la Identidad Auto-Soberana (SSI)

La identidad se entiende como un conjunto de características únicas que distinguen a una persona o grupo de los demás [5]. La descentralización, por otro lado, se refiere a la dispersión de funciones, poderes, personas o cosas lejos de una ubicación o autoridad central [6]. Ambos conceptos se unen en la identidad soberana descentralizada (*Self-Sovereign Identity*) , un enfoque emergente para la gestión de la identidad digital que se fundamenta en ciertos principios esenciales [7].

2.1.1. Sistemas Centralizados versus Descentralizados: una comparativa

Los sistemas centralizados y descentralizados difieren principalmente en la estructura y gestión de la autoridad. En un sistema centralizado, una única entidad tiene el control total sobre la gestión de la identidad. En contraste, en un sistema descentralizado, la autoridad y el control están distribuidos entre múltiples nodos o entidades. En la Figura 2.1 se muestra un esquema básico sobre la diferenciación entre los modelos centralizados y descentralizados.

Dentro del ámbito de los sistemas descentralizados, las aplicaciones móvi-

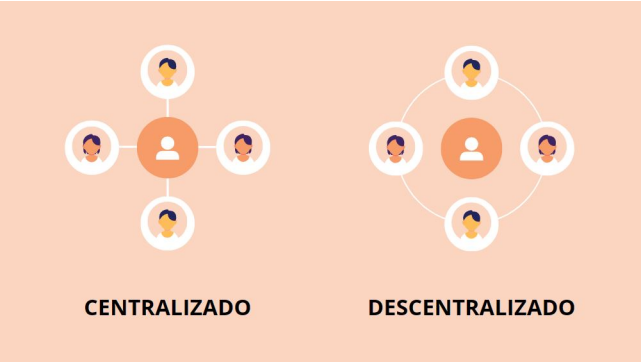

Figura 2.1: Modelo centralizado frente a descentralizado

les de cartera de identidad digital han surgido como una evolución en el paradigma de la identificación. Estas aplicaciones utilizan la tecnología blockchain para almacenar y gestionar identidades digitales de forma segura y descentralizada.

2.1.2. Reinventando la Autenticación

En la actualidad, los métodos de autenticación tradicionales como la autenticación basada en contraseñas, la autenticación multifactor y la biometría, presentan limitaciones en términos de seguridad, confianza y privacidad. Estas limitaciones se hacen evidentes al contrastar estos sistemas con las nuevas iniciativas como Hyperledger que se fundamentan en tecnologías emergentes, tales como la tecnología blockchain y las Credenciales Verificables (*VC*) las cuales definiremos más adelante.

Al investigar los métodos de autenticación actuales más comúnes, se presentan varios problemas:

El modelo de usuario y contraseña, puede ser inseguro en caso de usar contraseñas cortas o por el contrario, de excesiva dificultad de acuerdo a la complejidad de la misma, aunque la realidad es que la mayoría de usuarios utilizan la misma para múltiples cuentas [8, 9], lo cual expone la información en la mayoría de servicios utilizados por el usuario si se consigue averiguar la contraseña en un único sitio.

La autenticación multifactor (de siglas *MFA*) es una tecnología de seguridad que combina dos o más credenciales independientes: por ejemplo, una contraseña y el uso de métodos de verificación biométrica. Aunque este método parezca más robusto, tampoco es infalible [10], por lo que análogamente al modelo de usuario y contraseña presenta un riesgo que trae consigo la biometría: las características físicas son inmutables, por lo que si se ven

comprometidas no pueden ser modificadas.

Por último, el uso de proveedores de identidad como Google o Facebook, para realizar la autenticación de doble factor (de siglas *2FA*). Este modelo presenta un punto de fallo único, ya que, si se compromete nuestra cuenta por parte del proveedor de identidad, se dispone del acceso a todos los datos de los servicios en los que el usuario se haya registrado [11].

Los métodos que se mencionan anteriormente comparten un problema común, compuesto por la correlación de los datos a través del uso de identificadores en diferentes plataformas, que puede provocar la creación de perfiles o recopilación de información sobre los lugares en que el usuario se identifica, sin el consentimiento o el consentimiento implícito del mismo.

2.2. Definiciones Clave

Antes de adentrarnos en el análisis de las innovadoras propuestas de Hyperledger y en cómo la normativa de Hyperledger busca reforzar los principios de la identidad auto-soberana, es crucial entender que este proyecto permite a las entidades emitir, almacenar y verificar de manera segura y privada las credenciales digitales. Para tal fin, se estructura alrededor de los siguientes principios esenciales [12]:

- **Interoperabilidad**: La capacidad de los sistemas de identidad de interactuar entre sí y con otros sistemas, independientemente de las tecnologías o estándares subyacentes que empleen.
- **Portabilidad**: Las identidades deben ser transportables, permitiendo a los usuarios tener el control y la posibilidad de mover sus identidades entre diferentes plataformas y servicios.
- **Escalabilidad**: Las soluciones de identidad deben tener la capacidad de abarcar una amplia gama de aplicaciones, desde transacciones personales de pequeña escala hasta sistemas empresariales de gran envergadura.
- **Seguridad y privacidad**: Las identidades deben ser seguras y garantizar la privacidad del usuario, incluyendo la posibilidad de compartir únicamente la información necesaria en cada transacción.
- **Usabilidad**: Las soluciones de identidad deben ser de fácil uso e integración en los flujos de trabajo existentes, tanto para los usuarios finales como para los desarrolladores y proveedores de servicios.

En este contexto, es importante entender los siguientes conceptos:

2.2.1. Credencial

Una credencial es un documento que acredita a una persona para desempeñar una determinada función [13], una prueba de una afirmación cualificada sobre una entidad, validando características como identidad, edad, nacionalidad o calificaciones. En el ecosistema de la identidad digital, estas credenciales se convierten en elementos fundamentales para confirmar y verificar las afirmaciones sobre un individuo o una entidad.

2.2.2. Blockchain

Una blockchain, o cadena de bloques, es una tecnología de almacenamiento de datos que se basa en una lista creciente de registros, llamados bloques, que están vinculados y protegidos mediante técnicas criptográficas. Cada bloque contiene un enlace a un bloque anterior, una marca de tiempo y datos de transacción. Este diseño hace que una blockchain sea resistente a la modificación de los datos: una vez que los datos se han registrado en un bloque, no pueden ser alterados sin la alteración de todos los bloques siguientes, lo cual requiere la colaboración de la mayoría de la red [14] [15].

El proyecto Hyperledger presenta una blockchain única, que aunque comparte características como la inmutabilidad del registro con otras, se enfoca principalmente en la identidad, impidiendo el intercambio de activos o la ejecución de smart contracts.

Las blockchains se clasifican en públicas o privadas según el tipo de acceso, y con permisos o sin permisos según la validación. Ejemplos de blockchains públicas y sin permisos son las usadas por Bitcoin o Ethereum, accesibles por cualquier persona que pueda participar en el proceso de minería.

Hyperledger propone una blockchain accesible a todos pero solo modificable por participantes autorizados. Esto resulta en un sistema colaborativo y amigable comparado con el modelo de las criptomonedas.

Dada la naturaleza pública de la blockchain, solo se deben registrar datos públicos, ya que incluso los datos encriptados podrían ser vulnerables con el avance tecnológico. Los datos privados, como las credenciales, deben estar bajo control del propietario, resguardados en su monedero digital.

En la blockchain de Hyperledger se registran cuatro elementos: DIDs públicos, esquemas, definiciones de credenciales y registros de revocación como se muestra en la Figura 2.2.

Los expedidores de credenciales o *Issuer* deben registrar sus DIDs públicos (se profundiza en este concepto en la Subsección 2.2.7) en la blockchain para que los verificadores o *Verifiers* puedan identificar al *Issuer* que ha emitido una credencial específica presentada por un propietario de una cartera

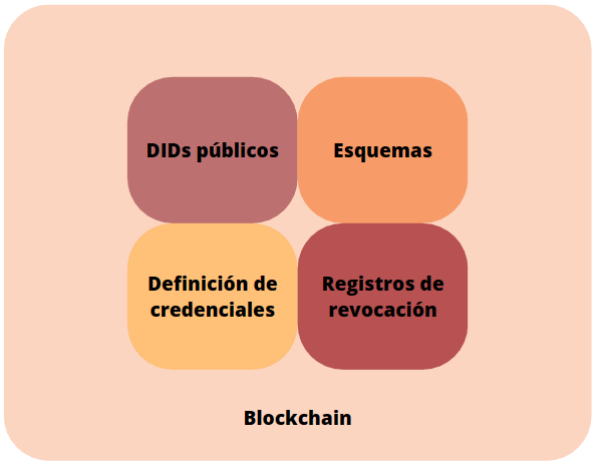

Figura 2.2: Elementos que se registran en Hyperledger

digital o *Holder*. Esto permite a los *Verifiers* determinar si pueden confiar en la información proporcionada por el *Holder*.

Cuando un *Issuer* quiere emitir una credencial, necesita saber qué campos debe completar en esa credencial. Un esquema es una plantilla que el *Issuer* utiliza para determinar qué campos debe incluir en una credencial verificable o *VC* específica (se profundiza en este concepto en la Subsección 2.2.6). Los *Issuers* pueden utilizar esquemas creados por otros *Issuers*, lo que ayuda a reducir la cantidad de esquemas registrados en la blockchain y permite la estandarización de ciertas credenciales. Por ejemplo, todas las compañías de seguros podrían emitir *VCs* de pólizas de seguro a sus clientes. Dado que se espera que los atributos de dicha credencial sean los mismos para todas las compañías de seguros (nombre del titular, tipo de seguro, cobertura, entre otros), todas podrían utilizar el mismo esquema, que ha sido registrado una sola vez en la blockchain.

2.2.3. Identidad Auto-Soberana (SSI)

La identidad auto-soberana (*SSI o Self-Sovereign Identity*) emerge de la aplicación de la blockchain a la gestión de identidades. La SSI es un enfoque que da el control de la identidad digital a los propios individuos. En lugar de depender de terceros para validar y almacenar identidades, el SSI permite a los individuos generar y controlar sus propias identidades. [16]

2.2.4. eIDAS2

El concepto de SSI se alinea con el reglamento eIDAS (en Inglés *Electronic Identification, Authentication and Trust Services*), una normativa de la Unión Europea que establece las reglas para la identificación electrónica y los servicios de confianza para transacciones electrónicas. eIDAS2, una evolución de esta regulación publicada el 3 de Junio de 2021, obliga tanto a la administración pública como a muchos sectores privados a adaptarse a los sistemas de SSI, estableciendo así un nuevo paradigma en el trato de la identidad digital y la confianza en Internet. [4]

2.2.5. Ledger y Wallet Fría

Los ledgers son bases de datos distribuidas usadas en la tecnología blockchain para registrar transacciones. En la gestión de la identidad digital, un ledger puede utilizarse para almacenar y verificar las credenciales. Estas credenciales pueden ser almacenadas de forma segura en lo que se conoce como una "wallet fría", un tipo de almacenamiento de criptomonedas que permanece desconectado de internet, proporcionando así un nivel adicional de seguridad contra ataques cibernéticos.

2.2.6. Credenciales Verificables

Las credenciales verificables (*VCs*) son una innovación esencial en la gestión de la identidad digital, y uno de los pilares fundamentales de Hyperledger [17]. Estas credenciales se componen de varios atributos que pueden ser verificados de manera independiente, permitiendo un intercambio de información con un alto grado de privacidad.

Una credencial verificable puede considerarse como una tarjeta de identificación digital con atributos tales como fecha de nacimiento, nombre completo y dirección. En lugar de compartir toda la tarjeta de identificación, es posible optar por compartir únicamente el atributo relevante, como la fecha de nacimiento, para demostrar la mayoría de edad.

Hyperledger respalda el uso de las *VCs* en su marco de trabajo implementando la norma propuesta por el Consorcio World Wide Web (W3C) [18]. Las *VCs* en Hyperledger se dividen en tres componentes esenciales:

1. **Emisor (*Issuer*)**: La entidad que crea y otorga la credencial al titular.

2. **Titular (*Holder*)**: La entidad que recibe la credencial del emisor y la presenta cuando se le solicita.

3. **Verificador (*Verifier*)**: La entidad que solicita y verifica la credencial presentada por el titular.

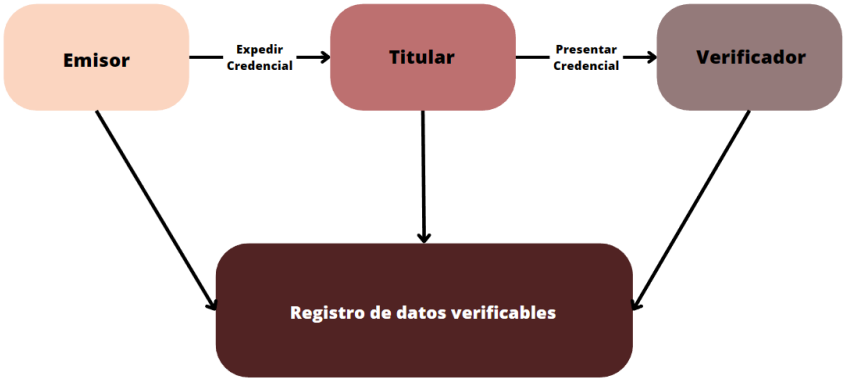

Figura 2.3: Flujo de uso de Credenciales Verificables en Hyperledger

Las *VCs* también introducen el concepto de cartera digital, un lugar seguro para almacenar las credenciales de un usuario. Esta cartera digital, similar a una cartera física, permite a los usuarios decidir cuándo y dónde usar sus credenciales, aumentando así la privacidad, ya que el usuario tiene el control total sobre cuándo y con quién compartir sus datos.

Al utilizar las credenciales verificables en Hyperledger, el verificador no necesita ponerse en contacto con el emisor en ningún momento. El titular de la credencial proporciona toda la información necesaria al verificador, que luego puede verificar la autenticidad de los datos directamente desde el registro de datos verificable en la cadena de bloques.

Todo el procedimiento que se ha mencionado anteriormente se puede identificar facilmente a través de la Figura 2.3

Este enfoque se alinea perfectamente con el concepto de Identidad Auto-Soberana (*SSI*), donde los usuarios tienen el control total de sus propios datos. En Hyperledger, no hay una autoridad central que almacene los datos del usuario; en su lugar, el usuario posee y controla sus datos. Este modelo descentralizado, en combinación con el uso de criptografía, permite la presentación y verificación de pruebas de credenciales de manera segura y confiable.

