Una aplicación DLT de actividad ciudadana con la plataforma TAPLE

Grado en Ingeniería Informática

Trabajo Fin de Grado

Autor:
Francisco Miras García
Tutor/es:
Jesús Joaquín García Molina
José Ramón Hoyos Barceló

Facultad
Informática
Universidad
Murcia

3 de Julio de 2023

Una aplicación DLT de actividad ciudadana con la plataforma TAPLE

Autor
Francisco Miras García

Tutor/es
Jesús Joaquín García Molina
Informática y Sistemas
José Ramón Hoyos Barceló
Informática y Sistemas

Facultad de Informática

Grado en Ingeniería Informática

DIS
Departamento de Informática y Sistemas

UNIVERSIDAD DE MURCIA

Murcia, 3 de Julio de 2023

Preámbulo

La blockchain proporciona una forma descentralizada, segura y confiable de almacenar y gestionar datos, sin necesidad de intermediarios y con garantías de integridad de la información. En este documento se aborda el uso de las bases de datos datos en la blockchain desde la perspectiva de la gestión de los datos, analizando las características y ventajas de la tecnología blockchain en bases de datos, así como plataformas y herramientas para su implantación.

El objetivo de este documento es profundizar en el uso de tecnología de bloques aplicadas a base de datos, su potencial utilización en la industria y el día a día.

Agradecimientos

Deseo expresar mi más sincero agradecimiento a mis dos tutores del Trabajo Final de Grado (TFG), Jesús Joaquín García Molina y José Ramón Hoyos Barceló, por su invaluable guía, apoyo constante y dedicación en las múltiples revisiones de la documentación. Su experiencia y conocimientos han sido fundamentales para el éxito de este trabajo.

Quiero agradecer especialmente a Antonio Estévez y al equipo de Open Canarias por brindarnos la oportunidad de conocer y utilizar la plataforma Tracking (Autonomous) of Provenance and Lifecycle Events (TAPLE). Su disponibilidad para resolver nuestras dudas y problemas ha sido de gran ayuda y ha enriquecido nuestro trabajo de manera significativa.

Agradecer a mi pareja, Elena, por estar a mi lado durante todo el proceso de este trabajo. Su constante apoyo, aliento y comprensión han sido pilares fundamentales que me han brindado la fuerza y la motivación necesarias para seguir adelante.

A mi familia, quiero agradecerles su constante apoyo y confianza a lo largo de toda mi carrera universitaria, en especial a mi Madre. Su amor incondicional y su fe en mis habilidades han sido pilares fundamentales en mi desarrollo académico y personal. Estoy enormemente agradecido por tener una familia tan maravillosa y cercana a mi lado.

Quiero expresar mi más sincero agradecimiento a mis amigos, en particular a Edu, por brindarme confianza inquebrantable y por ser una fuente constante de aliento a lo largo de este camino. Su amistad ha sido un pilar fundamental en mi vida y en el desarrollo de este proyecto. También quiero agradecer a Miguel por su apoyo incondicional y por estar presente en los momentos en los que más lo necesitaba.

Por último, quiero agradecer a José Antonio Ces, mi jefe en el trabajo, por ofrecerme flexibilidad horaria y permitirme dedicar tiempo laboral en las últimas semanas para pulir y finalizar este trabajo. Su comprensión y apoyo han sido de gran importancia para alcanzar mis metas académicas.

Declaración firmada sobre originalidad del trabajo

D./Dña. **Francisco Miras García**, con DNI **23810304Z**, estudiante de la titulación de **Grado en Ingeniería Informática** de la Universidad de Murcia y autor del TF titulado "**Una aplicación DLT de actividad ciudadana con la plataforma TAPLE** ".

De acuerdo con el Reglamento por el que se regulan los Trabajos Fin de Grado y de Fin de Máster en la Universidad de Murcia (aprobado C. de Gob. 30-04-2015, modificado 19-07-2018 y 23-07-2020), así como la normativa interna para la oferta, asignación, elaboración y defensa delos Trabajos Fin de Grado y Fin de Máster de las titulaciones impartidas en la Facultad de Informática de la Universidad de Murcia (aprobada en Junta de Facultad 27-11-2015)

DECLARO:

Que el Trabajo Fin de Grado presentado para su evaluación es original y de elaboración personal. Todas las fuentes utilizadas han sido debidamente citadas. Así mismo, declara que no incumple ningún contrato de confidencialidad, ni viola ningún derecho de propiedad intelectual e industrial

Murcia, a 3 de Julio de 2023

Fdo.: Francisco Miras García
Autor del TF

Resumen

En la actualidad, muchos sectores de la industria está siendo revolucionados por la integración de la tecnología blockchain. La blockchain proporciona una forma descentralizada, segura y transparente de almacenar datos. Teniendo como principales características la inmutabilidad de los datos y la resistencia a la manipulación, dos atractivos que ayudan a generar confianza y le permiten destacar en contextos donde la seguridad y fiabilidad son necesarias.

En este documento se abordan una introducción sobre la tecnología y conceptos básicos. Análisis de características, ventajas y desventajas de la tecnología, así como los diferentes tipo de bases de datos que pueden beneficiarse de la blockchain, como bases de datos transaccionales, distribuidas, orientadas a documentos. Seguido por un breve estudio de las principales herramientas y plataformas para su implantación, entre ellas Hyperledger Fabric, BigchainDB y TAPLE. Para continuar, una ampliación sobre la plataforma taple y sus elementos básicos. Finalizando con el diseño de un caso de estudio para TAPLE y el desarrollo de una prueba de concepto para la misma.

Extended Abstract

Bitcoin was born in 2009 and with it the concept of blockchain. For some years now, blockchain has become a trend, considered one of the main trends in the world of software and is causing a digital transformation with scope in all areas of human endeavour, proof of which is the interest shown by large technology companies, consultancies and government organisations, as is the case of IBM, Open Canarias and the European Union. It has also become popular on a social level thanks to cryptocurrencies, while at the same time the blockchain has raised suspicions caused by bitcoin mining and environmental concerns.

The blockchain is a type of distributed ledger, as such it offers the advantages of being a system that is decentralised, immutable and tamper resistant. Thanks to the use of an append only policy, obtaining a consensus of data in the network and while it provides the guarantee that once a transaction is added it cannot be revoked once it has been stored. The ledger concept comes from the notion of accounting books, which record all the transactions instead of updating the final value. Apart from the blockchain, there are other types of distributed ledgers such as the Directed Acyclic Graphs. These features allow blockchain technology to provide benefits such as integrity of the information, increased trust in the data, better security of the information and improved auditing and monitoring of transactions. These features and benefits come with the following problems, the first problem is known as "Byzantine Fault", the problem of consensus, at the same time preventing the possible manipulation or introduction of transactions by malicious actors, to solve this problem the use of consensus algorithms is made. Within the consensus algorithms there are several different varieties, the simplest of which consists of a vote among the network's participants, as well as designating certain participants to verify and aggregate the transactions, and others that are more computationally expensive, such as the proof of work. The choice of an algorithm is determined in part by the openness of the blockchain network in that the public networks need an algorithm that ensures that consensus cannot be forced by a small part of the network, while a private network that enjoys authorisation and authentication of participants can make use of a less computationally complex consensus algorithm. The second problem faced is the participation, especially in public networks, it is required that there is a sufficient number of participants to ensure the security and integrity of the information and to prevent manipulation of the network by malicious actors, to solve this problem it is necessary to provide the right incentives, varying from the information available on the network to financial rewards for participation. The third problem is the technical cost of adopting the blockchain technology,

in addition to the operational and infrastructure cost, especially if a centralised system was previously used. Finally, in a paradoxical way, the advantage of the immutability of the data is at the same time a limitation, due to the fact that it prevents compliance with data protection requirements and data protection laws such as in the case of the General Data Protection Regulation of the European Union, which has led to some blockchain platforms to try include the possibility of deleting information, with the aim of complying with these legal requirements. The blockchain provides the aforementioned benefits, thanks to the way it works. This consists of chaining groupings of one or more transactions, these groupings are called blocks, by means of a cryptographic hash, which is obtained from the content of the current block and the cryptographic hash of the previous block, creating a chain, which is why it is called blockchain. This chain guarantees that each block has a cryptographic signature containing references to its content and the cryptographic hash of the previous block, making it practically impossible to alter an existing block or introduce a new one between two blocks. Despite the disadvantages, thanks to its benefits, there are many use cases in which the use of blockchain technology fits perfectly, such as the creation of virtual currency, the substitution of traditional banking systems, the management of digital identity, the tracking of asset life cycles, the traceability and monitoring of processes, and it is even a viable solution for existing problems such as the management of clinical records between healthcare areas. There are examples of some use cases, such as Bitcoin or Ethereum cryptocurrencies, or the management of the lifecycle of assets such as the case of NFTs. Thanks to the rise of the Internet of Things in the last few years, the need for distributed and secure information systems, such as the blockchain platforms, means that their use is increasingly being taken into consideration. Furthermore, the blockchain technology gives life to the concept of the smart contracts, being the conceptualisation of small software programs stored in the blockchain, with the ability to control, document or execute operations in the blockchain according to the specifications of the contract, providing the possibility of automating transactions, tasks, managing assets or verifying compliance with conditions or regulations.

The blockchain platforms need some type of mechanism to store their information. Therefore, different platforms have built new storage systems that combine the features characteristic of database management systems and the features characteristic of the blockchain. Alternatively, it is also possible to build a blockchain platform on top of an existing database management system, by extending its functionality to include the features characteristic of the blockchain technology. It is also possible to use existing databases to store a copy of the blockchain data in order to perform data analysis or to perform operations outside the blockchain and then reflect the results back to the blockchain.