En la Figura 2.4 se aclara cualquier ambigüedad que pueda surgir entre los conceptos credencial y atributo a lo largo del presente trabajo, ya que son conceptos que se tratan con frecuencia a lo largo del mismo.

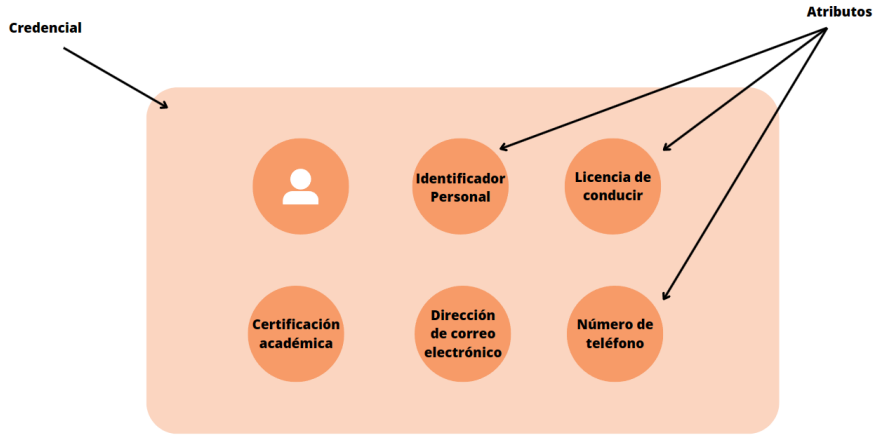

Figura 2.4: Credencial y atributos

2.2.7. Identificadores Descentralizados (DIDs)

¿Qué es un Identificador Descentralizado (DID)?

Un Identificador Descentralizado (*Decentralized IDentifier*) es una nueva forma de identificación en el mundo digital. En términos sencillos, un *DID* es como un número de teléfono o una dirección de correo electrónico, pero con características mucho más avanzadas.

A diferencia de los identificadores tradicionales que usamos hoy en día, como los nombres de usuario, los *DIDs* son "autosoportados", lo que significa que no dependen de ninguna autoridad centralizada, como una empresa de tecnología o un gobierno, para funcionar. Los *DIDs* son creados, controlados y mantenidos por la persona o entidad que los usa, y pueden ser verificados de forma independiente por cualquier persona o entidad en Internet. Estos juegan un papel crucial en otorgar a los usuarios el control total sobre sus identidades digitales, en línea con los principios de privacidad que se discutieron anteriormente [19].

¿En qué consiste un DID?

Un *DID* consta de tres partes principales:

1. **Esquema**: La primera parte de un *DID* es siempre "did", que simplemente significa "identificador descentralizado". Esta parte del *DID* indica que se está utilizando un identificador descentralizado, en lugar de un identificador tradicional.

2. **Método**: La segunda parte de un *DID* es el "método", que indica qué reglas y protocolos se utilizarán para crear, leer, actualizar y eliminar el *DID*. Algunos ejemplos de métodos son "sov" para la red de Sovereign, y "ethr" para la red de Ethereum.

3. **Identificador específico**: La tercera parte de un *DID* es el identificador específico, que es único para cada *DID*. Este es el elemento que hace que cada *DID* sea diferente de todos los demás DIDs.

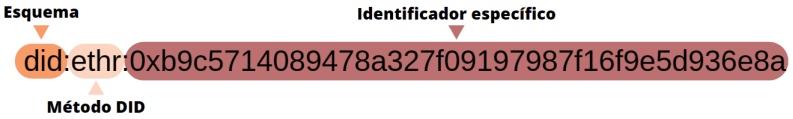

Figura 2.5: Ejemplo de un identificador descentralizado (*DID*)

En la Figura 2.5 se representa un ejemplo de identificador descentralizado en el que se diferencian las tres partes principales: esquema, método e identificador específico.

¿Cómo funcionan los DIDs?

El funcionamiento de los *DIDs* se basa en la tecnología de cadena de bloques (blockchain). Cuando se crea un nuevo *DID*, la información se almacena en una cadena de bloques, donde se puede verificar de forma independiente. Este registro en la cadena de bloques también se utiliza para gestionar los *DIDs*.

Los *DIDs* son una parte fundamental de los sistemas *SSI*.

Para entender mejor cómo funcionan los *DIDs*, considera el siguiente ejemplo esquematizado en la Figura 2.6: Imagine que desea comprar un libro en línea, pero no quiere darle a la tienda en línea su dirección de correo electrónico, por temor a que la tienda venda su información a los publicistas. En lugar de darle a la tienda su dirección de correo electrónico, le da un *DID*. La tienda puede entonces utilizar este *DID* para verificar que eres tú, sin necesidad de acceder a ninguna de tu información personal.

En resumen, los DIDs son una forma innovadora de gestionar la identidad digital que pone el control y la privacidad en manos de los usuarios. A través de la tecnología blockchain, los DIDs permiten la creación de identidades digitales autosoportadas, lo que puede abrir nuevas formas de interacción en línea que son más seguras.

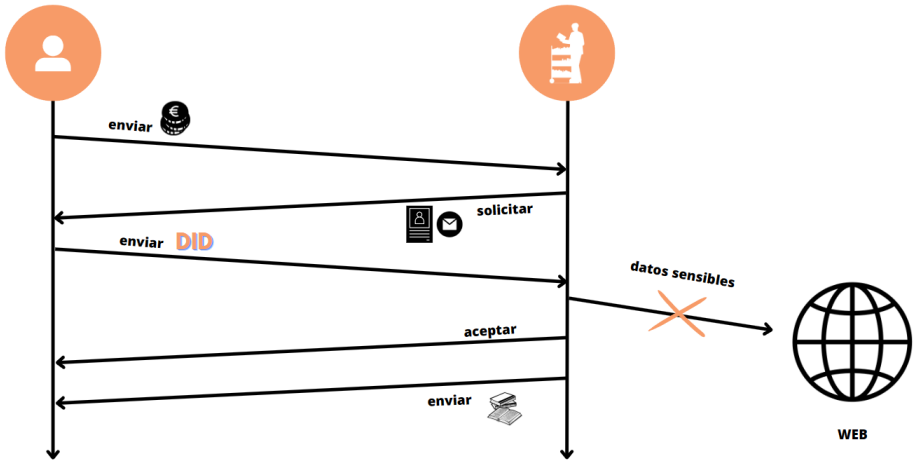

Figura 2.6: Ejemplo de aplicación de DID

2.2.8. Zero Knowledge Proof (ZKP) y Divulgación Selectiva

El esquema de credenciales verificables del proyecto Hyperledger, desarrollado por la Linux Foundation, potencia la privacidad de las entidades cuando intercambian información gracias a las pruebas de conocimiento cero (*ZKP*) y la divulgación selectiva de información. Este marco utiliza una variedad de tecnologías y servicios como el *Key-management Service* (*KMS*) y ARIES.

El *Key-management Service KMS* es un componente crítico en la infraestructura de seguridad que se utiliza para administrar las claves criptográficas en un sistema de criptografía. Esto puede abarcar la generación, el intercambio, el almacenamiento, el uso, la destrucción y la renovación de las claves. Su función es fundamental para preservar la seguridad de la información durante las transacciones que utilizan credenciales verificables.

Una prueba de conocimiento nulo tiene como objetivo confirmar una información determinada sin revelar ningún identificador que pueda correlacionarse con esa información. Las pruebas de conocimiento cero *ZKP* son métodos criptográficos que permiten a una entidad (verificador) validar una afirmación realizada por otra entidad sin necesidad de revelar los datos subyacentes de esa información. [20]

En la Figura 2.7 se representa un usuario con varios tipos de atributo sensibles encriptados donde se requiere utilizar el menor número de ellos posibles para identificarse frente a una entidad y comprometer su privacidad y seguridad lo menor posible.

Figura 2.7: Ejemplo de funcionamiento pruebas de conocimiento cero

Además, las pruebas de conocimiento cero deben cumplir tres propiedades:

- **Completitud**: Si la afirmación es verdadera, el verificador estará convencido de la veracidad de la prueba presentada por la otra entidad.
- **Solidez**: La probabilidad de que el verificador sea engañado por una declaración falsa es casi nula.
- **Conocimiento cero**: El verificador solo puede determinar si la afirmación realizada por la otra entidad es verdadera o falsa, pero no podrá conocer ningún detalle sobre la información.

Para ilustrar el concepto de pruebas de conocimiento cero *ZKP*, se considera uno de los ejemplos más conocidos: la cueva de Ali Baba. En este escenario, una persona, Alice, quiere demostrar a otra, Bob, que tiene la llave de una puerta en una cueva en forma de anillo, pero Alice no quiere mostrarle la llave a Bob. A través de una serie de interacciones, Alice puede demostrar a Bob que tiene la llave sin necesidad de mostrársela.

Además de las pruebas de conocimiento cero *ZKP*, el modelo de credenciales verificables también permite la divulgación selectiva de información. Las credenciales constan de atributos que pueden ser verificados individualmente. Esto permite seleccionar qué datos específicos de una credencial se desean compartir, y el Verificador puede verificarlos sin problemas.

Un ejemplo práctico de divulgación selectiva podría ser la verificación de que una persona es mayor de edad. En lugar de requerir documentos de identidad completos que contienen más información de la necesaria, a través del modelo de credenciales verificables, utilizando la divulgación selectiva de datos y las pruebas de conocimiento cero *ZKP*, se puede verificar que una persona es mayor de edad de una manera segura, confiable y respetuosa con la privacidad.

El marco ARIES, que se utiliza en este proyecto, proporciona herramientas para la creación, emisión, verificación y revocación de credenciales verificables. Incorpora una variedad de tecnologías, incluyendo INDY, una

biblioteca de blockchain distribuida, y URSA, una biblioteca de criptografía compartida que ofrece implementaciones de algoritmos criptográficos como firmas digitales y pruebas de conocimiento cero.

La combinación de las pruebas de conocimiento cero *ZKP*, la divulgación selectiva y el uso de tecnologías como *Key-management Service KMS* y ARIES en el marco de Hyperledger permite una reducción significativa de los datos expuestos durante el proceso de identificación. Al no revelar un identificador que pueda correlacionar datos y utilizar la divulgación selectiva para exponer solo los datos necesarios para una transacción, se da un paso importante para preservar la privacidad.

2.3. Descripción y relación de INDY, ARIES y URSA

Para entender completamente el marco de trabajo ARIES, es esencial discutir los componentes individuales, INDY, ARIES y URSA, y cómo interactúan entre sí para proporcionar una solución de identidad digital integral dentro del Ecosistema Hyperledger como se muestra en la Figura 2.8.

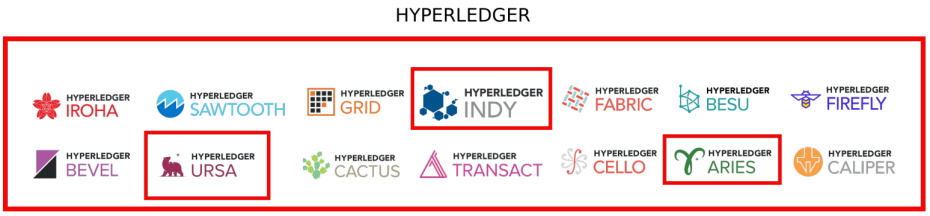

Figura 2.8: Ecosistema del Proyecto Hyperledger

2.3.1. ARIES

Hyperledger Aries es una infraestructura de software de código abierto para la gestión de la interacción entre pares (peer-to-peer), la emisión y verificación de credenciales digitales, y la interoperabilidad entre diferentes sistemas de identidad descentralizados. Aries está diseñado para facilitar la implementación de aplicaciones, productos y servicios de identidad digital en varios sistemas y redes. [21]

2.3.2. INDY

Hyperledger Indy es una biblioteca de software distribuida, basada en la tecnología blockchain, diseñada para la creación de identidades digitales auto-soberanas. Indy es único entre las plataformas blockchain debido a su énfasis en la interoperabilidad, facilitando a los usuarios el control sobre sus interacciones en línea y la capacidad de elegir con quién compartir información. Indy es la base sobre la cual se basan otras herramientas de Hyperledger, como ARIES y URSA. [22]

2.3.3. URSA

Hyperledger Ursa es una biblioteca de criptografía compartida que permite a las personas (y proyectos) evitar duplicar otro trabajo criptográfico y aumentar la seguridad en el proceso. Ursa hace que sea mucho más seguro y fácil para los proyectos de Hyperledger manejar la criptografía, al proporcionar implementaciones de algoritmos criptográficos de alta calidad y fácil de usar. [23]

2.3.4. Relación entre INDY, ARIES y URSA

Indy, Aries y Ursa son todas partes de la "pila de identidad"de Hyperledger. Indy proporciona la capa de almacenamiento de datos y el modelo de identidad, Ursa proporciona la criptografía necesaria para mantener la seguridad y la privacidad de los datos, y Aries proporciona los protocolos y las herramientas para permitir la interacción y el intercambio de datos entre pares.

En conjunto, estas tres herramientas trabajan juntas para proporcionar una solución completa para la creación, gestión y uso de identidades digitales auto-soberanas. Cada herramienta se enfoca en una parte diferente del problema de la identidad digital, y juntas proporcionan una solución que es más grande que la suma de sus partes.

2.4. Uso actual de SSI

La identidad auto-soberana (SSI) ha encontrado aplicaciones en diversas áreas, incluyendo la salud y la optimización del rendimiento de las aplicaciones basadas en blockchain.

Un estudio reciente de Merlin George y A. Chacko titulado "Health Passport: A blockchain-based PHR-integrated self-sovereign identity system" explora los requisitos para un sistema generalizado de Pasaporte de Salud. Los

autores utilizan la modelización orientada a agentes (AOM) para diseñar un sistema de SSI basado en blockchain integrado con el Registro de Salud Personal (PHR). Implementaron un concepto de prueba en Hyperledger Indy y Aries, integrado con el PHR - MediTrans, demostrando la viabilidad de la solución [24].

Además de las aplicaciones de SSI en el campo de la salud, también se están realizando avances significativos en la optimización del rendimiento de las aplicaciones de SSI basadas en blockchain. Un estudio de T. Pflanzner, Hamza Baniata, y A. Kertész titulado "Latency Analysis of Blockchain-Based SSI Applications" presenta un análisis detallado de la latencia de un sistema de blockchain privado y con permisos construido con Indy y Aries para aplicaciones de SSI. Los autores desarrollaron una aplicación en Python utilizando componentes de Indy y Aries contenerizados de los repositorios oficiales de Hyperledger. Implementaron su aplicación experimental en varias máquinas virtuales en la plataforma pública de Google Cloud y en su nube privada local utilizando una plataforma Docker con Kubernetes. Evaluaron y compararon su rendimiento con las métricas de latencia de lectura y escritura [25].

Hasta ahora, hemos discutido la teoría y los estudios en torno a la SSI. Sin embargo, la verdadera prueba de cualquier tecnología se encuentra en su aplicación práctica. A continuación, se presentan algunos ejemplos de cómo se puede utilizar la SSI en el mundo real:

2.4.1. Verificación de Edad

La verificación de edad es un caso de uso común en el que la SSI puede ser muy útil. En lugar de tener que compartir su fecha de nacimiento completa, un usuario puede simplemente presentar una Credencial Verificable que demuestra que es mayor de edad [26].

2.4.2. Verificación de Cualificaciones

En la verificación de cualificaciones, las universidades o instituciones educativas pueden emitir Credenciales Verificables para los títulos o certificaciones de los estudiantes. Los empleadores o cualquier entidad interesada pueden verificar estas credenciales de forma rápida y segura [24].

2.4.3. Nuevas tecnologías aplicadas a la SSI

La Intersección de la IA y la SSI

La inteligencia artificial (IA) y la identidad soberana descentralizada (SSI) son dos campos de estudio en rápido desarrollo que pueden tener un impacto significativo en la gestión de la identidad digital. La IA puede ayudar a mejorar la eficiencia y la seguridad de los sistemas de SSI, mientras que la SSI puede proporcionar una base segura y confiable para el desarrollo y la implementación de soluciones de IA [27].

La integración de IoT y SSI

El Internet de las Cosas (IoT) y la identidad soberana descentralizada (SSI) también pueden beneficiarse mutuamente. En un mundo donde cada vez más dispositivos están conectados a Internet, la gestión de la identidad de estos dispositivos se vuelve cada vez más importante.

La SSI puede proporcionar una solución para la gestión segura y eficiente de las identidades de los dispositivos IoT. Por otro lado, los dispositivos IoT pueden proporcionar una fuente valiosa de datos para alimentar los sistemas de SSI y mejorar la eficiencia y la seguridad de estas plataformas.

Quantum Computing y SSI

La aparición de la computación cuántica tiene un efecto dual en la Identidad Soberana Descentralizada (SSI, por sus siglas en inglés). Por un lado, esta innovación tecnológica tiene el potencial de mejorar la seguridad y eficiencia de los sistemas de SSI. Sin embargo, también plantea la amenaza de comprometer los actuales algoritmos criptográficos que mantienen seguros estos sistemas.

Una parte crucial de la SSI, la Prueba de Conocimiento Cero (Zero-Knowledge Proof), se destaca en este contexto. A diferencia de otras operaciones criptográficas, no requiere de una computación de alto rendimiento para sus cálculos. Además, su robustez criptográfica la hace resistente incluso a los intentos de descifrado por computadoras cuánticas, reforzando la seguridad de los sistemas de SSI ante los avances en la computación cuántica.[28]

2.4.4. Desafíos y Oportunidades de la SSI

La identidad soberana descentralizada (SSI) presenta una serie de desafíos y oportunidades. Algunos de los desafíos incluyen la necesidad de crear

una infraestructura confiable para la gestión de identidad, la necesidad de educar a los usuarios y a las entidades sobre cómo utilizar la SSI de manera segura y eficaz, y la necesidad de garantizar la interoperabilidad entre diferentes sistemas SSI.

A pesar de estos desafíos, la SSI también presenta oportunidades significativas. La SSI puede reducir la dependencia de los proveedores de identidad centralizados y puede facilitar la verificación de las identidades digitales.

En resumen, la adopción de la identidad digital auto-soberana puede abordar numerosos problemas asociados a los actuales sistemas de identificación en línea, que abarcan aspectos como la privacidad, la seguridad y el control del usuario sobre sus propios datos. La identidad digital auto-soberana se plantea como una solución potencial y atractiva a los retos contemporáneos de la identidad digital.

La creciente digitalización de nuestras actividades cotidianas y nuestra dependencia de los servicios en línea subraya la necesidad de un sistema de identificación seguro, eficiente y orientado al usuario. La identidad digital auto-soberana y las tecnologías que la habilitan, como la blockchain y las herramientas provistas por proyectos como Hyperledger, representan un paso significativo hacia esa dirección.

En este marco, es imprescindible continuar la investigación y el desarrollo de estas tecnologías, con la meta de alcanzar un mundo donde cada individuo tenga total control sobre su identidad y datos personales.

2.5. Estado del Arte

En el contexto de este trabajo, que se centra en el diseño e implementación de aplicaciones tipo cartera de credenciales utilizando Indy y Aries, se han analizado y comparado varias aplicaciones relevantes en el campo de la identidad digital. Las aplicaciones examinadas incluyen:

- **Trinsic Wallet:** Permite establecer conexiones entre agentes y guardar credenciales. Es sencilla de usar, tiene más de mil descargas, está disponible para Android e iOS, pero es de pago.
- **IdRamp Cello:** Permite establecer conexiones entre agentes y también guardar credenciales. No es sencilla de usar, tiene más de cincuenta descargas, está disponible para Android e iOS y es gratis.
- **Connect.Me:** Permite establecer conexiones entre agentes y no guardar credenciales. Es sencilla de usar, tiene más de diez mil descargas, está disponible para Android e iOS y es gratis.

- **iGrant.io:** Permite establecer conexiones entre agentes pero no guardar credenciales. No es sencilla de usar, tiene más de mil descargas, está disponible para Android e iOS y es gratis.

La Figura 2.9 presenta una comparativa detallada de estas aplicaciones, destacando sus diferencias en términos de funcionalidad, facilidad de uso, disponibilidad y costo. Cada una de ellas representa una aproximación única al desafío de la verificación y emisión de credenciales en el marco del proyecto Hyperledger ARIES.

Figura 2.9: Comparativa de aplicaciones SSI

La revisión de estas aplicaciones ha revelado una variedad de enfoques y soluciones en el campo de la identidad digital. Sin embargo, también ha expuesto áreas donde existen oportunidades para mejoras, como la integración de funcionalidades adicionales y la mejora de la usabilidad. La aplicación desarrollada en este trabajo se propone abordar estos desafíos, ofreciendo una solución integral y fácil de usar que satisface las necesidades de los usuarios y avanza en la tecnología de identidad digital.

Capítulo 3

Objetivos y Planificación

Este capítulo se centra en detallar los aspectos relacionados con la planificación y el presupuesto de este proyecto, además de establecer los objetivos generales y específicos. Comenzaremos delineando los distintos objetivos que esperamos lograr, tanto a nivel general como en términos específicos.

Después, se procede a definir las diversas etapas en las que se ha estructurado el proyecto, proporcionando un desglose detallado del tie mpo asignado a cada fase. Este análisis se complementará con un diagrama que reflejará la secuencia temporal del proyecto.

Por último, se realiza una evaluación exhaustiva de los recursos humanos, el hardware y el software utilizados o necesarios, con el propósito de estimar el presupuesto requerido para la realización del proyecto. Este enfoque integral permitirá apreciar el esfuerzo y los recursos necesarios para la ejecución de este importante proyecto.

3.1. Objetivo General

El objetivo general de este trabajo es el estudio exhaustivo del ecosistema ARIES, con el fin de diseñar e implementar una aplicación Android que funcione como cartera de credenciales verificables. Esta aplicación permitirá tanto generar credenciales válidas como almacenarlas, proporcionando a los usuarios la flexibilidad de operar en el modo que elijan (*Holder* o *Issuer*).

3.2. Objetivos Específicos

En base al objetivo general, se han establecido los siguientes objetivos específicos:

1. Revisión detallada de la literatura y documentación existente sobre el ecosistema ARIES y el proyecto Hyperledger de la *Linux Foundation*.
2. Diseño de la arquitectura de la aplicación Android y funcionalidad de la aplicación.
3. Implementación de las funcionalidades de la aplicación, integrando la tecnología ARIES.
4. Realización de pruebas que demuestren el correcto funcionamiento de la aplicación desarrollada
5. Elaboración de la documentación.

3.3. Planificación

La planificación para lograr estos objetivos se organiza de la siguiente manera:

1. *Fase de estudio (5-7 meses)*: Durante esta fase crítica, se emprende una revisión exhaustiva de la literatura y la documentación disponible relacionada con el ecosistema ARIES y el proyecto Hyperledger de la *Linux Foundation*.

   El proceso de aprendizaje comienza con la exploración de recursos como "Introduction to Hyperledger" [29] y "Becoming an Hyperledger" [30], que proporcionan una base sólida pero limitada en cuanto a la comprensión de los conceptos esenciales.

   La documentación clave que se encuentra en el repositorio "Aries Framework JavaScript" [31] es valiosa, pero se ve empañada por dificultades en la instalación y la falta de claridad en las instrucciones. Las incompatibilidades con NodeJS y React generan complicaciones técnicas importantes, y la actualización constante del proyecto en GitHub complica aún más el proceso.

   La búsqueda de soluciones alternativas conduce a la exploración de diversos repositorios, como "aries-agent-test-harness" [32], "aries-cloudagent-python" [33], "aries-acapy-controllers" [34], "aries-mobile-agent-xamarin" [35] y "aries-mediator-service" [36]. Sin embargo, muchos de estos repositorios resultan incompatibles con el enfoque móvil deseado.

   En resumen, la fase de estudio inicialmente planificada para comprender y asentar las bases del proyecto Hyperledger y ARIES se ve afectada por una serie de desafíos técnicos y de compatibilidad.

2. *Fase de diseño (1 mes)*: En esta fase, se diseñará la arquitectura de la aplicación Android.

3. *Fase de implementación (2-3 meses)*: Esta fase incluirá la implementación de las funcionalidades para la generación y almacenamiento de credenciales.

4. *Fase de prueba (1 mes)*: En esta última fase, se probará la aplicación en diferentes escenarios y contextos.

Se prevé que el proyecto tenga una duración total de 9 a 12 meses como se muestra en la Tabla 3.1 y en la Figura 3.1, pero este plazo puede variar dependiendo de los obstáculos y desafíos que puedan surgir durante la fase de implementación.

Planificación		
Fases	Descripción	Tiempo (horas)
1	Revisión de literatura y documentación	110
2	Diseño de la arquitectura de la aplicación	35
3	Implementación de funcionalidades	90
4	Pruebas en diferentes escenarios y contextos	40
5	Documentación	80
Total		355

Tabla 3.1: Tiempo invertido en el proyecto

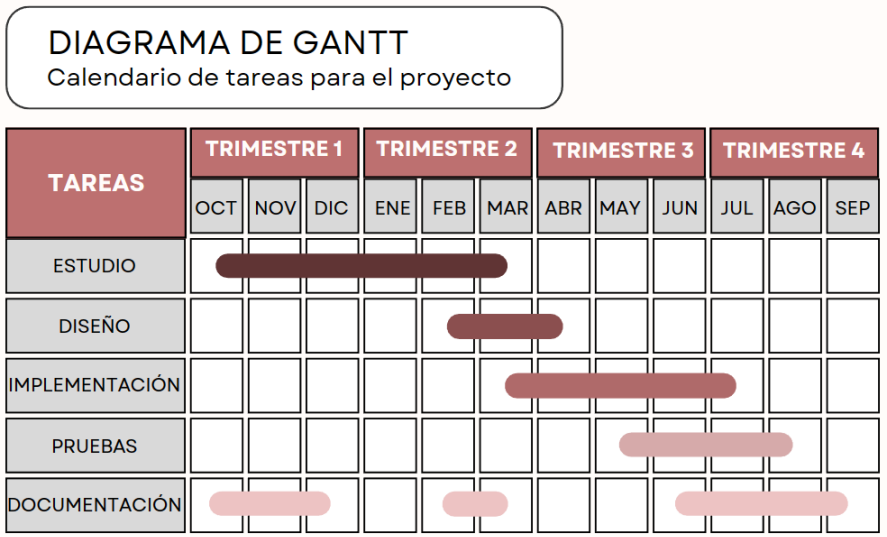

Figura 3.1: Diagrama de Gantt seguido para el desarrollo del proyecto

3.4. Recursos y Presupuesto

En este apartado se describirán los distintos recursos utilizados durante todo el desarrollo del trabajo

3.4.1. Recursos Humanos

Los gastos asociados con los recursos humanos empleados en este proyecto se determinarán en base al total de horas dedicadas a su desarrollo.

Tutor: El profesor Rafael Alejandro Rodríguez Gómez, miembro del departamento de Teoría de la Señal, Telemática y Comunicaciones en la Universidad de Granada, se asigna un valor estimado de 100 euros por hora (incluyendo costos asociados).

Estudiante: Como estudiante del Grado en Ingeniería Informática, se estima una tarifa de 40 euros por hora.

Rol	**Horas**	**Costo por hora (€/h)**	**Coste (€)**
Tutor	20	100	2000
Estudiante	355	40	14200
Total			16200

Tabla 3.2: Costos de los recursos humanos.

El costo total del proyecto es de 16,200 euros y se calcula en la Tabla 3.2.

3.4.2. Recursos Hardware

Para la realización de este proyecto solo ha sido necesario contar con un Único equipo como se indica en la Tabla 3.3, tanto para el diseño y desarrollo del software, así como para su despliegue y las mediciones tomadas.

Ordenador personal del alumno

Recurso	**Coste (€)**
Ordenador personal	1200
Total	1200

Tabla 3.3: Costos de los recursos hardware.

3.4.3. Recursos Software

Para realizar este proyecto, seleccionamos un conjunto de herramientas de software en función de las necesidades específicas del trabajo. Canva se utilizó por su facilidad de uso en la creación de gráficos y diseños; Notion, para gestionar y organizar eficientemente el proyecto; Java y Android Studio, debido a su robustez y amplio soporte para el desarrollo de aplicaciones; ACA-Py, para manejar agentes Aries; Docker, para el despliegue de aplicaciones en contenedores, garantizando así la reproducibilidad y la portabilidad; Postman, para las pruebas de APIs; y VON-Network, para desplegar una serie de nodos Indy a nivel local y realizar pruebas en una red Indy.

Recurso	**Coste (€)**
Canva	-
Notion	-
Java	-
Docker	-
Android Studio	-
Postman	-
ACA-Py	-
VON-Network	-
Total	0

Tabla 3.4: Costos de los recursos hardware.

En la Tabla 3.4 se aprecia el uso exclusivo de herramientas de software de código abierto, lo que ha permitido mantener los costos al mínimo.