There are currently many different blockchain platforms that allow the development of blockchain applications. Some of these platforms are Hyperledger Sawtooth, initially designed to be used for building public networks but adaptable to private networks, it has a modular character with a high level of abstraction between the application

layer and the core layer allowing support for multiple programming languages, support for several different types of consensus algorithms, it has the possibility of being compatible with the Ethereum network through a plugin, however its main attraction is the ability to execute transactions in parallel. Belonging to the same organisation, Hyperledger Fabric has been widely extended and popularised, despite only allowing the setting up of private networks, it has a fully modular architecture that supports the use of many different consensus algorithms, several different identity managers and a number of different cryptographic tool suites, it incorporates support for the use of smart contracts, but its main features characteristic is the implementation of an architecture for executing transactions called execute-order-validate. On the other hand, BigchainDB is a blockchain platform that builds on the use of MongoDB, achieving a blockchain platform capable of performing a high volume of transactions, providing a low response latency and supporting a more complete query language, for both private and public networks. KERI is a platform that focuses exclusively on identity management, employing the concept of self-signed keys and eliminating the need for a trusted authority to issue keys or verify them. Last but not least, TAPLE is a new platform, which is still under development, that we have been able to experiment with thanks to the collaboration of the ModelUM research team, part of the Software Engineering development group of the University of Murcia, with the software company Open Canarias.

TAPLE is a platform that is being developed by Open Canarias, with strong inspiration from KERI. Its main objective is to serve as a solution for lifecycle tracking, traceability of processes and assets. At the same time it allows the integration of the Internet of Things devices directly into the network. These objectives are achieved by the use of micro ledgers, an isolated ledger for a specific process or asset instead of a ledger for everything, thus achieving better scalability and reduced hardware requirements. Support for the use of a variety of cryptographic schemes and the use of a number of different cryptographic mechanisms. As well as the focus on the reduction of power consumption per transaction.

TAPLE is currently being developed on the Rust programming language, while keeping collaborations with development groups. Their main approach is the use of a single ownership model in which there is only one owner and the owner is in charge of ordering the changes and adding them to the micro ledger. To ensure data integrity and trust, it relies its confidence on a model of verifiers, the verifiers being to check the validity of the events and to check that the changes made can be performed. All this by means of a governance that is nothing more than a system of rules to define the rules in the network, with the possibility that in the same network there may be one or more governance systems and that one participant may be in several of them. For this purpose, TAPLE defines the following elements. First, a subject is the entity, process or asset on which we want to trace its changes over time, each subject constitutes a micro ledger and the data structure that follows it is defined by a json schema. The specification of the schemas is stored into the governance and the schemas define the

possible types of subjects that can be created and the structure they follow. The governance, which is itself a subject, also establishes the relationship between participants and subjects, defining the participants of the governance, the available schemas and the permissions and roles that each participant has over each of the subjects. Inside of a governance, the same participant can have one or more roles. First of these being the owner of a micro ledger. The approver can vote for or against possible changes over a subject. The validator is in charge of verifying the changes and ensuring that only one version of each event will exist. Finally, the witness can view a subject's information and the Invoker can request changes to a subject. The changes are propagated through the network by means of events, being generated by the owner or the invokers, the available events are genesis, it creates a subject. State, which modifies a subject. Fact, related events of the subject, that do not alter the information. Transfer, change of the ownership of a subject. Finally End of life, closing a life cycle and preventing future modifications over the subject.

The TAPLE platform is still under development and still needs some more refinement, in later versions it will incorporate the usage of smart contracts, several software development kits in multiple languages including one for android, the removal of a micro ledger to comply with data protection laws. As well as the inclusion of the witness role, therefore the witness role could not be tested and has influenced the design of the case study.

Índice general

Índice de figuras

Índice de tablas

Índice de Códigos

1. Introducción

Desde hace algunos años, la tecnología *blockchain* es considerada una de las principales tendencias en el mundo del software y una de las innovaciones detrás de la transformación digital que ha alcanzado todas las áreas del quehacer humano, demostrando un potencial para revolucionar ciertas áreas de la industria. A pesar de que se ha hecho popular por su aplicación a las criptomonedas y el intercambio de bienes, su utilidad se extiende a muchos dominios en los que la trazabilidad es crucial como la gestión de activos, logística, gestión de salud o identidad digital, entre otros muchos. La tecnología blockchain es muy útil en la gestión de datos dado que aporta alta descentralización y seguridad, y facilita la trazabilidad de todas las operaciones realizadas sobre los datos. Cabe señalar que esta tecnología es un caso particular de Distributed Ledger Technology (DLT) y que este trabajo se enfoca sobre DLT más que blockchain.

1.1. Objetivos

El equipo de investigación ModelUM, que es parte del grupo de investigación de Ingeniería del Software de la Universidad de Murcia, y la empresa Open Canarias han colaborado en varios proyectos de modernización de software dirigida por modelos y mantienen una relación desde hace casi dos décadas. De este modo, a lo largo del proyecto se ha mantenido una relación con Antonio Estévez, responsable de desarrollo d software de Open Canarias[1], y se ha estado al tanto del lanzamiento de *Tracking (Autonomous) of Provenance and Lifecycle Events (TAPLE)*, plataforma DLT destinada a soportar la trazabilidad del ciclo de vida de objetos de diferente naturaleza de una forma escalable, eficiente y sostenible.

El objetivo del trabajo fin de grado (TFG) presentado en esta Memoria ha consistido en realizar una exploración de la tecnología DLT, su relación con la blockchain, sus posibles casos de uso, conocer algunas de las principales herramientas blockchain actualmente disponibles. Además se ha decidido probar en este TFG la plataforma a través de un pequeño caso de estudio.

1.2. Metodología

El trabajo se ha organizado en las siguientes 5 etapas:

[1]Empresa galardonada con el premio Ángela Ruiz Robles en los Premios de Investigación Sociedad Científica Informática de España. https://www.opencanarias.com/news/premio-fundacion-bbva/

- Adquisición de conocimientos necesarios para la realización del documento: Ledgers Distribuidos, Blockchain y plataformas blockchain.
- Diseño del caso de estudio: concepto de partida, casos de uso, arquitectura de la red.
- Construcción de la prueba de concepto.
- Escritura de la memoria de la práctica.

Durante el desarrollo del TFG, se llevaron a cabo una serie de actividades en cada etapa, las cuales se describen en la Tabla 1.1. Estas actividades incluyeron reuniones con los tutores y el equipo de desarrollo de TAPLE, en las cuales se realizó un seguimiento del trabajo y se establecieron los hitos futuros.

El proyecto comenzó con una reunión inicial para definir el alcance del TFG y planificar el trabajo. A lo largo de las etapas, se recopiló documentación y se estableció contacto con Antonio Estévez, CEO de Open Canarias, para afinar el objetivo del trabajo. Se definió un caso de estudio viable para realizar una prueba de concepto representativa. Las últimas etapas del trabajo se realizaron simultáneamente, escribiendo la memoria y dando forma a la prueba de concepto.

Durante el proceso, se utilizaron herramientas como GitHub para control de versiones, WhatsApp para la comunicación ágil con el equipo de TAPLE y los tutores, y ChatGPT y GitHub Copilot para agilizar la generación de estructuras de datos, documentación del código y búsqueda de conceptos.

1.3. Organización del documento

El documento continua con las siguiente estructura: Explicación de los conceptos básicos de una DLT y la tecnología blockchain. Seguido de un análisis de distintas plataformas y herramientas de blockchain. Profundización en la tecnología TAPLE. Descripción de un pequeño caso de estudio, relacionado con conceptos de identidad soberana, ciudades inteligentes y el diseño de una prueba de concepto haciendo uso de . Finalizando con conclusiones y vías futuras obtenidas durante la realización del trabajo.

Etapa	Tipo de actividad	Descripción	Participantes	Duración
Adquisición de conocimientos	Reunión	Presentación del trabajo	Francisco Jesús Juaquín José Ramón	2 h
	Investigación	Investigación sobre blochain	Francisco	1 Diciembre - 24 Junio
	Reunión	Cierre de Objetivos/ Presentación TAPLE	Francisco Jesús Juaquín José Ramón Antonio Estévez	1 h
	Webminar	Webminar sobre TAPLE	Francisco	1.25 h
	Reunión	Presentación de conceptos para el trabajo	Francisco Jesús Juaquín José Ramón	1 h
Diseño del caso de estudio	Reunión	Discursión sobre TAPLE y posibles casos de uso	Francisco Jesús Juaquín José Ramón Antonio Estévez José Hidalgo	2.5 h
	Diseño/Investigación	Diseño de posibles escenarios	Francisco Jesús Juaquín José Ramón Antonio Estévez José Hidalgo	16 Marzo - 3 Mayo
	Reunión	Cierre caso de estudio	Francisco Jesús Juaquín José Ramón Antonio Estévez José Hidalgo	1 h
	Diseño/Investigación	Desarrollo del caso de estudio	Francisco	5 Mayo - 27 Junio
Construcción de la prueba de concepto	Desarrollo	Desarrollo de la prueba de concepto	Francisco	5 Mayo - 2 Junio
Escritura de la memoria	Redacción	Redacción de la documentación del TFG	Francisco	30 Abril - 2 Junio

Tabla 1.1: Metodologa del proyecto

2. Fundamentos

En este capítulo se exploran los conceptos de DLT y la implementación más extendida, la blockchain.