3.4.4. Presupuesto Total

El presupuesto total del proyecto se muestra en la Tabla 3.5, haciendo la suma de todos los distintos recursos utilizados, asciende a los **19.200** euros.

Recurso	**Coste (€)**
Humanos	16200
Hardware	1200
Software	0
Total	17400

Tabla 3.5: Costo total del proyecto.

Capítulo 4

Diseño

Este capítulo describe el enfoque de diseño adoptado en el desarrollo de la aplicación de identidad auto-soberana (*SSI*) basada en agentes Aries. Se discuten las decisiones de diseño clave, incluida la elección de la metodología de desarrollo de software, la arquitectura de los agentes Aries y las tecnologías utilizadas. Además, se presentan los requisitos funcionales y no funcionales de la aplicación, así como los casos de uso que guían el diseño y desarrollo del sistema. A través de este capítulo, se proporciona una comprensión clara de cómo se diseña la aplicación para cumplir con los objetivos y requisitos del proyecto.

Dentro del ámbito de este proyecto, es crucial entender la estrategia de desarrollo de software adoptada. Según la Real Academia de la Lengua, la metodología es el "Conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal" [37]. En el ámbito del desarrollo de software, esta descripción se traduce en un conjunto de técnicas y fundamentos de ingeniería que facilitan la creación de software de manera coste-efectiva, confiable y que opere eficientemente en sistemas reales.

4.1. Enfoque Secuencial o Cascada

El enfoque secuencial, también conocido como modelo en cascada, es una metodología de desarrollo de software que sigue un flujo de trabajo lineal y ordenado. En este enfoque, el desarrollo de software se divide en fases secuenciales, donde cada fase depende de la finalización de la fase anterior. Las fases típicas incluyen el análisis de requisitos, diseño, implementación, verificación y mantenimiento como se muestra en la Figura 4.1.

Este enfoque es adecuado para proyectos donde los requisitos son bien conocidos y es poco probable que cambien durante el desarrollo. A continuación, se discutirán las fortalezas y limitaciones del enfoque en cascada,

así como su aplicación en este proyecto.

Las **fortalezas** del enfoque en cascada incluyen:

- Captura temprana de requisitos, lo que facilita al equipo establecer el propósito del proyecto, planificar de manera integral y esbozar la solución.
- Optimización del uso de recursos al segmentar o combinar tareas según las habilidades del equipo.
- Una estructura definida que clarifica los requisitos y objetivos.
- Documentación detallada de procesos y resultados.
- Facilidad para evaluar el progreso del proyecto con una planificación detallada.

Las **limitaciones** del enfoque en cascada son:

- La complejidad de establecer requisitos al comienzo, dado que el cliente puede no tener una visión clara y los requisitos pueden evolucionar.
- La necesidad de una clara segmentación de tareas y objetivos, lo que puede ser un desafío dada la experiencia o recursos al inicio.
- Dificultades para determinar el progreso entre fases.
- La demora en obtener un producto funcional hasta las etapas finales.
- El riesgo creciente de desviaciones en tiempos, costos y expectativas a medida que el proyecto se extiende, debido a la tendencia a abarcar demasiado (conocido también como enfoque "big bang").

La elección de esta estrategia sobre metodologías ágiles se basó en el tamaño reducido del equipo de desarrollo (un solo miembro), la detallada documentación en el informe, la estructura organizada y la implementación.

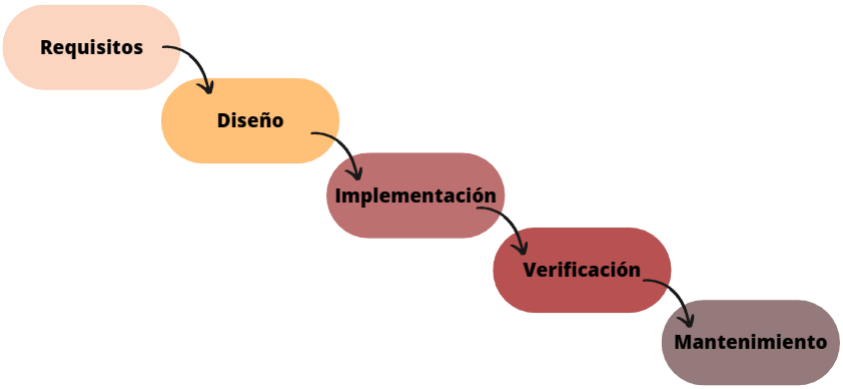

Figura 4.1: Modelo en cascada

4.2. Diseño y Desarrollo de Agentes Aries

Los agentes Aries actúan como intermediarios seguros en la comunicación entre diferentes partes y la blockchain Indy, una especie de cartera de registros digital. Estos agentes manejan información sensible como las Credenciales Verificables (*VCs*) que se detallan en la sección 2.2.6, y otros datos relacionados con la identidad y conexiones entre agentes.

Imagina la arquitectura de un agente como una máquina compleja. El proyecto Aries ha hecho esta máquina más sencilla, eliminando partes innecesarias y dejando solo lo esencial. En la Figura 4.2 se ilustra esta simplificación. Dentro de esta máquina, hay algo llamado *framework* que tiene todas las habilidades necesarias para que los agentes hagan su trabajo, como interactuar con las *VCs* y almacenar DIDs como se comenta en la sección 2.2.7.

Sin embargo, este *framework* es como un robot sin control remoto; tiene todas las capacidades, pero no sabe cuándo o cómo usarlas. Aquí es donde entra el *controlador*, un software que le dice al agente qué hacer en cada situación. Es como el cerebro de la operación, guiando al agente en sus tareas. Los desarrolladores de agentes Aries ahora solo necesitan enfocarse en este controlador, haciendo su trabajo más sencillo y directo.

En el marco de este proyecto, se llevará a cabo el desarrollo de varios controladores de agentes Aries, desempeñando roles específicos como Issuer y *Holder* en diferentes contextos. Estos controladores utilizarán el *framework* Hyperledger *Aries Cloud Agent Python*, desarrollado en Python.

El *framework* se comunica con el controlador a través de la REST Application Programming Interfaces (API), un protocolo de comunicación que

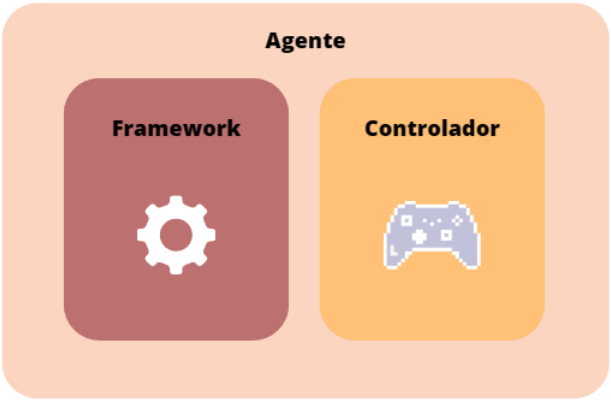

Figura 4.2: Simplificación de los componentes de un agente

se define por [38]:

- **Arquitectura cliente-servidor:** Facilita la comunicación entre el cliente (controlador) y el servidor (ACA-Py), donde el servidor ofrece una API con acceso a sus funciones y datos.
- **Protocolo HTTP:** Utiliza los métodos POST y GET para enviar órdenes y obtener información, respectivamente.
- **Formato de intercambio de datos:** Emplea JavaScript Object Notation (JSON) para la transferencia de datos.
- **Operaciones CRUD:** Permite realizar operaciones de Crear, Leer, Actualizar y Eliminar sobre los recursos.

Adicionalmente, este proyecto ha sido desarrollado utilizando el lenguaje de programación Java en el entorno de Android Studio.

La elección de Java, un lenguaje ampliamente utilizado en el desarrollo de aplicaciones Android, se basa en su robustez, eficiencia y amplia comunidad de desarrolladores [39, 40]. Estudios comparativos han respaldado esta elección, destacando las ventajas de Java en comparación con otros lenguajes como Kotlin [41].

En el contexto de la identidad auto-soberana (*SSI*), los agentes Aries actúan en nombre de un usuario para interactuar con otros agentes y sistemas. La figura 4.3 ilustra esta interacción.

Los agentes pueden ser de dos tipos: *Holder* e *Issuer*. Un *Holder* solicita y gestiona credenciales en nombre de un usuario, mientras que un *Issuer* emite credenciales a los *Holders*.

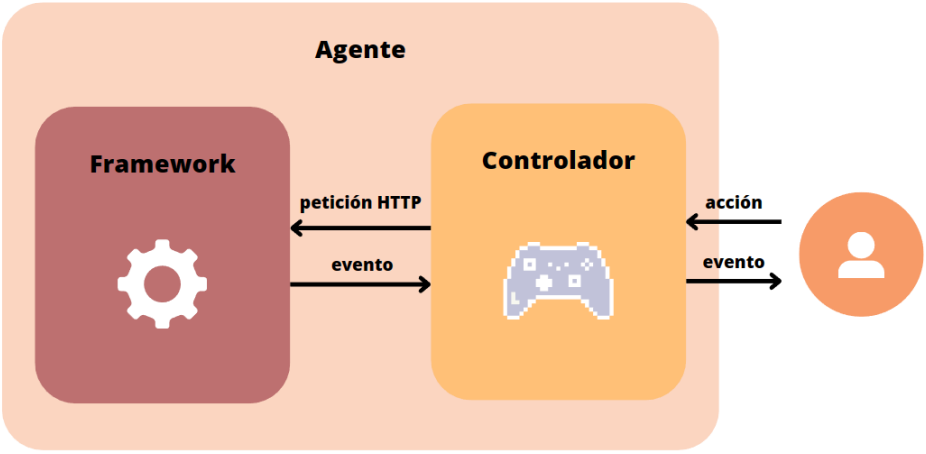

Figura 4.3: Interacción entre los elementos del agente

La arquitectura del controlador y la elección de tecnologías son fundamentales para facilitar estas interacciones.

A continuación, se explorarán los requisitos y casos de uso que definen las expectativas y restricciones del sistema, cruciales para entender cómo la aplicación debe ser diseñada y desarrollada.

4.2.1. Requisitos

Para garantizar el correcto funcionamiento y la eficiencia de la aplicación, es esencial definir tanto los requisitos funcionales como los no funcionales. Los primeros se refieren a las funcionalidades específicas que la aplicación debe ofrecer, mientras que los segundos se centran en las características generales y las restricciones que debe cumplir.

Requisitos Funcionales

La aplicación tiene dos modos principales de operación: *Holder* e *Issuer*. Cada modo tiene sus propias funcionalidades:

1. **Modo Holder:**
   - Verificar la existencia de una conexión con el agente *Issuer*.
   - Establecer una conexión con el agente *Issuer* mediante una invitación.
   - Proponer credenciales al agente *Issuer*.

- Eliminar propuestas de credenciales o credenciales.
- Solicitar credenciales especificando atributos y el esquema correspondiente.

2. **Modo Issuer:**
   - Generar y visualizar invitaciones en texto o código QR.
   - Consultar la última invitación generada.
   - Gestionar el proceso de emisión de credenciales: desde la recepción de propuestas hasta la emisión final.
   - Generar esquemas de credenciales con atributos específicos sin requerir interacción del *Holder*.

Requisitos No Funcionales

1. **Seguridad:** La aplicación debe garantizar la privacidad y seguridad de los datos de los usuarios, especialmente las credenciales y propuestas.
2. **Usabilidad:** La interfaz de usuario debe ser intuitiva y fácil de usar.
3. **Eficiencia:** Los tiempos de respuesta de la aplicación deben ser rápidos y eficientes.

4.3. Casos de Uso

Los casos de uso describen las interacciones típicas entre los usuarios y la aplicación. A continuación, se presentan los principales casos de uso, acompañados de diagramas de flujo o elementos gráficos para una mejor comprensión:

4.3.1. Caso de Uso: Generar Esquema de Credenciales

- **Actor principal:** *Issuer*
- **Precondiciones:** El *Issuer* ha iniciado la aplicación y seleccionado el modo *Issuer*.
- **Flujo principal:** El *Issuer* genera un esquema de credenciales con atributos determinados y un nombre de esquema.
- **Postcondiciones:** El esquema de credenciales está disponible para su uso en la solicitud y emisión de credenciales.

En todos los escenarios se realiza la emisión de credenciales por parte de un *Issuer*, por lo que este paso será común para todos los *Issuers* de los distintos casos de uso.

Para que un *Issuer* pueda emitir una credencial a un *Holder*, es necesario que el *Issuer* registre en el *ledger* un esquema y su definición, como se comentó en la Sección 2.2.2

Figura 4.4: Interfaz de usuario en modo Issuer para generar un esquema de credenciales

La aplicación Android permite elegir la cantidad de atributos, además del nombre de estos y el nombre que se da al esquema de credenciales para ser registrado en el *ledger* a través de una interfaz como se muestra en la Figura 4.4

4.3.2. Caso de Uso: Establecer Conexión entre Holder e Issuer

- **Actor principal:** *Holder*
- **Precondiciones:** El *Holder* ha iniciado la aplicación y seleccionado el modo *Holder*.
- **Flujo principal:** El *Holder* acepta una invitación del *Issuer* para establecer una conexión.

- **Postcondiciones:** Se establece una conexión entre el *Holder* y el *Issuer*.

En el proceso de establecimiento de conexión, dos agentes están involucrados, a diferencia de la creación y registro del esquema y de la definición de la credencial, en los que solo un agente interactúa con el ledger.

Antes de que dos agentes puedan intercambiar mensajes, emitir credenciales mutuamente o verificarlas, necesitan establecer una conexión para comunicarse. Esto se define en el Request for Comments (RFC) de Aries 0160: Connection Protocol [42].

Figura 4.5: Interfaz de usuario en modo Issuer para generar una invitación de conexión

El *framework* de Aries permite que, una vez recibida la invitación, el proceso de aceptación y confirmación de la conexión se realice automáticamente en el *framework* (ACA-Py), sin la intervención del controlador. Esto acelera el procedimiento, ya que ACA-Py no tiene que consultar al controlador cómo responder a todos los eventos generados durante la creación del

canal de comunicación. A continuación se describe el procedimiento seguido para establecer la conexión, aunque en la implementación se ha decidido que este proceso se realice automáticamente por el *framework*.

El primer paso es la generación de una invitación por parte de cualquiera de los agentes. La invitación consiste en datos en texto plano que contienen un identificador único e información similar a una clave pública y una URL. La invitación puede enviarse por cualquier medio, como correo electrónico, página web, mensajería instantánea, etc. Al ser datos en texto plano, se puede enviar en ese mismo formato, o incluso a través del escaneo de un código QR que se muestra en la interfaz de usuario.

En la Figura 4.5 se muestra la interfaz de usuario en modo *Issuer* con la que se puede solicitar la creación de una invitación y los dos tipos de salida obtenidas a través de la función */connections/receive-invitation* de ACA-Py.

Figura 4.6: Interfaz de usuario en modo Holder para aceptar una invitación de conexión

Una vez que el agente invitado tiene la invitación, puede usar la información contenida en esta para establecer la conexión entre agentes.

Si el agente invitado acepta la invitación como se muestra en la Figura 4.6, creará un nuevo DID para la nueva relación con el agente que envió

la invitación.

Finalmente, creará en su almacenamiento seguro un identificador de conexión y un registro que contendrá toda la información que recopila sobre la conexión. Ahora, los dos agentes están conectados a través de un canal seguro, privado y con encriptación de extremo a extremo.

En la Figura 4.7 se muestra el diagrama de flujo del proceso de establecimiento de una conexión.

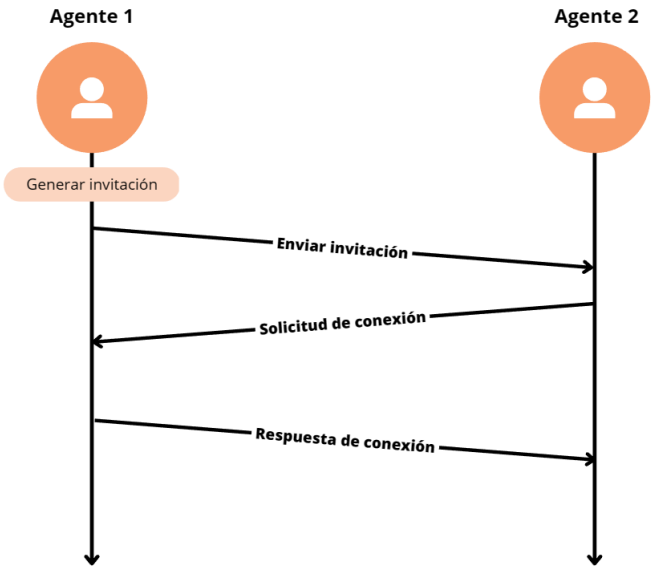

Figura 4.7: Diagrama de flujo del establecimiento de una conexión

4.3.3. Caso de Uso: Solicitar Credenciales

- **Actor principal:** *Holder*
- **Precondiciones:** El *Holder* ha establecido una conexión con el *Issuer* y el *Issuer* ha generado un esquema de credenciales.
- **Flujo principal:** El *Holder* solicita credenciales especificando el nombre y valor de cada atributo y el nombre del esquema.
- **Postcondiciones:** El *Issuer* recibe la solicitud de credenciales con los atributos y el nombre del esquema especificados.

4.3.4. Caso de Uso: Emitir Credenciales

- **Actor principal:** *Issuer*

- **Precondiciones:** El *Issuer* ha recibido una propuesta de credenciales con valores específicos para los atributos del *Holder*.

- **Flujo principal:** El *Issuer* emite los credenciales solicitados con atributos específicos.

- **Postcondiciones:** El *Holder* recibe los credenciales emitidos.

Una vez que se ha establecido la conexión entre dos agentes, pueden comenzar a intercambiar mensajes. A continuación, se describe el proceso de emisión de una credencial, según el protocolo de emisión de credenciales de Aries RFC 0036: Issue Credential Protocol 1.0 [43].

En la Figura 4.10 se muestra el diagrama de flujo del proceso completo de la expedición de una credencial.

El proceso comienza cuando el *Holder* envía una propuesta de credencial al *Issuer*, indicando cómo le gustaría que fuera la credencial. En esta propuesta, el *Holder* puede especificar los atributos que espera recibir en la credencial. Al enviar la propuesta al framework con la función */issue-credential/send-proposal*, ACA-Py se encargará de enviar el mensaje al agente con el que se mantiene la conexión.

Cuando el *Issuer* recibe la propuesta de credencial, verifica si puede emitir la credencial con la información solicitada por el *Holder*. Si es así, el *Issuer* responde con una oferta de credencial, confirmando los atributos que enviará al *Holder*.

A continuación, el *Holder*, al recibir la oferta de credencial y estar satisfecho con la información enviada por el *Issuer*, confirma que desea recibir la credencial enviando un mensaje de solicitud de credencial al *Issuer*.

Una vez que el *Issuer* recibe la confirmación del *Holder*, envía la credencial esbozada en el mensaje de oferta de credencial. Finalmente, el *Holder* almacena la *VC* en su cartera digital y envía un mensaje de confirmación de recepción (ACK) al *Issuer*.

En la Figura 4.8 se muestra un credencial tras ser almacenado, el contenido en texto plano y el estado del mismo en el modo *Holder*.

En la Figura 4.9 se muestra un credencial tras ser emitido, el nombre del esquema utilizado, el contenido en texto plano y el estado del mismo en el modo *Issuer*.

En cada tipo de controlador, se ha programado un módulo de *Listeners* para recibir actualizaciones de los estados de cada mensaje y tanto mostrar, como habilitar opciones en cada uno de los modos disponibles (*Issuer* o *Holder*) para que el usuario pueda decidir qué hacer en cada momento.

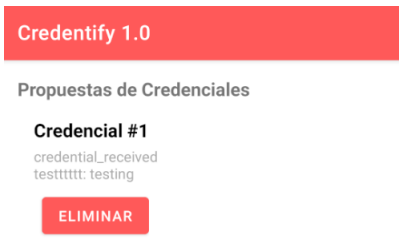

Figura 4.8: Interfaz de usuario tras almacenar un credencial - modo Holder

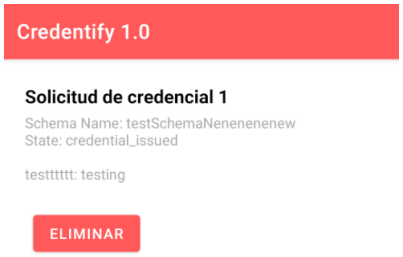

Figura 4.9: Interfaz de usuario tras enviar un credencial - modo Issuer

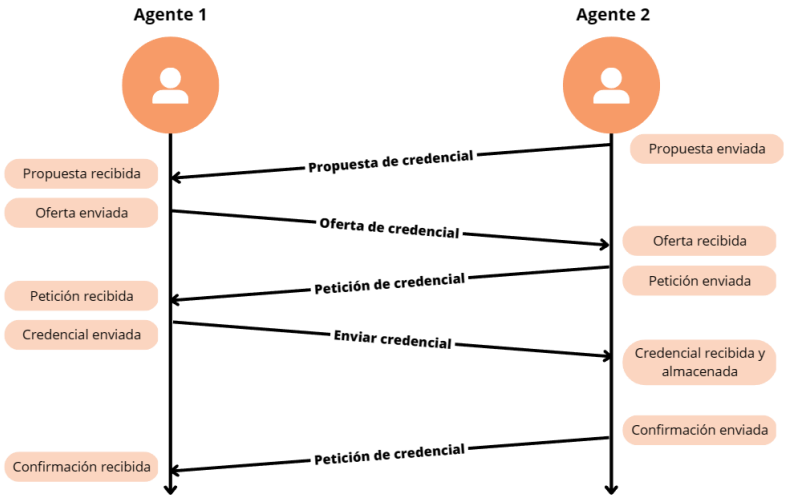

Figura 4.10: Diagrama de flujo del proceso de emisión de una credencial

Es importante mencionar que cuando se indica que cualquiera de los agentes recibe un mensaje, en realidad es el framework del agente el que lo recibe y el controlador de cada agente es quien se mantiene a la escucha del

cambio de estado de las ofertas y peticiones.

Capítulo 5

Implementación

La implementación de este proyecto requiere la combinación de diversas herramientas y tecnologías para lograr un sistema cohesivo y funcional. A continuación, se presenta una breve descripción de las herramientas utilizadas, así como las fuentes de donde se extrajo la información pertinente para su uso.

5.1. Herramientas y Tecnologías

5.1.1. ACA-Py (Aries Cloud Agent - Python)

ACA-Py es un agente de nube basado en el proyecto Hyperledger Aries. Está diseñado para ser utilizado por organizaciones para construir aplicaciones que interactúen en redes de agentes Aries, permitiendo la emisión, prueba y administración de credenciales verificables. Este término se introduce en la Sección 4.2

5.1.2. VON-Network

Para llevar a cabo las pruebas en este proyecto, se requiere el uso de una blockchain Indy. Sin embargo, el uso de redes de producción como Sovrin no es una opción viable, ya que se necesitan permisos para escribir en el ledger. Para superar esta limitación, se optó por desplegar una red Indy de manera local, lo que permite tener control total sobre la red y realizar las operaciones necesarias sin depender de terceros.

La solución más sencilla para desplegar una red Indy local es hacer uso de la VON-Network (Verifiable Organizations Network), una red blockchain basada en Hyperledger Indy, pre-empaquetada y mantenida por el gobierno de Columbia Británica. Esta red permite desplegar una red Indy de cuatro

nodos de manera rápida y sencilla utilizando contenedores de Docker. La VON-Network se puede obtener directamente desde su repositorio de GitHub [44].

El despliegue de la VON-Network es sumamente sencillo y solo requiere la ejecución de dos comandos para poner en funcionamiento la red Indy de cuatro nodos como se muestra en el Listado de código 5.1.

```
git clone https://github.com/bcgov/von-network.git
cd von-network
./manage build
./manage start
```

Listado de código 5.1: Comandos para clonar y ejecutar von-network

Una vez desplegada la red, se crea automáticamente el archivo Génesis del ledger. Este archivo es esencial para la operación de la red, ya que contiene información sobre los nodos de la red Indy, incluyendo direcciones IP y puertos, así como el material criptográfico necesario para comunicarse de manera segura con dichos nodos.

El archivo Génesis es necesario para que los agentes sepan cómo pueden contactar con los nodos de la red Indy aunque de todo ese proceso se encargará internamente el *framework* ACA-Py.

5.1.3. Docker

Docker es una plataforma que utiliza contenedores para facilitar el desarrollo y la ejecución de aplicaciones. Se trata de una herramienta de código abierto escrita en Golang, que automatiza el despliegue de aplicaciones dentro de contenedores. Estos contenedores empaquetan las aplicaciones junto con su entorno de ejecución, es decir, todos los archivos necesarios para que la aplicación funcione. Así, Docker crea contenedores que aseguran que la aplicación funcione correctamente en cualquier entorno en el que se utilice [45].

El proyecto Aries sugiere desplegar ACA-Py como un contenedor Docker. Siguiendo esta recomendación, todas las instancias del framework se han desplegado como contenedores usando ACA-Py en la versión **0.6.0**. El código necesario para desplegar instancias de este framework se puede obtener del repositorio de Github de ACA-Py.

En el código mostrado a continuación, se ilustra cómo iniciar dos instancias del framework, una que actuará como emisor (*Issuer*) y otra como titular (*Holder*).

```
git clone https://github.com/hyperledger/aries-cloudagent-python.git
cd ./aries-cloudagent-python/scripts
git checkout tags/0.6.0

# Lanzar contenedor ACA-Py Issuer
PORTS="8000 8001" ./run_docker start -l Issuer -it http 0.0.0.0 8000 -
    ot http --admin 0.0.0.0 8001 --admin-insecure-mode -e http://host.
    docker.internal:8000 --genesis-url http://host.docker.internal
    :9000/genesis --wallet-type indy --wallet-name issuerwallet --
    wallet-key issuerkey --log-level 'info' --auto-provision --auto-
    accept-invites --auto-accept-requests --auto-ping-connection --
    webhook-url http://localhost:11001/webhooks --tails-server-base-
    url https://71af-87-222-0-67.ngrok.io  --seed
    testing12300000000000000000000

# Lanzar contenedor ACA-Py Holder
PORTS="10000 10001" ./run_docker start -l Holder -it http 0.0.0.0
    10000 -ot http --admin 0.0.0.0 10001 --admin-insecure-mode -e http
    ://host.docker.internal:10000 --genesis-url http://host.docker.
    internal:9000/genesis --debug-connections --webhook-url http://
    host.docker.internal:12001/webhooks --wallet-type indy --wallet-
    name holderwallet --wallet-key holderkey --log-level 'info' --auto
    -provision --auto-accept-invites --auto-accept-requests --auto-
    ping-connection --tails-server-base-url https://71af-87-222-0-67.
    ngrok.io --wallet-local-did
```

Listado de código 5.2: Comandos para clonar y ejecutar ACA-Py

A continuación se define la utilidad de alguno de los parámetros más relevantes al iniciar una instancia de ACA-Py:

- `-it` *(inbound-transport)*: Indica el protocolo, dirección y puerto en los que la instancia de ACA-Py estará disponible para comunicarse con otros agentes Aries. En este caso, se utiliza el protocolo HTTP en la dirección 0.0.0.0 y el puerto 8000.
- `-ot` *(outbound-transport)*: Especifica el protocolo que el framework utilizará para comunicarse con otros agentes Aries. En este caso, se utiliza el protocolo HTTP.
- `--admin`: Define la dirección y el puerto a través de los cuales el controlador puede comunicarse con la API del framework.
- `--admin-insecure-mode`: Permite acceder a la API del framework sin proporcionar una clave. Se utiliza en este caso porque el objetivo es realizar pruebas en un entorno controlado.
- `--genesis-url`: Indica la dirección en la que se encuentra el archivo Génesis del ledger.
- `--auto-accept-invites`: Acepta automáticamente las invitaciones sin disparar un evento de webhook o esperar una solicitud de administrador.

Como resultado de la ejecución de ambos agentes en diferentes terminales obtendremos un esquema similar al que se muestra en Figura 5.1

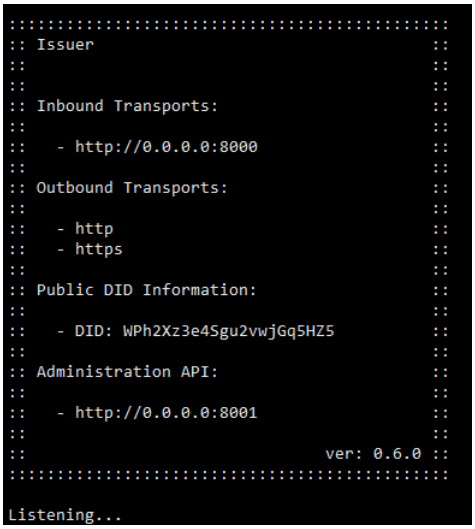

Figura 5.1: Ejecución exitosa de un agente ACA-Py V0.6.0 como Issuer

Ahora, también se puede acceder a la URL: {`Tu IP pública`}`:8001`, y si tienes éxito, verás una larga lista de APIs abiertas definidas para el agente ACA-Py como se muestra en Figura 5.2.