2.1. Conceptos básico de la tecnología DLT

El termino inglés ledger (en español "libro de contabilidad") hace referencia a un libro de cuentas en el que se almacenan las transacciones que modifican los balances. Su característica principal consiste en una política de escritura *append-only*, por la que nunca se eliminan transacciones registradas, salvo excepciones muy limitadas. Esta política fuerza a que cualquier operación de modificación se realice añadiendo transacciones, nunca eliminando. De este modo, siempre se dispone de un histórico de todos los cambios de los valores. La idea detrás de la tecnología DLT es la construcción de un ledger distribuido y compartido, en el cuál existe un sistema de consenso que garantiza que todas las réplicas tengan la misma información y esta tenga una coherencia en el tiempo. El uso de una DLT aporta resistencia a los fallos, transparencia, inmutabilidad y mayor seguridad [2, 3].

Dentro de la tecnología DLT, se pueden clasificar las distintas implementaciones según la estructura de datos utilizada. La más ampliamente empleada es mediante *blockchain*, de la cual se profundizará más adelante. No obstante, también existen otras implementaciones que hacen uso de Grafos Acíclicos Dirigidos (DAG) y estructuras híbridas de datos [3].

Las principales características de la tecnología DLT son las siguientes[2, 4, 5]:

- *Descentralización*: Distribución de los datos a lo largo de una red de nodos, en vez de estar almacenados en un único nodo centralizado. Cada nodo tiene una réplica de los datos y esta replicación de los datos otorga mayor seguridad, reduciendo el riesgo de pérdida o corrupción de los datos. Además, al distribuir los datos, se dificulta la posibilidad de que una alteración no deseada o unilateral de los datos no sea detectada y rechazada.

- *Política Append-only*: Se refiere a almacenar todos los cambios que sufren los datos con las transacciones, de manera que se recuerda el pasado de un dato, permitiendo un histórico del mismo y seguir su trazabilidad. Esta política, en definitiva, establece que cualquier operación supone añadir datos nunca eliminar datos existentes.

- *Inmutabilidad*: No es posible alterar una transacción una vez almacenada. En caso de que se haya introducido una transacción errónea, la única forma de subsanarla es realizar una transacción adicional corrigiendo el fallo.

- *Consenso*: Se considera válida una transacción cuando todos los participantes aceptan su validez. Esto se logra mediante un algoritmo de consenso.

- *Finalidad*: Es la confianza de que un bloque añadido a la red no puede ser revocado, consigue que la información sea confiable y garantiza que se pueda trazar el origen, la pertenencia de un activo o la completitud de la transacción.

Estas características inherentes proporcionan una serie de beneficios que se exponen a continuación[2, 4, 5]:

- *Integridad de los datos*: Se garantiza la inmutabilidad de los datos y resistencia a la manipulación. Esto se consigue gracias a la descentralización, pruebas criptográficas y política append-only.

- *Confianza*: Aumenta la confianza que se tiene en los datos gracias a la integridad de los datos y la descentralización. Esto conlleva de que los participantes tengan acceso al histórico de datos, mayor resistencia a fallos y convergencia en única verdad.

- *Mejor auditoría*: El uso de un ledger compartido supone que se puede usar como fuente veraz de los hechos ocurridos mejorando la capacidad de monitorización y auditoría. Además dado que todos los participantes tienen una copia del ledger facilita las auditorías.

- *Mayor seguridad*: Los mecanismos que controlan la manipulación de los datos dificultan que se produzca fraude o delitos.

El principal desafío de una DLT, que también afecta a los sistemas distribuidos en general, es el problema del consenso, también conocido como "Fallo Bizantino" [6]. En este caso es resuelto mediante algoritmos de consenso, a continuación algunos ejemplos de los más genéricos:

- Practical Byzantine Fault Tolerance (PBFT) (Tolerancia Práctica a Fallas Bizantinas): Una de las primeras propuestas para la solución del Fallo Bizantino, propuesta en 1999 por Miguel Castro y Barbara Liskov[7]. Consiste en realizar realizar múltiples rondas de votación entre los participantes hasta llegar a un consenso.

- Proof of Authority (PoA) (Prueba de autoridad): Las transacciones en los bloques son validadas por participantes definidos como verificadores a los que se les permite añadir transacciones. El fundamento es usar como incentivo el hecho de ser

verificador y asociarlo la reputación del participante con el participante. La principal desventaja es el requisito de mantener evitar se corrompa los participantes con autoridad [8].

2.2. Características de la blockchain

La tecnología blockchain surge en 2009 como infraestructura de la Bitcoin[1]. La blockchain es una implementación de una DLT que se caracteriza por agrupar las transacciones en bloques, y estos son enlazados entre ellos formando una cadena en la que cada bloque está conectado al anterior, para lo cual se utilizan hashes criptográficos. De esta forma se consigue crear un registro inmutable y descentralizado[4, 5].

En una solución blockchain, un *participante* es una persona, empresa, organización o entidad que participa directamente en la red del sistema distribuido manteniendo un nodo. Por el contrario, una entidad que hace uso de la red a través de terceros no se considera un "participante". En el caso de Blockchain, cada participante actúa como un nodo dentro de red, en la que todos tienen una copia de la cadena completa de bloques (*ledger*), formando la red blockchain, como se puede observar en la Figura2.1.

Las redes blockchain se pueden clasificar en públicas (sin permisos) o privadas (permisionada). En una red pública, cualquiera puede participar[2], los datos almacenados son totalmente públicos, suelen tener un mayor número de participantes pero suelen usar algoritmos de consenso más costosos. Por otro lado las redes privadas, la participación esta reducida a un numero de participantes autorizados e identificables, dentro de la red los datos pueden tener distintos niveles de visibilidad y disponen de una selección de algoritmos de consenso más amplia[2, 4, 5].

La tecnología blockchain aporta los beneficios del uso de una DLT, aunque debido a su funcionamiento sufre de los siguientes problemas [2, 4, 5]:

- *Coste operativo y técnico*: El coste operativo y técnico requerido por parte de una infraestructura de blockchain es elevado. En el caso de una red privada, la entidad deberá ocuparse de mantener un número considerable de nodos siempre funcionando, y de gestionar la política de permisos y la seguridad.

- *Coste de recursos y sostenibilidad*: Dependiendo del mecanismo de consenso utilizado, el coste de recursos puede ser muy elevado tanto en el despliegue como de mantenimiento. Además de el coste energético o de hardware asociado a cada transacción.

- *Inmutabilidad de los datos*: La inmutabilidad de los datos, incluye también el impedir el borrado de datos dentro de la red, por consiguiente aunque un dato

[1]Primera criptomoneda creada por una persona sin identificar bajo el seudónimo de *Satoshi Nakamoto*: https://bitcoin.org/es/

[2]Que cualquiera pueda participar no reduce la seguridad de los datos

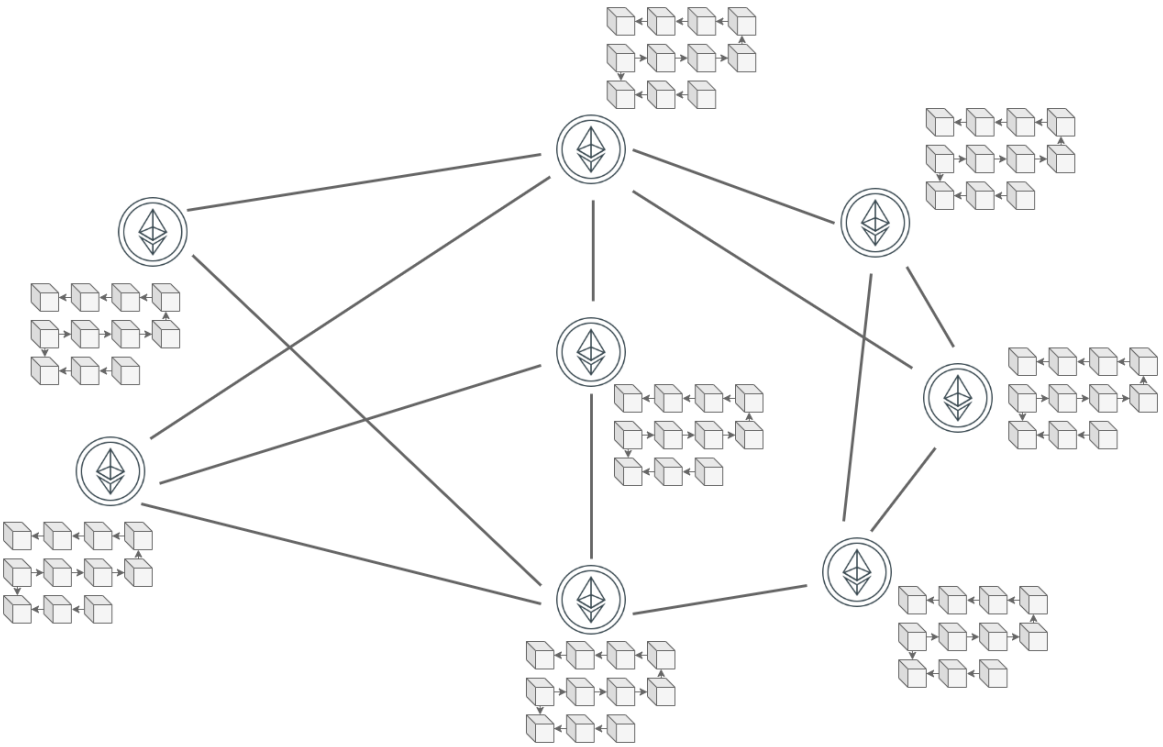

Figura 2.1: Diagrama de una red blockchain[1]