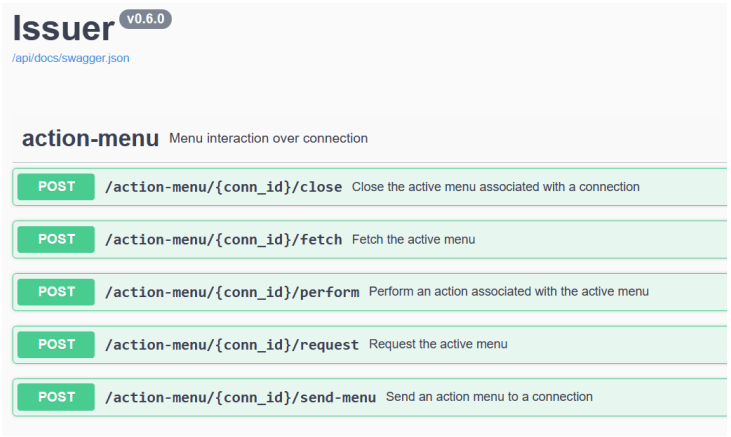

Figura 5.2: APIs abiertas definidas para el agente Issuer

5.1.4. Postman

Postman es una herramienta esencial para el desarrollo y prueba de APIs, que permite diseñar, simular, probar y documentar APIs de forma eficiente. Es especialmente útil para probar servicios RESTful, ya que facilita la validación y el análisis de respuestas [46].

En este proyecto, se utiliza Postman para interactuar con la API de los agentes ACA-Py y analizar las respuestas. Esto permite verificar la interacción adecuada con el ledger y confirmar que los resultados se registran correctamente en el ledger o en la cartera correspondiente.

Como se muestra en la Figura 5.3, se ha creado una colección en Postman que clasifica las peticiones a la API según si se dirigen al *Issuer* o al *Holder*. Esta organización agiliza el proceso de prueba, ya que permite ejecutar peticiones específicas para cada rol de manera más eficiente.

La utilización de Postman en este proyecto es crucial para garantizar que las operaciones de los agentes, como la emisión y verificación de credenciales, se realicen de manera adecuada. Además, ayuda a identificar y resolver rápidamente cualquier problema en la interacción con la API.

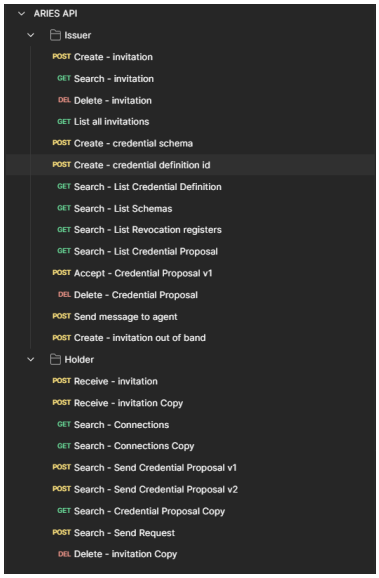

Figura 5.3: Colección Postman para validar funcionalidades de agentes ARIES

5.1.5. Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones Android. Proporciona una serie de herramientas que facilitan la creación, prueba y depuración de aplicaciones para Android en múltiples dispositivos y configuraciones [47].

Con estas herramientas y tecnologías como base, se procedió a la implementación específica del proyecto.

Capítulo 6

Validación y pruebas

En este capítulo se describirá el proceso que se ha seguido para llevar a cabo las pruebas de rendimiento, así como los resultados que se han obtenido a partir de la ejecución de estas pruebas.

6.1. Pruebas realizadas

En este trabajo, se han llevado a cabo pruebas centradas en las tres funcionalidades descritas en la Sección 4.3. Cada uno de estos escenarios de prueba se ha diseñado para evaluar aspectos específicos del sistema.

En el primer escenario, se evalúa la capacidad del sistema para generar un esquema de credenciales válido utilizando el modo *Issuer*. En este caso, se añaden atributos definidos por el usuario al esquema, y se espera que el nombre del identificador aparezca registrado en el ledger como confirmación de que el proceso se ha completado con éxito.

El segundo escenario involucra la interacción entre dos agentes, uno actúa como *Issuer* y el otro como *Holder*. El objetivo de este escenario es evaluar el proceso de establecimiento de conexión entre los agentes. Para ello, el agente *Issuer* crea una invitación que el agente *Holder* debe aceptar. El diagrama de flujo que representa este proceso de prueba se muestra en la Figura 4.7.

Finalmente, en el tercer escenario, se examina la gestión de credenciales por parte de cada agente después de haber establecido una conexión entre ellos. En este caso, se vuelve a utilizar un agente en modo *Issuer* y otro en modo *Holder*. El objetivo es evaluar cómo los agentes manejan las credenciales una vez que la conexión ha sido establecida.

6.2. Resultados

Cabe destacar que se desplegaron el *framework* ACA-Py y el controlador (para cada agente) en contenedores Docker separados. Esta configuración facilitó el uso y análisis independiente de cada agente.

Adicionalmente, en cada escenario se utilizó la herramienta Postman para verificar la interacción con el ledger y confirmar que los resultados se registraron adecuadamente en el ledger o en la cartera correspondiente.

6.2.1. Escenario 1: Generación de un esquema de credenciales válido

Este primer escenario de pruebas se centra en la capacidad del sistema para generar un esquema de credenciales válido utilizando el modo *Issuer* del agente ACA-Py. En este caso, el agente *Issuer* interactúa únicamente con el ledger para asegurar la validez del esquema.

Descripción del Escenario:

- **Generación de Esquema de Credenciales:** En este escenario, el agente *Issuer* genera un esquema de credenciales válido. Se añaden atributos definidos por el usuario al esquema de credenciales como se muestra en la Figura 6.1.
- **Registro del Identificador:** Una parte esencial de este escenario es verificar que el identificador asociado al esquema de credenciales se registre correctamente en el ledger. Este registro es una confirmación de que el proceso se ha completado con éxito y que el esquema de credenciales generado es válido.

Intercambio de Mensajes:

1. El agente *Issuer* inicia el proceso enviando una solicitud al ledger para crear un nuevo esquema de credenciales, especificando los atributos que desea que contenga el esquema.
2. El ledger responde proporcionando un identificador único para la definición del esquema de credenciales.
3. El agente *Issuer* utiliza este identificador para mostrar al usuario cómo la definición del credencial se guarda exitosamente.

A continuación, se presenta un proceso detallado de los pasos que se siguen a partir de la interacción de un usuario con la aplicación. Este proceso comienza con la creación del esquema de credencial en el agente *Issuer*, como se ilustra en la Figura 6.1 y se comprueba en la Figura 6.2 . Una vez que el esquema se ha registrado en el agente *Issuer*, se procede a su registro en la *VON-Network*, como se muestra en la Figura 6.3.

Posteriormente, en la Figura 6.4, se detalla cómo se almacena un identificador para esa definición de credencial y se comprueba el registro en la *VON-Network* en la Figura 6.5. Es relevante destacar que, debido a la simplicidad del proceso y el intercambio de mensajes en este escenario, no se ha proporcionado una representación gráfica en forma de diagrama de flujo.

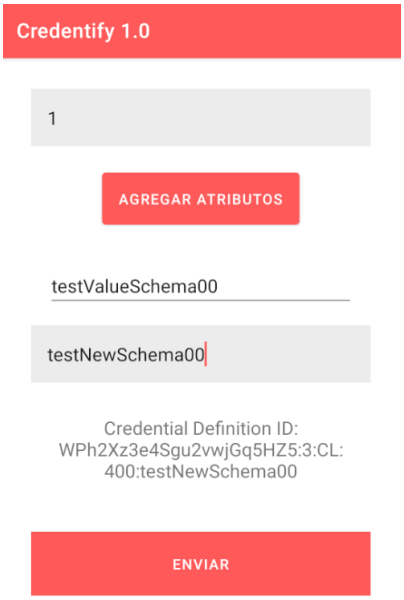

Figura 6.1: Esquema de credencial creado en modo Issuer

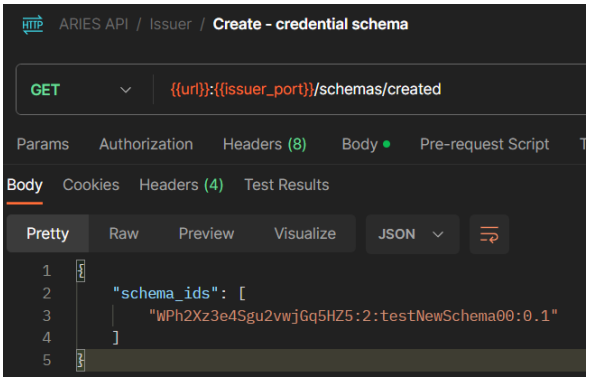

Figura 6.2: Esquema de credencial registrado en agente Issuer

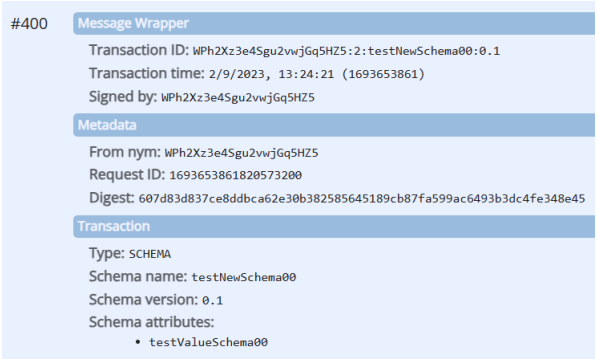

Figura 6.3: Esquema de credencial registrado en VON Network

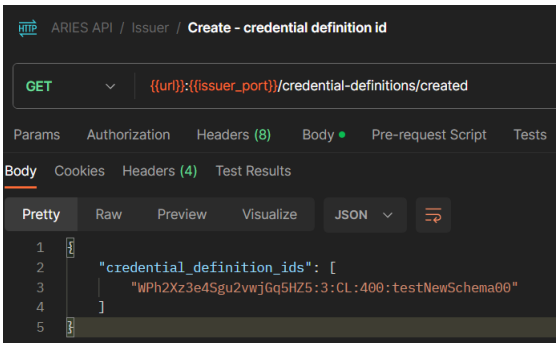

Figura 6.4: Definición de credencial registrado en agente Issuer

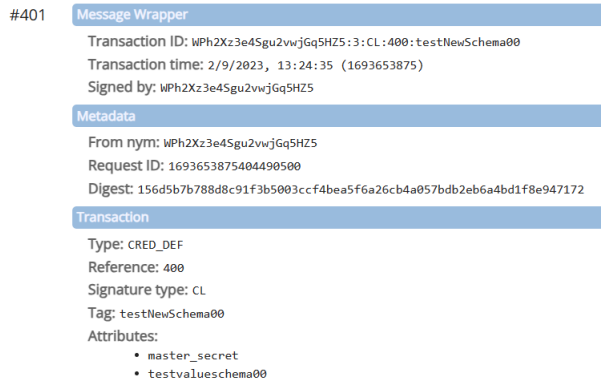

Figura 6.5: Definición de credencial registrado en VON Network

Este proceso se caracteriza por su eficiencia y claridad en la interacción, lo que lo convierte en una representación sencilla sin necesidad de un diagrama de flujo adicional

6.2.2. Escenario 2: Conexión entre agentes

Este segundo escenario de pruebas se enfoca en evaluar la capacidad del sistema para establecer una conexión efectiva entre dos agentes, uno actuando como *Issuer* y el otro como *Holder*.

Descripción del Escenario:

- **Conexión entre Agentes:** En este escenario, el agente *Issuer* y el agente *Holder* deben establecer una conexión efectivas.
- **Creación de Invitación:** El agente *Issuer* crea una invitación para que el agente *Holder* se una a la conexión.
- **Aceptación de la Invitación:** El agente *Holder* acepta la invitación del *Issuer*, lo que inicia el proceso de conexión entre los dos agentes.
- **Intercambio de Mensajes:** Una vez que la conexión está establecida, los agentes pueden intercambiar mensajes relacionados con la emisión de credenciales. Esto incluye la negociación de los atributos de la credencial y otros detalles relevantes.

Intercambio de Mensajes:

1. El *Issuer* crea una invitación para el *Holder*, incluyendo detalles necesarios para la conexión como se muestra en la parte izquierda de Figura 6.6.

2. El *Holder* acepta la invitación, lo que inicia la conexión entre ambos agentes como se muestra en la parte derecha de Figura 6.6.

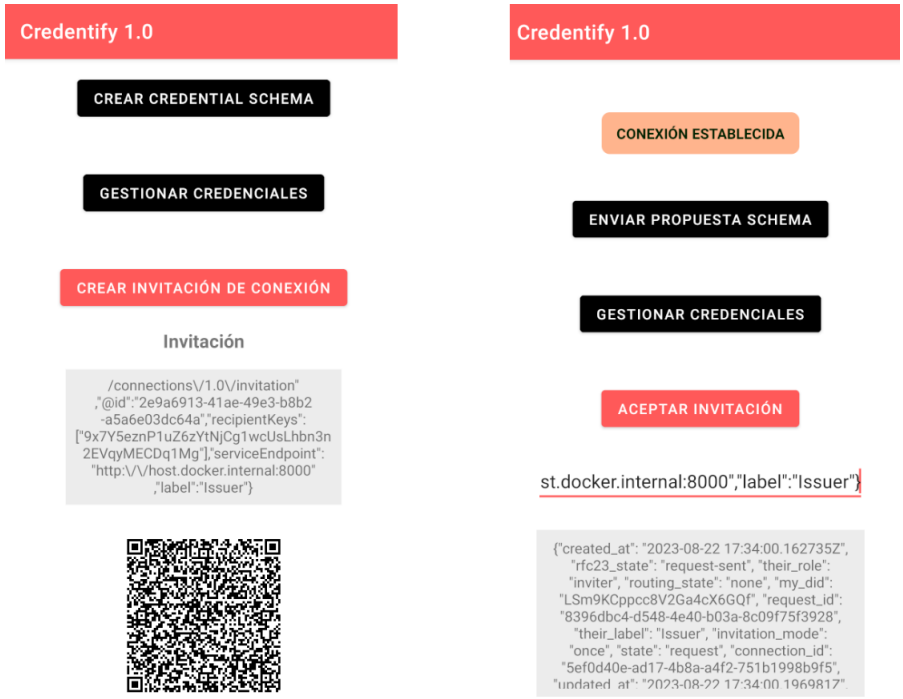

Figura 6.6: Conexión establecida entre ambos agentes mediante la aplicación

La Figura 4.7 representa el diagrama de flujo que subyace al establecimiento de la conexión entre un agente *Issuer* y un agente *Holder*.

A continuación, se verifica a través de Postman la creación exitosa de la invitación de conexión, observando que comienza en un estado de *invitation*, como se indica en el campo *state* de la Figura 6.7

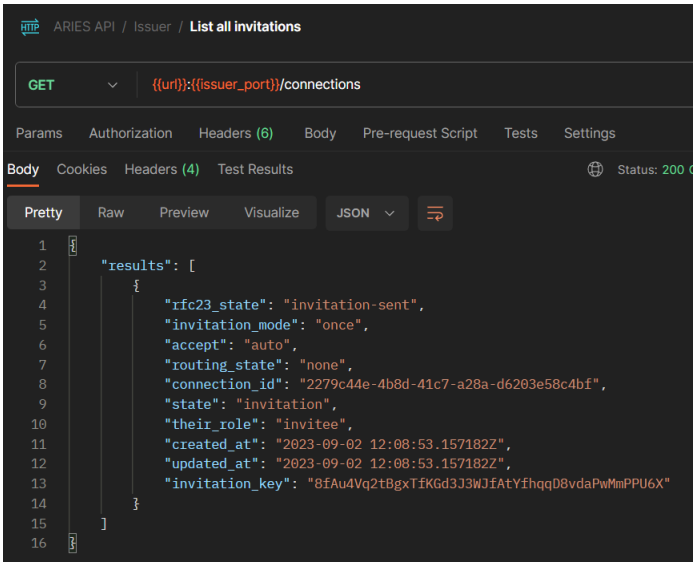

Figura 6.7: Creación de invitación de conexión en Postman - Issuer

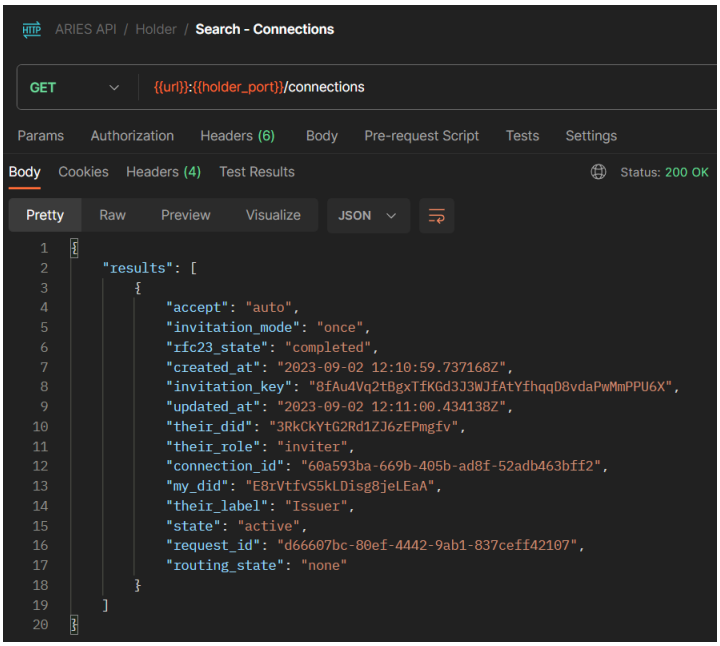

Figura 6.8: Aceptación de invitación de conexión en Postman - Holder

Una vez que el agente *Holder* acepta la invitación de conexión, el campo *state* cambia a *completed* en la parte del agente *Holder*, como se muestra en

la Figura 6.8. Esto confirma una conexión exitosa desde el punto de vista del agente *Holder*. Al mismo tiempo, el agente *Issuer* actualiza el estado de la invitación a .ªctive,"lo que indica el éxito en la conexión entre ambos agentes. Este cambio se puede apreciar fácilmente en la Figura 6.9.

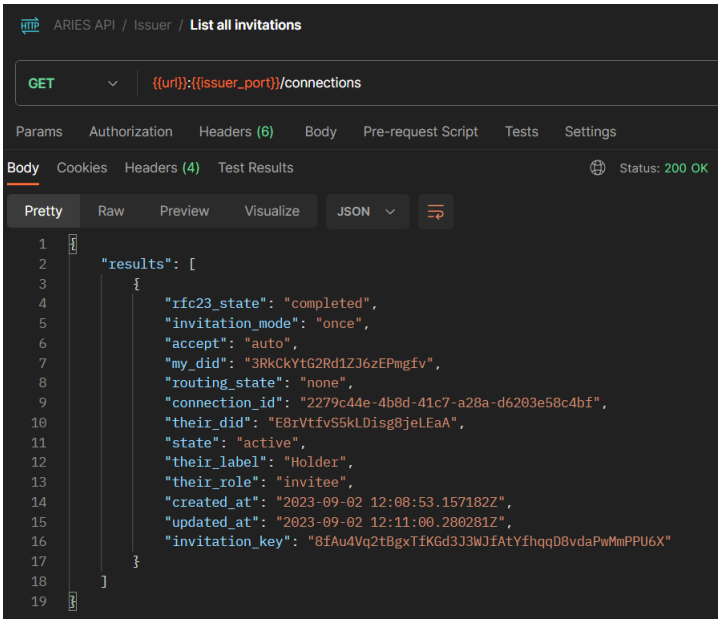

Figura 6.9: Confirmación de invitación de conexión en Postman - Issuer

A continuación, una breve descripción de los campos más relevantes del JSON de respuesta:

- **my_did**: El identificador de la entidad que realiza la acción.
- **their_label**: Etiqueta asociada a la entidad con la que se realiza la conexión.
- **connection_id**: Identificador de la conexión establecida.
- **their_did**: Identificador de la entidad con la que se estableció la conexión.
- **invitation_mode**: Modo de la invitación.
- **rfc23_state**: Estado según el RFC 23.
- **their_role**: Rol de la entidad con la que se realiza la conexión.
- **state**: Estado de la conexión.
- **invitation_key**: Clave de la invitación de conexión.

6.2.3. Escenario 3: Administración de credenciales en ambos agentes

El tercer escenario se enfoca en la administración de credenciales una vez que se ha establecido la conexión entre dos agentes.

Descripción del Escenario:

Una vez que la conexión entre el agente *Issuer* y el agente *Holder* se ha establecido, ambos pueden comenzar a intercambiar mensajes relacionados con la emisión y recepción de credenciales.

El proceso se detalla en la Subsección 4.3.4 junto al diagrama de flujo en la Figura 4.10, aunque a continuación se comentan ligeramente los pasos adjuntando evidencias de la interfaz de usuario de la aplicación desarrollada.

Cabe destacar que en las imágenes que se adjuntan, encontramos a la **izquierda** la vista de nuestro agente en modo *Issuer*, y a la **derecha** la vista de nuestro agente en modo *Holder*.

En la Figura 6.10, el *Holder* envía una propuesta de credencial al *Issuer*, indicando los atributos y su valor, además del nombre del esquema de credencial que se utiliza.

Figura 6.10: Elaboración de propuesta de credencial en modo Holder

Cuando el *Issuer* recibe la propuesta de credencial (Figura 6.11), verifica si puede emitir una credencial que cumpla con los requisitos del *Holder*.

Si es posible, el *Issuer* responde con una oferta de credencial, confir-

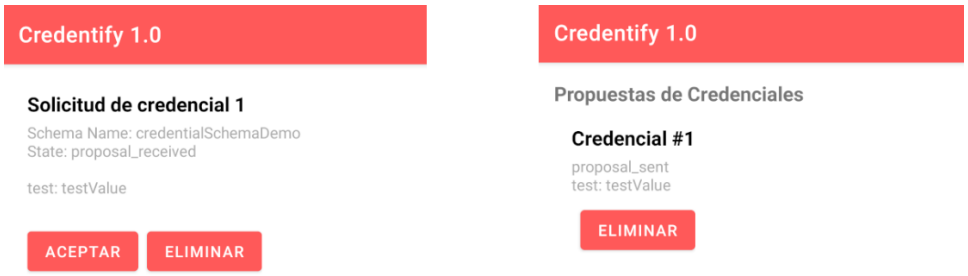

Figura 6.11: Propuesta de credencial enviada por el Holder a Issuer

mando los atributos que incluirá en la credencial como se muestra en la Figura 6.12.

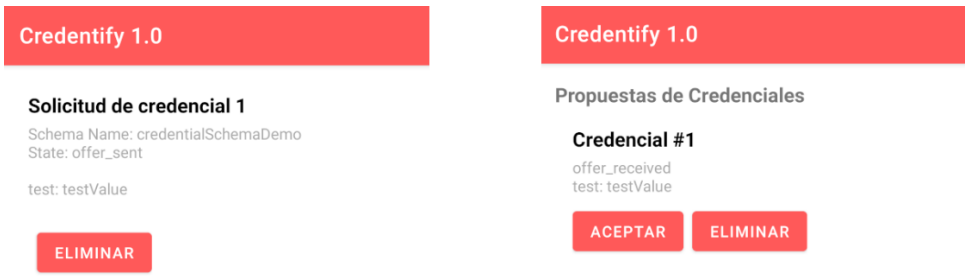

Figura 6.12: Oferta de propuesta de credencial enviada por el Issuer realizada

A continuación, el *Holder*, al recibir la oferta de credencial y estar satisfecho con la información proporcionada por el *Issuer*, confirma su deseo de recibir la credencial enviando un mensaje de solicitud de credencial al *Issuer*. La Figura 6.13 se usa para demostrar esta interacción.

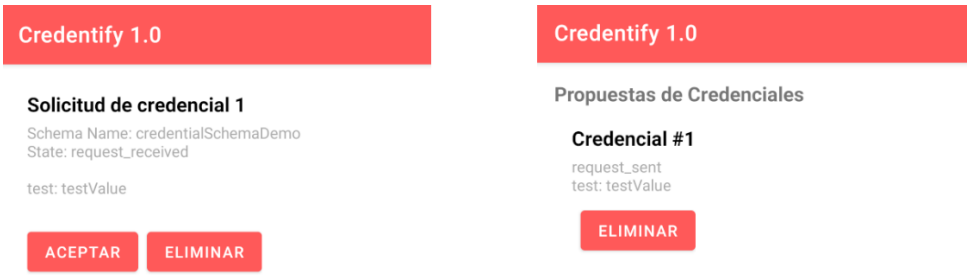

Figura 6.13: Petición de Holder para recibir credenciales válidas realizada

Una vez que el *Issuer* recibe la confirmación del *Holder*, procede a emitir la credencial según lo acordado en la oferta de credencial. Finalmente, el *Holder* almacena la credencial en su cartera digital y envía un mensaje de confirmación de recepción (ACK) al *Issuer* como se muestra en la Figura 6.14.

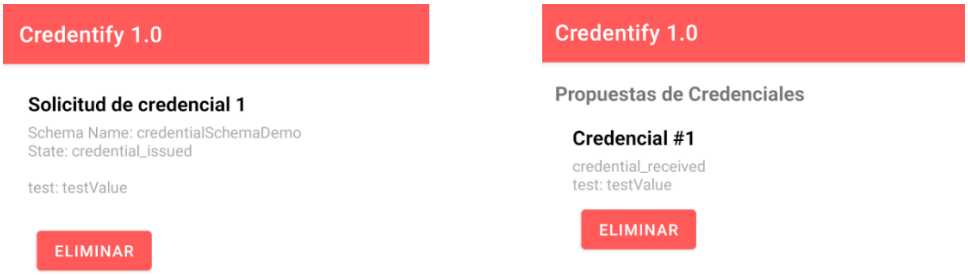

Figura 6.14: Credenciales emitidos por Issuer y almacenados en Holder

Con motivo de profundizar en el proceso de administración de credenciales dentro del framework se considera la propuesta de credencial que se muestra en la Figura 6.15, la cual será utilizada para realizar pruebas con Postman.

Una vez que nuestra entidad *Issuer* ha aceptado la solicitud de la credencial, recibiremos un JSON de respuesta que contiene información detallada sobre la credencial emitida. Puedes observar una vista parcial de esta respuesta en la Figura 6.16, y la estructura parcial de datos se encuentra en la Figura 6.17.

De manera análoga, podemos verificar cómo se almacena nuestra credencial en el agente *Holder* después de una serie de interacciones entre agentes. La visualización parcial de esta credencial almacenada se muestra en la Figura 6.18, y la estructura parcial de datos se encuentra en la Figura 6.19.

Figura 6.15: Propuesta de credencial emitida en Holder

Por último, es importante destacar que en el proceso de establecimiento de conexión, expedición de credenciales y propuestas de credenciales, no se realizan registros en la *blockchain*, al menos a través del protocolo específico que se utiliza en este contexto.