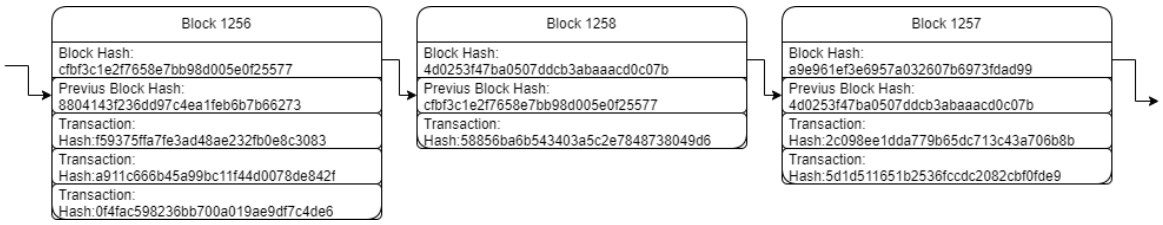

Figura 2.2: Bloques de una red blockchain

se puede invalidar o marcar como borrado, este no desaparece del histórico. En aplicaciones concretas puede entrar en conflicto con leyes o políticas. Como es por ejemplo el derecho al olvido[3] de la Unión Europea.

- *Participación*: Para que una red blockchain sea segura e integra necesita tener un elevado número de participantes. Una red publica necesita encontrar personas o entidades que deseen mantener nodos de la red. Dependiendo de si la red es pública o privada, se necesita de un número mayor o menor. Esto conlleva a la necesidad de incentivar la participación en las redes públicas. Un claro ejemplo de esto son los mineros de las red Bitcoin.

2.2.1. Funcionamiento de la blockchain

Como se ha indicado anteriormente, el método utilizado para almacenar la información en una red blockchain consiste en agrupar los bloques en una cadena o secuencia que se confirma en el tiempo. Cada bloque que se va a añadir contiene un hash criptográfico a partir de su material criptográfico, su contenido y del hash criptográfico del bloque anterior, creando una cadena. Figura2.2 Gracias a este hash criptográfico se consigue que un bloque se puede alterar o introducir un bloque entre dos existentes.[4, 5].

En una blockchain no existe una autoridad central siendo necesario que los nodos lleguen a un consenso sobre el estado de la red y las transacciones válidas. Sin un consenso, la blockchain podría ser vulnerable a ataques o manipulaciones maliciosas.

Existe una variedad de algoritmos de consenso usados en blockchain y algunos de los más conocidos son:

- Proof of Work (PoW) (Prueba de Trabajo): Forma de prueba criptográfica cuyo funcionamiento es el siguiente: Para añadir un bloque los participantes deben resolver un problema computacionalmente costoso, el primer participante en resolverlo (probador) informa de la solución y el resto debe verificar que la solución es correcta y que se ha invertido un trabajo, la verificación tiene un coste computacional inferior. Es el algoritmo de consenso usado en Bitcoin, a los participantes que compiten para probar la validez de un bloque, se les llama *mineros* dado que "minan bloques" a cambio de una comisión[9]. El principal problema

[3]https://eur-lex.europa.eu/ES/legal-content/summary/right-to-be-forgotten-on-the-internet.html

de este algoritmo es la necesidad de incentivar a los participantes debido al coste computacional y energético asociado a generar la PoW. Cabe mencionar el elevado consumo eléctrico [10].

- Proof of Stake (PoS) (Prueba de participación): El funcionamiento consiste en seleccionar un grupo de verificadores en proporción de sus activos en red. Su coste computacional y energético es muchísimo más reducido que el de PoW. El principal problema es que se queda expuesto a más tipos de ataques, como secuestro de una parte de la red o el soborno de verificadores[4, 11]

- Multi-signature (Firma-múltiple): Una parte mayoritaria de los verificadores deben aceptar que el bloque es válido[4]. Es uno de los algoritmos de consenso más eficientes, el principal problema recae en definir una mayoría significativa necesaria, suficientemente alta, pero no como para bloquear una transacción en el tiempo a falta de un verificador. Su uso tiene más sentido dentro de una blockchain privada en la cuál los verificadores puedes ser definidos y autentificados.

- Proof of personhood (Prueba de la persona): La idea es llegar a un consenso mediante una votación en la cual se otorga un voto a cada participantee y cada voto tiene el mismo valor, basándose en que cada participante humano único vote, independientemente de su cantidad de activos de la red. De ese modo se incentiva con recompensas el votar. Las principales desventajas que presenta es la necesidad de verificar que el participante es una persona, y que cada persona tenga un solo voto.[12]. En una red privada se mitiga en parte esta desventaja.

Cada algoritmo de consenso tiene sus beneficios y desventajas. PoW es uno de los más conocidos pero supone un coste de computación muy elevado para ser práctico en aplicaciones de trazabilidad, redes más reducidas o simplemente la inversión en hardware necesaria. Seguido la otra más conocida es PoS, usada por ejemplo en la red de Ethereum[4], una alternativa ideada para redes públicas. El resto de algoritmos de consenso son alternativas para llegar a un consenso con un coste computacional menor, el anteriormente mencionado PoA o Firma-múltiple, pensados en para redes privadas o Proof of personhood que puede ser utilizado en redes publicas y privadas.

La blockchain también ha dado lugar al concepto de "contratos inteligentes" (*Smart Contracts*): pequeños programas que se almacenan en la blockchain pudiéndose ejecutar para controlar o documentar eventos y acciones acorde a los términos del acuerdo. Con los smart contracts se consigue la automatización del cumplimiento de acuerdos de forma transparente y segura. Sus capacidades incluyen la gestión de activos en la red, realizar transacciones, y verificar el cumplimiento de condiciones, entre otras[4, 13].

[4]https://ethereum.org/es/

2.3. Casos de uso

Algunos de los posibles casos de uso de las DLTs son los siguientes: [1, 2, 4, 5, 14, 15]

- *Criptomonedas*: Uno de los casos de uso más reconocidos de la blockchain es su aplicación en el almacenamiento de transacciones de dinero y la gestión de monedas virtuales. La tecnología blockchain permite la creación y el seguimiento de criptomonedas, ofreciendo un medio seguro y transparente para realizar transacciones financieras digitales.

- *Servicios financieros*: La implementación de blockchain en los sistemas bancarios tiene el potencial de acelerar los procesos bancarios y facilitar las auditorías. Esto incluye la capacidad de agilizar operaciones como cambios de moneda y transferencias internacionales.

- *Identidad Digital*: Las DLT se pueden usar para identificar dentro de una red a personas, al mismo tiempo que se mitiga la suplantación de identidades.

- *Trazabilidad de activos*: Permite el seguimiento de forma más confiable y segura de ciclos de vida, pertenencias de un activo o cambio de localización.

- *Gestión y trazabilidad de cadenas de suministros*: Gracias a la inmutabilidad y transparencia de las DLTs aporta confianza a la monitorización y auditoría.

- *Servicios sanitarios*: Los servicios sanitarios se puede beneficiar de una DLT para la gestión de datos médicos, esto permitiría una mejora de la seguridad y experiencia de usuario. Aunque los registros médicos digitales suplen este caso, presentan problemas de seguridad, disputas por la propiedad de los datos e integridad de los datos. Por ejemplo una red privada permitiría a cada paciente ser dueño de su historial médico, al mismo tiempo que el personal sanitario tendría acceso más rápido cuando fuera necesario.

Especialmente con el auge del Internet Of Things (IoT), los puntos fuertes de las blockchain solucionan algunos de los retos a los que se enfrentan los IoT, como algunos de los siguientes[16, Chapter 3].

- La descentralización de la blockchain mejora la robustez y mejora el flujo de los entornos todos a uno, evitando que un fallo en un solo punto.

- Una tolerancia a fallos, que viene de la resistencia a la manipulación, mejora la seguridad de la red frente a clientes comprometidos o maliciosos.

- La inmutabilidad de la blockchain proporciona un nivel de transparencia y seguridad al almacenar datos, necesarios en una infraestructura IoT.

2.4. Ejemplos de uso

En la actualidad ya se pueden encontrar ejemplos de uso de la blockchain como los siguientes:

- *Criptomonedas*: Posiblemente el ejemplo más conocido, la idea de la creación de monedas totalmente virtuales y descentralizadas[4, 5]. Bitcoin y Ethereum son las dos monedas más conocidas.

- *Trazabilidad de activos*: El ejemplo más conocido de esta categoría son los *Non-fungible token* (NFT)[17].

- *Trazabilidad de procesos*: Es difícil encontrar documentación correspondiente, pero *Walmart* utiliza una blockchain para trazar el proceso de la lechuga y la espinaca [5, 18]. Por otra parte, TAPLE está siendo utilizado en algunas startups para trazabilidad de procesos, como la trazabilidad del ciclo del agua por *Acciona*[5].

- *Nombres de dominios*: El servicio de Namecoin, permite el soporte de ".bit" como dominio de alto nivel, controlado mediante clave privada, impidiendo la censura de sitios web.[5]

2.5. Bases de Datos y Blockchain

Las blockchain necesitan algún mecanismo para almacenar la cadena. Algunas plataformas blockchain parten de la idea de construir un nuevo sistema de almacenamiento que reúne algunas de las características básicas de los sistemas gestores de bases de datos y aquellas propias de la blockchain. Ejemplos de este tipo son las plataformas *Hyperledger Fabric* y *BigChainDB* que se describen en el siguiente capítulo. Por otro lado, también es posible construir una plataforma blockchain sobre un sistema gestor de bases de datos existente, ya sea relacional o NoSQL, añadiendo la funcionalidad que caracteriza a una blockchain, y algunos proyectos en este sentido son aquellos descritos en [16, 19, 20]. Además de almacenar la cadena de bloques, las aplicaciones blockchain pueden usar una base de datos tradicional para realizar operaciones típicas sobre los datos que son muy difíciles de llevar a cabo sobre la cadena almacenada [2]. Por tanto, se pueden establecer tres niveles en la relación entre bases de datos y blockchain figura 2.3 [2, 19]:

- Almacenamiento de la cadena en una base de datos blockchain pura: Un sistema de almacenamiento específico para registrar los bloques de la cadena y dotar de las características de blockchain.

[5]Información obtenida en una de las reuniones con Antonio Estévez

- Almacenamiento de la cadena en una base de datos relacional o NoSQL extendida: Se extiende un sistema gestor de bases de datos existente con características blockchain.
- Carga de la cadena en una base de datos externa: Una base de datos puede almacenar una copia de la cadena con el objetivo de hacer operaciones de análisis o realizar múltiples operaciones fuera de la cadena y posteriormente reflejar el resultado final de las transacciones en la blockchain.

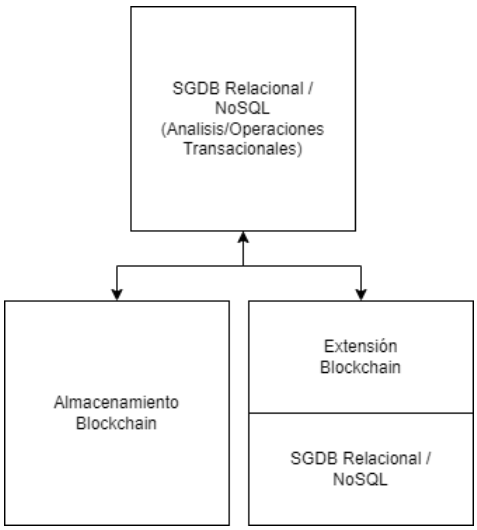

Figura 2.3: Representación niveles de relación

3. Estado del arte

En este capitulo se presentan algunas de las implementaciones de blockchain más conocidas como son *Hyperledger Sawtooth*, *Hyperledger Fabric*, *BigChainDB*, *KERI* y *TAPLE*.

3.1. Hyperledger Sawtooth

Hyperledger Sawtooth es una solución empresarial para la construcción, despliegue y utilización de *ledgers* distribuidos (*blockchains*), de código abierto (licencia Apache 2.0), de carácter pública, aunque cuenta con soporte para redes privadas. Proveyendo una plataforma modular y flexible para implementar transacciones compartidas entre participantes sin confianza mediante un algoritmo de consenso [21]. La versión 1.0 fue lanzada el 29 de julio de 2018 [22]. Perteneciente a la fundación *Hyperledger* [23].

Se diferencia por proporcionar una alta abstracción entre la capa de aplicación y el núcleo, lo que permite el desarrollo de contratos inteligentes en múltiples lenguajes de programación. Además, ofrece la capacidad de ejecutar transacciones en paralelo mediante un planificador paralelo. También cuenta con compatibilidad con la red de Ethereum a través de un plugin de integración. Por último, destaca por su sistema dinámico de consenso, que permite la utilización de diferentes algoritmos de consenso en la misma red [24].

3.2. Hyperledger Fabric

Perteneciente también a la familia de Hyperledger, *Hyperledger Fabric* es considerado como el principal proyecto y el más popular y extendido [25]. Al igual que *Sawtooth*, es un *ledger* distribuido de código abierto (licencia Apache 2.0) de grado empresarial que está basado en una fuerte modularidad y flexibilidad. Pero a diferencia de *Sawtooth*, este es para redes privadas. La versión 1.0 fue lanzada el 11 de Julio de 2017 [26, 27].

Se diferencia por una arquitectura modular que permite el uso de distintos algoritmos de consenso, distintos gestores de identidad y herramientas criptográficas. Además, cuenta con soporte integrado para smart contracts y utiliza una arquitectura para transacciones llamada *execute-order-validate*, que es distinta de la tradicional *order-execute* [27].

3.3. BigChainDB

BigchainDB es una base de datos blockchain de grado empresarial de código abierto (licencia Apache 2.0) que ofrece soporte para redes públicas y privadas [28, 29].

Destaca por su alta modularidad y enfoque en el rendimiento, ofreciendo capacidades de alta velocidad y consulta estructurada gracias a su integración con *MongoDB* [28, 29].

3.4. KERI

Key Event Receipt Infrastructure (KERI) es un sistema descentralizado de administración de identidades en sistemas distribuidos, que se caracteriza por ser de código abierto (licencia Apache 2.0). Su objetivo principal es actuar como un protocolo descentralizado para la gestión de claves en entornos distribuidos.

Una de las características distintivas de KERI es su enfoque en el uso de claves *self-signed*, lo que significa que las claves son generadas y firmadas por el propietario mismo, eliminando así la necesidad de una autoridad de confianza externa. En las redes basadas en KERI, no solo se almacenan las claves, sino también los cambios en el estado de las claves (eventos), lo que garantiza la transparencia y la seguridad de las mismas [15].

3.5. TAPLE

La plataforma *Tracking (Autonomous) of Provenance and Lifecycle Events (TAPLE)* (Trazabilidad y Procedencia de Ledgers) es una solución de tecnología de registros distribuidos (DLT) de código abierto y permisionado, con licencia AGPL-3.0. Fue desarrollada por Open Canarias, tomando una fuerte inspiración de KERI. Su objetivo principal es la trazabilidad de activos y ciclos de vida. Además de la integración de los dispositivos IoT en este proceso [1, 30].

Sus principales características son: Un aumento de la escalabilidad mediante el uso de microledgers, permitir la integración dispositivos con hardware limitado (teléfonos, raspberrys,...) a la red mediante una mayor compatibilidad con el concepto de *fog computing* y los microledgers. Además del soporte para el uso de múltiples esquemas criptográficos y algoritmos de consenso, facilitando. Por último un bajo consumo energético minimizando del coste eléctrico asociado a cada operación [1, 30].

Aunque TAPLE se encuentra actualmente en fases tempranas de desarrollo, se puede trabajar con la herramienta para probar sus características y experimentar con DLT. En el siguiente capitulo se comentara en profundidad esta plataforma.

4. TAPLE

Este capítulo se dedicará a introducir la plataforma *TAPLE*, sus conceptos clave y sus componentes principales.

4.1. Conceptos básicos en *TAPLE*

Tracking (Autonomous) of Provenance and Lifecycle Events (TAPLE) está escrito en Rust[1] y actualmente esta en desarrollo, a la vez que colaboración con equipos de desarrollo e interesados en la adopción de la misma.

Parte de un modelo de propiedad única, basado en verificadores: el incentivo para participar no es ofrecer una recompensa, sino el acceso a los datos. Una red de nodos de *TAPLE* tiene la estructura mostrada en la figura4.1.

A diferencia de las redes tradicionales ilustradas en la figura2.1, cada nodo no guarda una copia completa de la cadena, sino que solamente tiene una copia de los microledgers en los cuales ese nodo es participante. Un microledger es similar a los ledgers utilizados por la blockchain con la diferencia de que en vez de contener agrupaciones de transacciones de múltiples cuentas/activos, cada microledger sólo almacena sobre una cuenta/activos, y cada uno de los bloques sólo guarda una transacción/evento [1, 30].

Cada micro-ledger tiene las siguientes características:[1, 30]

- *Modelo de propiedad única*: Cada uno de los múltiples ledgers en la red tiene un propietario único, siendo el único capaz de realizar modificaciones y siendo responsable del orden de las modificaciones. Esto evita los problemas derivados de la necesidad de llegar a un consenso, siendo que solamente hay una única fuente de verdad, a cambio introduce la posibilidad de que se realicen alteraciones maliciosas.