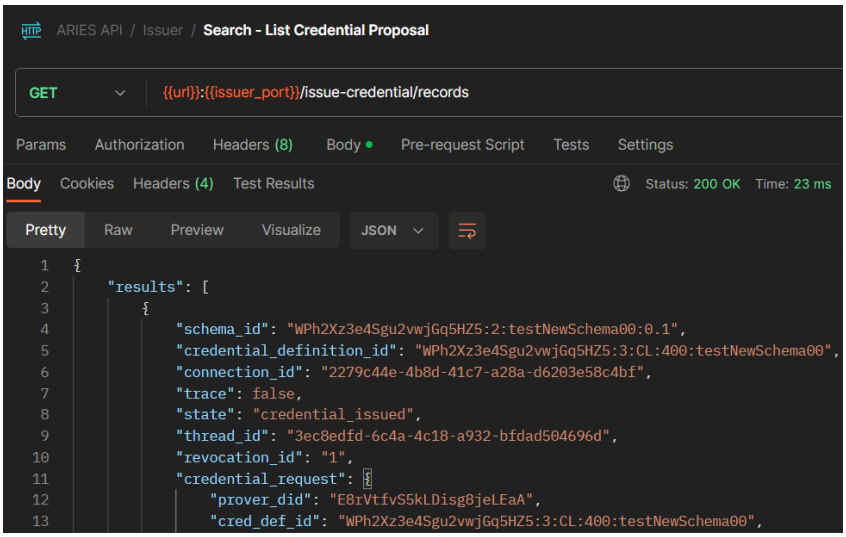

Figura 6.16: Credencial emitido por Issuer en Postman

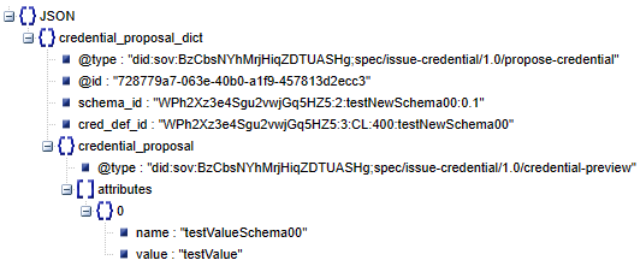

Figura 6.17: Respuesta en formato JSON de credencial expedido por Issuer

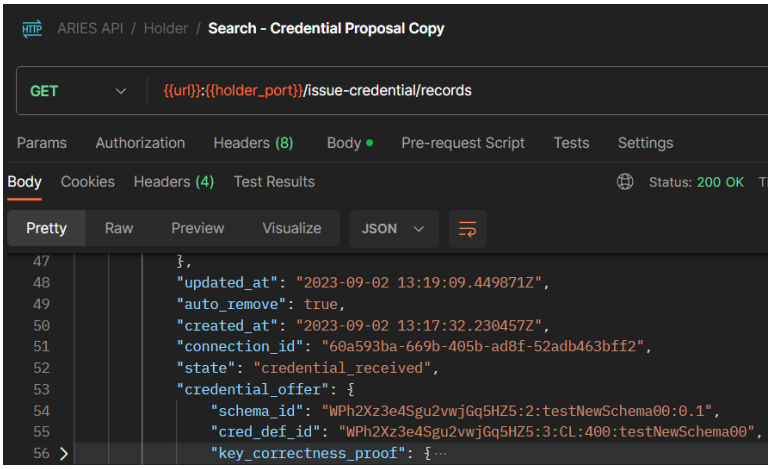

Figura 6.18: Credencial almacenada por Holder en Postman

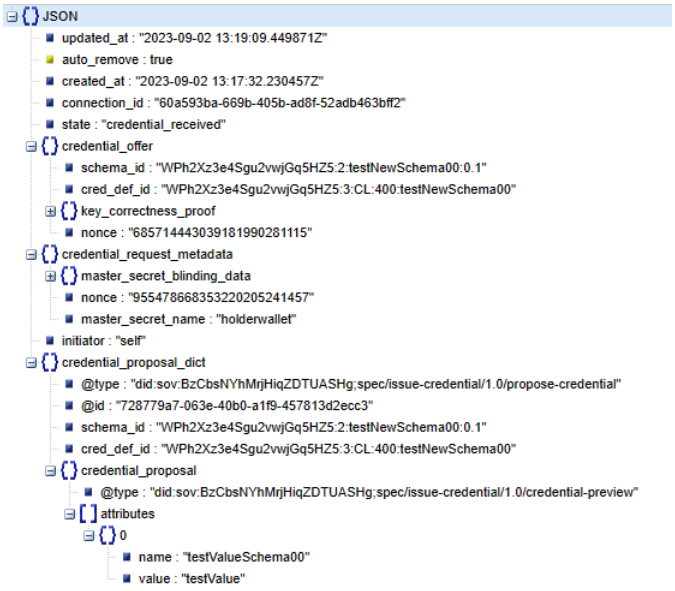

Figura 6.19: Respuesta en formato JSON de credencial almacenado por Holder

Capítulo 7

Conclusiones y Líneas de Trabajo Futuro

7.1. Conclusiones

El proyecto ha culminado con éxito en el desarrollo de una aplicación para dispositivos móviles basados en Android.

Se han alcanzado satisfactoriamente los objetivos propuestos, incluyendo el desarrollo del software necesario para gestionar los frameworks de los agentes. A continuación, se resumen los principales logros y resultados alcanzados a lo largo de este proyecto:

1. **Investigación en Identidad Digital Soberana (SSI):** Se ha llevado a cabo una exhaustiva investigación sobre la Identidad Digital Soberana, destacando el estudio minucioso de los proyectos Hyperledger de la Linux Foundation, específicamente Indy, Ursa y Aries. Estos proyectos permiten la creación de un entorno SSI, lo cual constituye un componente esencial para la gestión de identidades digitales seguras.

2. **Desarrollo de Controladores Aries:** Se ha realizado un desarrollo completo de controladores diseñados para interactuar con los agentes Aries. Estos controladores son fundamentales para el manejo adecuado de los agentes y la realización de diversas pruebas en un entorno de Identidad Digital Soberana.

3. **Implementación Práctica:** Se ha logrado implementar un entorno completo que incluye el despliegue de una red Indy local junto con los frameworks y controladores de los agentes Aries. Esto ha permitido la realización exitosa de múltiples pruebas.

4. **Eficiencia en el Uso de Recursos:** Los resultados obtenidos indican que los recursos necesarios para el despliegue de los agentes Aries son relativamente bajos. La implementación se ha llevado a cabo de manera eficiente, a pesar de algunas dificultades encontradas durante el estudio del proyecto Hyperledger.

En resumen, se representa un logro significativo en la implementación de tecnologías de Identidad Digital Soberana y demuestra su versatilidad y accesibilidad en una variedad de dispositivos.

7.2. Líneas de Trabajo Futuro

A medida que avanzamos hacia un mundo digital cada vez más interconectado y dependiente de la seguridad en línea, es crucial considerar las áreas clave en las que este proyecto puede continuar brindando un impacto significativo. A continuación, se comentan brevemente las áreas de enfoque futuro que tienen el potencial de fortalecer aún más la utilidad y relevancia de esta solución.

Áreas de Enfoque Futuro:

1. **Manual de Uso:** Desarrollo de un manual completo que detalle la implementación, configuración y solución de problemas para futuras implementaciones.

2. **Casos de Uso:** Ampliar la variedad de casos de uso para demostrar la aplicabilidad de la tecnología en diversas industrias y escenarios.

3. **Pruebas Funcionales y Eficiencia:** Realizar pruebas exhaustivas para medir la eficiencia y el rendimiento de la solución.

4. **Extensión a iOS:** Adaptar la herramienta para funcionar en dispositivos iOS, ampliando su alcance y accesibilidad.

Estas áreas representan oportunidades cruciales para mejorar y expandir la tecnología de Identidad Digital Soberana, contribuyendo al desarrollo de soluciones de gestión de identidades digitales seguras en un mundo digital en constante cambio.

Bibliografía

[1] Cisco España. (2023). *Cisco Global Networking Trends 2023: El Futuro de la Red en un Mundo Multi-Cloud.* https://news-blogs.cisco.com/emea/es/2023/05/23/cisco-global-networking-trends-2023-el-futuro-de-la-red-en-un-mundo-multi-cloud/

[2] The Linux Foundation. *Linux Foundation.* https://www.linuxfoundation.org/.

[3] Reglamento (UE) n ° 910/2014 del Parlamento Europeo y del Consejo, de 23 de julio de 2014, relativo a la identificación electrónica y los servicios de confianza para las transacciones electrónicas en el mercado interior y por la que se deroga la Directiva 1999/93/CE. http://data.europa.eu/eli/reg/2014/910/oj/spa.

[4] Propuesta de la Comisión Europea de 28 de abril de 2021 (COM (2021) 281 final) - Propuesta de Reglamento del Parlamento Europeo y del Consejo por el que se establece el marco para una Identidad Digital Europea, conocida como eIDAS 2. https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=CELEX%3A52021PC0281.

[5] Real Academia Española. Definición de Identidad. 2023. Disponible en: https://dle.rae.es/identidad.

[6] *Descentralización.* En Wikipedia, la enciclopedia libre. https://es.wikipedia.org/w/index.php?title=Descentralizaci%C3%B3n&oldid=147669985. Page Version ID: 147669985.

[7] *Becoming a Hyperledger Aries Developer.* edX. https://www.edx.org/course/becoming-a-hyperledger-aries-developer.

[8] Hayashi, M., & Hong, J. (2011). A diary study of password usage in daily life. CHI'11.

[9] Kavrestad, J., Zenzerović, Z., & Nohlberg, M. (2020). Analyzing the usage of character groups and keyboard patterns in password creation. IMCS.

[10] Ma, C., Wang, H., & Zhang, D. (2014). Security flaws in two improved remote user authentication schemes using smart cards. International Journal of Communication Systems, 27(10), 2305-2314.

[11] Developer. "How to Overcome Two-Factor Authentication Vulnerabilities?"Trustifi. 15 de junio de 2021. Disponible en: https://trustifi.com/how-to-overcome-two-factor-authentication-vulnerabilities/.

[12] *¿Qué es Hyperledger y por qué debería importarte?.* https://fastercapital.com/es/contenido/Que-es-Hyperledger-y-por-que-deberia-importarle.html.

[13] Diccionario de la lengua española, *Definición RAE de ≪credencial≫ según el Diccionario de la lengua española: 1. adj. Que acredita. 2. f. Documento que acredita a una persona para desempeñar una determinada función. 3. f. pl. cartas credenciales.*, URL: https://dle.rae.es/credencial.

[14] Descripción de la Cadena de Bloques, *URL: https://www.amd.com/es/technologies/blockchain-explained*.

[15] *¿Qué es la tecnología Blockchain? - IBM Blockchain* | IBM. https://www.ibm.com/es-es/topics/what-is-blockchain.

[16] Muñoz Moruno, Laia, *Identidad digital soberana*, Tesis de Licenciatura, Universitat Politècnica de Catalunya, Fecha de aceptación: 16 de septiembre de 2020, URL: https://upcommons.upc.edu/handle/2117/328800.

[17] Curran, S. and Howard, C., *Introduction to Hyperledger Sovereign Identity Blockchain Solutions: Indy, Aries & Ursa*, Linux Foundation. edX.

[18] W3C. (s.f.). *Verifiable Credentials Data Model v1.1.* https://www.w3.org/TR/vc-data-model/

[19] Ruiz Molina, M., *Identidad Descentralizada Aplicada*, Universitat Oberta de Catalunya (UOC). URL: https://openaccess.uoc.edu/handle/10609/148138

[20] Fiege, U., Fiat, A., and Shamir, A. (1987). *Zero knowledge proofs of identity.* In *Proceedings of the nineteenth annual ACM symposium on Theory of computing* (pp. 210-217). New York, NY, USA: Association for Computing Machinery. DOI: 10.1145/28395.28419

[21] Hyperledger Foundation, *Hyperledger Aries*, Hyperledger Foundation, 2023. https://www.hyperledger.org/projects/aries

[22] Hyperledger Foundation, *Hyperledger Indy*, Hyperledger Foundation, 2023. https://www.hyperledger.org/projects/hyperledger-indy

[23] Hyperledger Foundation, *Hyperledger Ursa*, Hyperledger Foundation, 2023. https://github.com/hyperledger-archives/ursa

[24] Merlin George and A. Chacko, *Health Passport: A blockchain-based PHR-integrated self-sovereign identity system*, Frontiers in Blockchain, 2023.

[25] T. Pflanzner, Hamza Baniata, and A. Kertész, *Latency Analysis of Blockchain-Based SSI Applications*, Future Internet, 2022.

[26] Taehoon Kim, Imyeong Lee, *A Study on DID and CP-ABE-Based Self-sovereign Identity of Smart Vehicles*, Computer Science & Information Technology, 2023.

[27] Gabriella Laatikainen, Ravikant Agrawal, Xiaofeng Wang, P. Abrahamsson, *The state of self-sovereign identity in spring 2021: Results of a survey*, JYU Reports, 2022.

[28] N V Kulabukhova, *Zero-Knowledge Proof in self-sovereign identity*, 2019.

[29] The Linux Foundation. 2023. Introduction to Hyperledger Blockchain Technologies. https://www.edx.org/es/learn/hyperledger

[30] The Linux Foundation. 2023. Becoming a Hyperledger Aries Developer. https://www.edx.org/learn/computer-programming/the-linux-foundation-becoming-a-hyperledger-aries-developer

[31] Hyperledger. (30 de julio de 2023). *Hyperledger Aries Framework for JavaScript*. GitHub. https://github.com/hyperledger/aries-framework-javascript.

[32] Hyperledger. "Aries Agent Test Harness."GitHub. https://github.com/hyperledger/aries-agent-test-harness.

[33] Hyperledger, *Hyperledger/aries-cloudagent-python: Hyperledger aries cloud agent python (ACA-PY)*, 2023, https://github.com/hyperledger/aries-cloudagent-python.

[34] Hyperledger. "aries-acapy-controllers."GitHub. https://github.com/hyperledger/aries-acapy-controllers.

[35] Hyperledger. "aries-mobile-agent-xamarin."GitHub. https://github.com/hyperledger/aries-mobile-agent-xamarin.

[36] Hyperledger. "aries-mediator-service." GitHub. https://github.com/hyperledger/aries-mediator-service.

[37] Real Academia Española, *Metodología*, Real Academia Española, 2023. [Online]. Disponible en: https://dle.rae.es/metodolog%C3%ADa.

[38] Fielding, Roy Thomas. *Architectural Styles and the Design of Network-based Software Architectures*. Ph.D. Dissertation, University of California, Irvine, 2000.

[39] Friesen, Jeff. *Learn Java for Android Development*. Apress, 2014. https://doi.org/10.1007/978-1-4302-6455-2

[40] Siyi Liu. *Explore Java Language and Android Mobile Software Development*. 2021. https://doi.org/10.25236/IJFET.2021.030202

[41] Anónimo. *A Comparative Study of Java and Kotlin for Android Mobile Application Development*. 2020, https://ieeexplore.ieee.org/document/9315483

[42] Hyperledger Aries. *Aries RFC 0036: Issue Credential 1.0*. 2021. [Online; accessed 30-July-2023]. https://github.com/hyperledger/aries-rfcs/tree/bb42a6c35e0d5543718fb36dd099551ab192f7b0/features/0036-issue-credential

[43] Hyperledger Aries. *Aries RFC 0036: Issue Credential Protocol 1.0*. https://github.com/hyperledger/aries-rfcs/tree/bb42a6c35e0d5543718fb36dd099551ab192f7b0/features/0036-issue-credential.

[44] BCGov GitHub. *VON Network*. https://github.com/bcgov/von-network.

[45] Docker Documentation. *What is Docker?*. https://docs.docker.com/get-started/overview/.

[46] Postman Learning Center. *Introduction to Postman*. https://learning.postman.com/docs/getting-started/introduction/.

[47] Android Developers. *Meet Android Studio*. https://developer.android.com/studio/intro.