- *Modelo de confianza mediante verificadores*: El concepto de propiedad única no impide la realización de comportamientos maliciosos. La solución es la introducción de nodos verificadores, estos nodos mantienen una copia del ledger y reciben los eventos modificación de los datos. La responsabilidad de estos nodos es criptográficamente comprobar estos eventos, comprobar si la gobernanza permite que el nodo que emite los cambios pueda realizar cambios y verificar la validez de los

[1] https://www.rust-lang.org/es

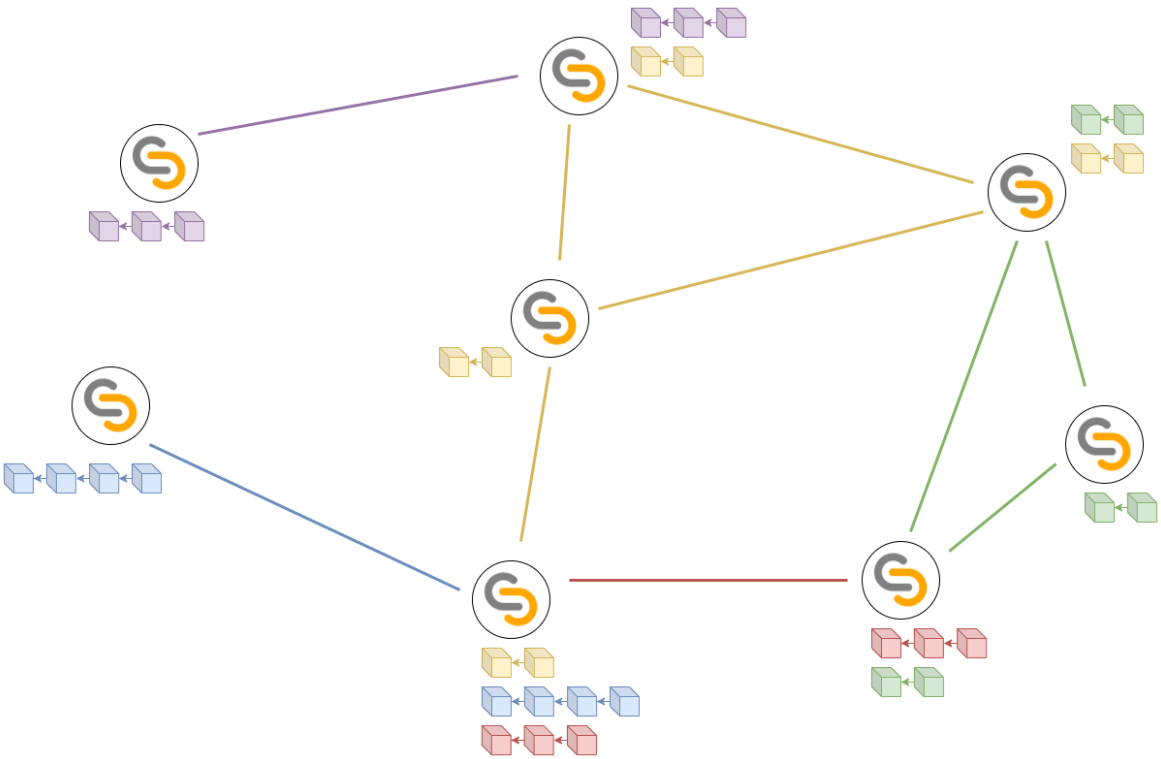

Figura 4.1: Diagrama red de TAPLE[1]

cambios. Depende de cómo se haya establecido la gobernanza, se puede configurar el número de firmas necesarias para generar el nuevo evento. Un cambio que sea rechazado por los verificadores es inválido en la red.

- *Gobernanza por reglas*: Hace uso de un conjunto de reglas para definir los nodos que participan en un ledger, sus roles, así como los requisitos para verificar un nuevo bloque. En una red existen múltiples gobernanzas (una por ledger). Además un nodo puede tener varios roles y participar en diferentes gobernanzas.

Nótese que las características esenciales de blockchain como la integridad de los datos y la resistencia a la manipulación, no se ven alteradas por el uso de microledgers en vez de un ledger.

4.2. Elementos básicos de una red TAPLE

4.2.1. Sujetos

"Una entidad o proceso que genera eventos durante su ciclo de vida, cuyo orden de emisión viene determinado por el propio sujeto"[30]. Los sujetos son entidades lógicas que representan un proceso o activo dentro de la red. Cada sujeto esta representado por un microledger, un esquema de datos, un único propietario y una gobernanza (las gobernanzas se representan en la red como sujetos). Dentro de un sujeto, solamente se encuentran los eventos que le afectan a él. Como veremos, en el caso de estudio, tendremos tres sujetos: Ciudadano, traza de reciclaje y transacción de puntos.

El *estado* de un sujeto es la representación de la información almacenada en un sujeto en un instante dado. Los distintos estados de un sujeto se obtienen mediante la sucesión de eventos a excepción del primer evento. Un sujeto tiene un único propietario, como se ha mencionado antes. Otros participantes pueden solicitar que se realicen cambios. Un sujeto pertenece siempre a un caso de uso, y una gobernanza es la definición de las reglas que gobiernan en el caso de uso, definiendo qué tipos de sujetos se pueden crear y quién puede crear los sujetos. En el caso de un ciudadano, su estado podría incluir la identidad digital y su dirección. Un sujeto sólo puede pertenecer a una gobernanza.

4.2.2. Esquemas

El estado de un sujeto viene determinado por un esquema que define y valida la estructura. Estos esquemas son especificados como parte del proceso de definición de la gobernanza y se distribuyen con la misma, diferentes gobernanzas tienen distintos esquemas. Esta estructura sigue el estándar de *JSON SCHEMA*.

4.2.3. Eventos

Un evento es la unidad de información del los ciclos de vida del los sujetos. El estado de un sujeto viene definido por una secuencia de eventos. La escritura de un evento se realiza como operación atómica.

El sistema de eventos define las siguientes categorías de eventos:

- *Start*: También llamado evento génesis, este evento inicializa a un sujeto, estableciendo los participantes y la gobernanza del mismo.
- State: Almacena un cambio en las propiedades de un sujeto.
- *Fact*: Hechos relacionados con una función o contexto del sujeto. No modifica.
- *Transfer*: Realiza el cambio de propietario del sujeto. Implica una rotación de claves para evitar manipulación por el anterior sujeto.
- *EOL (End of Life)*: Termina el ciclo de vida, impidiendo futuros cambios.

4.2.4. Identidad

Cada participante de la red tiene asociado un identificador único y una clave privada (usada para firmar transacciones).

4.2.5. Roles

Un nodo puede tener distintos roles, estos roles definen las acciones que puede realizar un nodo sobre un sujeto. Los roles existentes son los siguientes: [1]

- *Propietario*: Propietario de un sujeto, tiene control total del sujeto ya que pose el material criptográfico para modificar el sujeto.
- *Aprobador*: Puede votar a favor o en contra de ciertos eventos, con el fin de que se almacene en el ledger.
- *Validador*: Estos nodos aportan sus firmas criptográficas a un sujeto, estos nodos mantienen una copia completa del sujeto y su obligación es garantizar que solo se acepte una versión única de cada evento.
- *Testigo*: Pueden ver la información de un sujeto.
- *Invocador*: Pueden iniciar una solicitud para modificar un sujeto.

4.2.6. Gobernanza

Cada gobernanza establece la relación entre varios participantes y la relación con los sujetos. La gobernanza es responsable de definir los participantes en los casos de usos, los esquemas que se permite su uso, los permisos de cada participantes en la red y los roles de cada participante sobre cada sujeto.

La gobernanza establece una serie de reglas a seguir para cada caso de uso y un nodo puedo participar en más de una gobernanza y define la siguiente jerarquía:

- Cada participante tiene una identidad y un nodo representa a una identidad.
- Una identidad puede participar en varias gobernanzas, y cada gobernanza soporta al menos un caso de uso.
- En cada gobernanza, un participante puede tener distintos roles sobre distintos sujetos. Cada nodo puede almacenar múltiples sujetos, un sujeto siempre pertenece a una gobernanza.

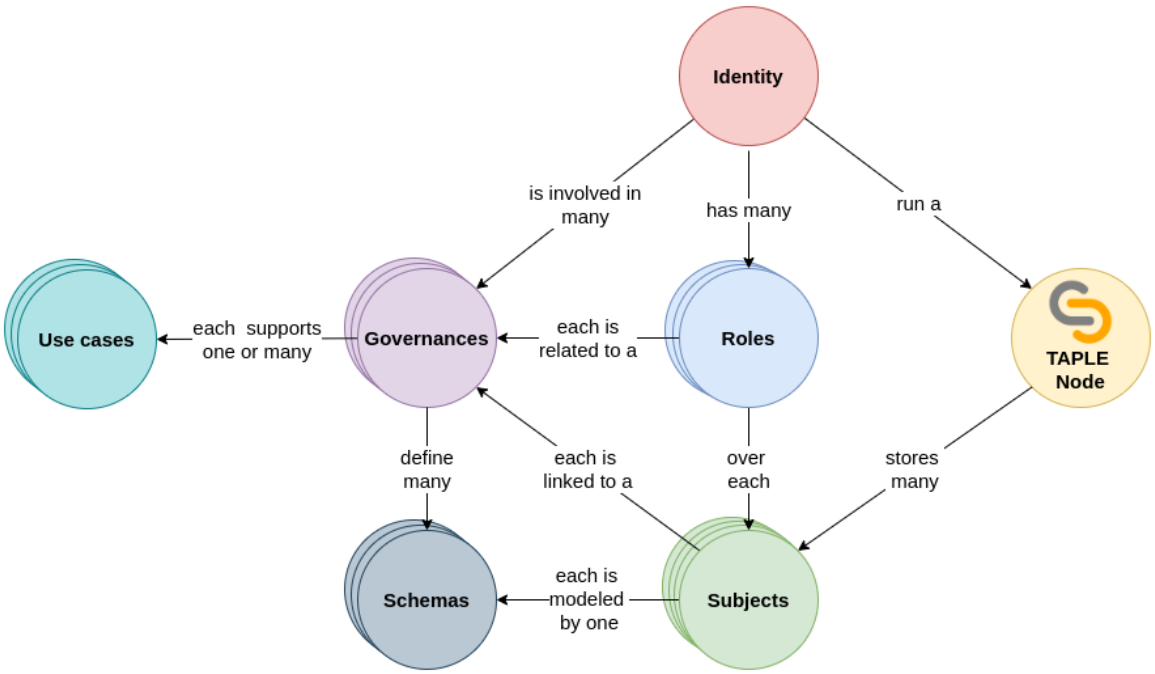

Figura 4.2: Jerarquía de las gobernanzas[1]

Los elementos que conforman una gobernanza son:

- *Miembros*: Participantes de una gobernanza,
- *Esquemas*: Esquemas de las estructuras de datos.

- *Políticas*: Definen los permisos y roles de los participantes.

Dentro de una red, la gobernanza es considerada un sujeto.[1]

4.3. Futuras versiones

Como se ha mencionado anteriormente TAPLE esta actualmente en desarrollo. En la versión actual, la 0.1, existen algunas limitaciones que han impuesto restricciones en el diseño, especialmente al diseñar la gobernanza. Estas limitaciones han surgido debido a un retraso en el lanzamiento de la versión 0.2, en la cual se introducirán roles de testigos y contratos inteligentes para automatizar aprobaciones y verificaciones de eventos.

Por otra parte, está previsto que para la versión 1.0 se incluirá un Kit de Desarrollo de Software (SDK) para Android con el objetivo de facilitar la integración de los teléfonos móviles a la red. También se están desarrollando SDKs en varios lenguajes de programación para simplificar el desarrollo de aplicaciones que no estén escritas en Rust. Estas mejoras permitirán una mayor accesibilidad y flexibilidad para los desarrolladores y usuarios de TAPLE. Por último el borrado de un ledger para garantizar el cumplimiento del *derecho al olvido*.

5. Caso de estudio

En este capitulo se discute sobre el caso de estudio elegido para probar la tecnología blockchain. Se consideraron varias posibilidades y finalmente se eligió el caso de estudio de actividad ciudadana por considerarse más apropiado por las razones que se indican más adelante.

5.1. Posibles casos de estudio

En un inicio se plantearon los siguientes escenarios para definir el caso de estudio:

1. *Identidad de una persona y la trazabilidad del ciclo de vida de una persona*: La idea consiste en la identificación de una persona así como la trazabilidad de eventos en su vida como, historial académico, historial médico, experiencia laboral, etc. Este sería un caso muy completo y está inspirado en algunas iniciativas europeas [1], la lectura del libro "Self-Sovereign Identity: Decentralized Digital Identity and Verifiable Credentials"[31] y en el conocimiento propio del problema de acceso a historiales médico entre áreas sanitarias.

2. *Trazabilidad de un proceso industrial*: Representación de un proceso industrial como, por ejemplo, la fabricación de componentes eléctricos, y utilizar la blockchain para trazar el proceso de fabricación y logística de los componentes y procesos hasta la salida de la fabrica. Otro análogo al anterior sería para la trazabilidad de la logística del reparto de paquetes, trazando todos los recorridos de los paquetes. Conceptualmente más simple pero una de las operaciones más prometedoras.

3. *Conceptos de economía circular y mercado de segunda mano* basado en la trazabilidad de activos, Un sistema de trazabilidad de vehículos, empezando desde la salida de fabrica y trazando los distintos propietarios que ha tenido (concesionario, vendedores, propietarios), visitas al taller, modificaciones del vehículo y accidentes. Creando una traza de la vida del vehículo, pudiendo disipar dudas acerca del estado del vehículo durante la compra de un vehículo de segunda mano y facilitando la transferencia de propiedad.

[1] https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/europe-fit-digital-age/european-digital-identity_es

5.2. Definición caso de estudio

Tras discutirlo con los tutores del trabajo se ha optado por una pequeña aplicación relacionada con el primer caso de uso de identidad digital y actividades de los ciudadanos a lo largo de su vida. En concreto, se ha pensado en abordar una aplicación en que se manejará la identidad de un ciudadano en un ayuntamiento y la creación de un programa de de recompensas por el reciclaje de residuos por parte de los ciudadanos. Este caso de estudio explora la identidad digital de un ciudadano y la trazabilidad del proceso de reciclaje, planteando como incentivo de participación recompensas por reciclar residuos.

Se definen cuatro entidades o actores interesados (stakeholders):

- Ayuntamiento: Interesado en la identidad de los ciudadanos, y el uso del sistema de recompensas por reciclaje.
- Ciudadano: Propietario de su identidad e interesado en el programa de recompensas.
- Empresa de reciclaje: Interesado en el programa de recompensas.
- Zona de reciclaje: Se encarga de iniciar eventos de recogida de residuos y dar puntos recompensa correspondientes.

A partir de estos cuatro tipos actores se crearía una pequeña red que incluiría a dos ciudadanos, un ayuntamiento, una empresa de reciclaje y dos zonas de reciclaje. Esta red posibilitará realizar una prueba de concepto significativa. La red será privada, garantizando la privacidad de los datos personales de los ciudadanos y sus carteras[2].

Como incentivo para fomentar la participación ciudadana se otorgarán puntos de recompensa por el reciclaje. Estos puntos podrán ser consumidos ha cambio de recompensas. Para la empresa de reciclaje, el incentivo radica en los datos de reciclaje que recopila, mientras que para el ayuntamiento, la red brinda la oportunidad de agregar servicios y procesos adicionales en el futuro, así como aprovechar la identidad digital de los ciudadanos para agilizar trámites y gestiones en una futura digitalización de la burocracia.

5.2.1. Casos de Uso

A continuación se definen los casos de uso que se han identificado para los actores del. A partir de los casos de usos se definirán los sujetos necesarios para cumplir con los casos de uso. Los casos de estudio han sido construidos teniendo en cuenta la versión 0.1 de TAPLE.

[2]Definimos cartera como el historial de transacciones de puntos

Ciudadano

La figura 5.1 muestra los casos de uso para el actor *ciudadano*. Se ha considerado la necesidad de consultar su información y poder actualizarla, así como ver y operar con los puntos recompensa.

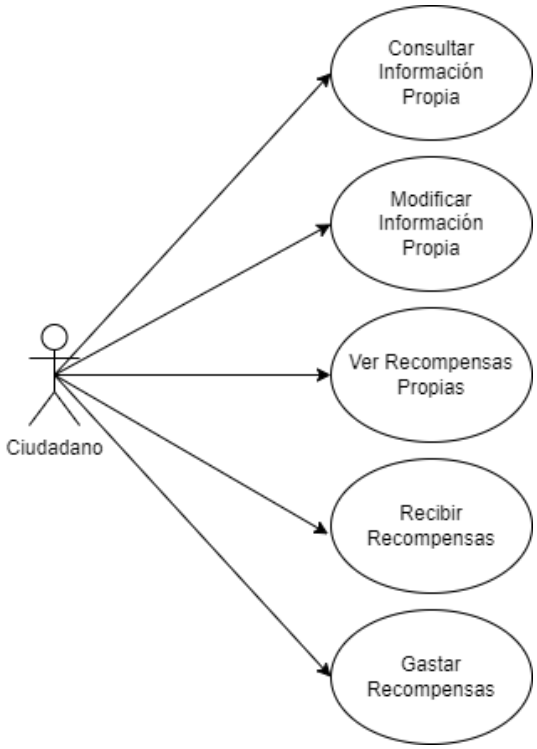

Figura 5.1: Casos de uso ciudadano

Ayuntamiento

La figura 5.2 muestra los casos de uso para el actor *ayuntamiento*. Tiene interés en ver cómo va el reciclaje, consultar información de los ciudadanos, validar los cambios que los ciudadanos realicen sobre sus propios datos, aprobar y validar la redención de recompensas de los ciudadanos.

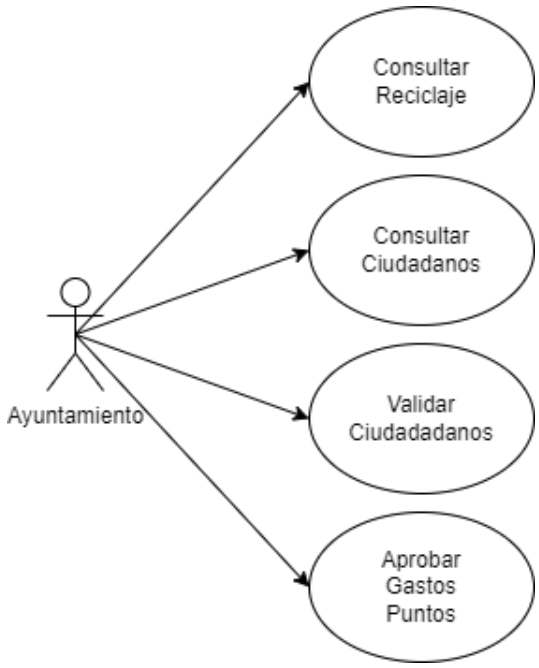

Figura 5.2: Casos de uso ayuntamiento

Zona Reciclaje

La figura 5.3 muestra los casos de uso para el actor *zona de reciclaje*. La zona de reciclaje necesita poder invocar modificaciones de la cartera de puntos de los ciudadanos para añadir puntos y poder trazar conforme se recicla.

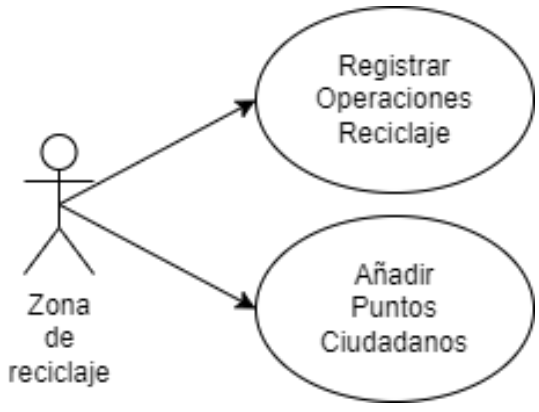

Figura 5.3: Casos de uso zona reciclaje

Empresa de reciclaje

La figura 5.4 muestra los casos de uso para el actor *empresa de reciclaje*. Su principal interés es consultar el estado del reciclaje.

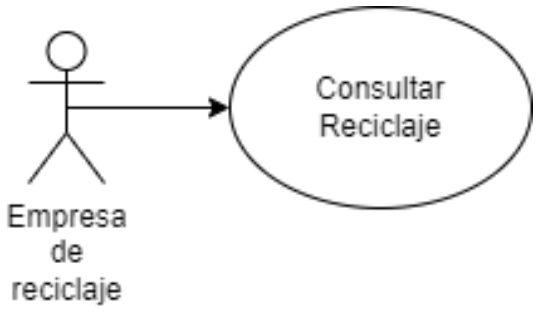

Figura 5.4: Casos de uso empresa de reciclaje

6. Diseño e implementación: Caso de estudio

En este capítulo se describirá el diseño e implementación del caso de estudio presentado en la sección 5.2.

6.1. Diseño

El diseño ha venido limitado por el uso de la versión 0.1 de TAPLE. Se han abordado tres aspectos del caso de estudio: identidad de los ciudadanos, gestión de activos (puntos de reciclaje) y trazabilidad del proceso de reciclaje.

Para la prueba de concepto, se simulara una red con los siguientes nodos: Un nodo representando el ayuntamiento, un nodo representando a la empresa de reciclaje, dos nodos representando zonas de reciclaje y dos nodos representando dos ciudadanos distintos. Se hará uso del lenguaje de *Python (3.11)* para la construcción de la lógica de la aplicación, del cliente de TAPLE[32], `LevelDB` para mapear asociaciones e información auxiliar y `Docker` para la construcción y despliegue de los nodos.

El código desarrollado para la prueba de concepto se puede encontrar en el siguiente repositorio de Github: https://github.com/franciscomirasg/TFG-Blockchaindatabase-Exploration-Release.

6.1.1. Sujetos y Esquemas

Se han definido tres sujetos: Ciudadano, traza de reciclaje y transacción de puntos. Con estos tres sujetos se pueden representar los estados necesarios para cumplir con los casos de uso.

A continuación sería necesario definir los sujetos de la gobernanza. Dado que es necesario representar estos esquemas en formato *JSON SCHEMA*, se ha utilizado la librería *Pydantic*[33] para la codificación de las estructuras de datos, dada su potencia y funcionalidad añadida como la de generar esquemas en el formato requerido de forma automática o la reconstrucción desde diccionarios. Para la generación de los esquemas se ha usado Pydantic y se han realizado ajustes menores a mano. Los esquemas JSON resultantes de cada uno de los sujetos pueden encontrarse en Anexo A.

Adicionalmente con la ayuda de ChatGPT se han generado modelos de Pydantic para los esquemas que usa la API REST de los nodos TAPLE [30].

Ciudadano

Un ciudadano está compuesto por su identidad personal, su dirección y le hemos añadido un identificador propio del ayuntamiento.

Código 6.1: Modelo de Ciudadano

```
class Citizen(BaseModel):
    """
    Data model for the citizen\n
    atributes:
        - identidad: Identidad. Identity of the citizen
        - direccion: Direcion. Address of the citizen
        - cuid: str. Unique identification code of the citizen
    """
    identidad: Identidad = Field(description="Identidad del ciudadano")
    direccion: Direcion = Field(description="Direcion del ciudadano")
    cuid: str = Field(
        description="Codigo de identificaion unico del ciudadano", min_length=36, max_length=36,
        default_factory=lambda: str(uuid4()))
```

El campo `identidad` contiene información referente a la identidad física del ciudadano, el campo `direccion` contiene los información referente a la residencia y el campo `cuid` se utiliza como identificador digital del ciudadano.

Una identidad esta compuesta por:

Código 6.2: Modelo de Identidad

```
class Identidad(BaseModel):
    """
    Data model for the identity of a citizen\n
    atributes:\n
        - nombre: str. Name
        - apellidos: str. Surnames
        - ni: str. DNI
    """
    nombre: str = Field(description="Nombre del ciudadano")
    apellidos: str = Field(description="Primer apellido del ciudadano")
    ni: str = Field(description="DNI del ciudadano",
                    min_length=9, max_length=9)
```

Se han seleccionado los campos de `nombre`, `apellidos` y `ni` (representando en Documento Nacional de Identidad (DNI) y Número de Identidad de Extranjero (NIE)), para representar la identidad física de una persona.

Y una dirección:

Código 6.3: Modelo de Dirección

```
class Direcion(BaseModel):
    """
    Data model for the address of a citizen\n
    atributes:
        - type: ViaType. Type of street
        - direccion: str. Address
        - codigo_postal: str. Postal code
        - comunidad: str. Community
        - ciudad: str. City
    """
    type: ViaType = Field(description="Tipo de vía")
```

```
37    direccion: str = Field(description="Direccion")
38    codigo_postal: str = Field(
39        description="Codigo Postal", max_length=5, min_length=5)
40    comunidad: str = Field(description="Comunidad/Region/Provincia")
41    ciudad: str = Field(description="Ciudad")
```

Para la dirección, se han seleccionado los siguientes campos representativos: `type` es un enumerado para indicar el tipo de vía (alameda,calle,etc...), `direccion` una cadena de texto incluyendo el nombre de la vía e información adicional, `codigo_postal`, `comunidad` comunidad autónoma y `ciudad` ciudad dentro de la comunidad autónoma.

Traza Reciclaje

Una traza de reciclaje es la codificación de la entrada de residuos de un tipo especifico en la zona de reciclaje, esta compuesta por un identificador para la zona de reciclaje, la clase de residuo y el peso de los residuos.

Código 6.4: Modelo de Traza Reciclaje

```
42 class RecycleOperation(BaseModel):
43     """
44     Data model for the recycling operation\n
45     atributes:
46         - type: TrashType. Type of trash
47         - peso: float. Weight of the trash, in grams
48         - container: str. Id of the recycle zone
49     """
50     type: TrashType = Field(description="Tipo de residuo")
51     peso: float = Field(description="Peso de los residuos, en gramos", gt=0)
52     container:str = Field(description="Id del contenedor", min_length=36, max_length=36)
```

Para representar la traza de reciclaje se usan tres campos: `type` un enumerado para indicar el tipo de residuo, `peso` para el peso de los residuos en gramos de la transacción de reciclaje, y `container` que es un identificador de la zona de reciclaje en el que se almacenan los residuos.

Transacción de Puntos

Una transacción de puntos es la codificación de una operación de la cartera de puntos, contiene el valor de la operación, el motivo de la transacción, el balance de la operación con los cambios aplicados y el identificador digital del ciudadano.

Código 6.5: Modelo de Transación de puntos

```
53 class PointTransaction(BaseModel):
54     """
55     Data model for the point operation\n
56     atributes:
57         - value: int. Value of the operation
58         - motivo: str. Reason of the operation
59         - balance: int. Balance of points after the operation
60         - cuid: str. Citizen unique id
61     """
62     cuid: str = Field(
63         description="Codigo de identificaion unico del ciudadano", min_length=36, max_length=36)
```

```
64    value: int = Field(description="Valor de la operación", ne=0)
65    motivo: str = Field(description="Motivo de la operación")
66    balance: int = Field(description="Balance actual de puntos", default=0)
```

Una transacción dentro de la cadena de puntos se representa con los siguientes campos: `value` que es un valor entero positivo o negativo indicando la modificación que se realiza sobre la cartera, `motivo` que es una breve descripción del motivo de la transacción, `balance` que es un valor de la cartera tras aplicar la modificación actual, y `cuid` que sirve para asociar una cartera con el identificador digital del ciudadano.

6.1.2. Generación de material criptográfico

Para la generación del material criptográfico se ha utilizado la herramienta `taple-keygen`, compilada del repositorio del repositorio de la compañia *taple-tools*[1].

Una vez compilada la herramienta para cada nodo se ha realizado el siguiente procedimiento:

Código 6.6: Generación material criptográfico

```
67 taple-keygen > <file_name>
```

Para cada nodo se ha obtenido una salida similar a la del ejemplo:

Código 6.7: Material criptográfico Ciudadano 1

```
68 ["./taple-keygen", "ed25519"]
69 PRIVATE KEY ED25519 (HEX): 9↩
     ↪ ec5ed4fd723d53eb186021e468043d2107cf653ee0663989db6afbfad1755fe
70 CONTROLLER ID ED25519: E6y61Ras_rC1DV_4obUnxGepq5U9S3OMlu9NL5gBkQbA
71 PeerID: 12D3KooWReRHaaPgGpmNu1ktaKQPDT5RkF54fHwMiQQsof1uZzud
```

`Private Key` es la clave que utiliza el nodo para generar las firmas y autentificar operaciones, `CONTROLLER ID` es el id del nodo que tiene esa clave criptográfica y se usa como identificador en la gobernanza. Por último, `PeerID` se utiliza durante la construcción del nodo para referenciar a otros nodos y las operaciones *P2P*.

6.1.3. Gobernanza

En un principio, la gobernanza se planteó para permitir a los ciudadanos, al ayuntamiento y a la empresa ver la traza de reciclaje, pero debido a los problemas del retraso de la versión 0.2, se ha tenido que optar por solamente permitir a la empresa y el ayuntamiento ver la traza de reciclaje. Con la finalidad de no añadir a los ciudadanos como verificadores en la traza de reciclaje, se ha optado por no permitirles ver las trazas del proceso de reciclaje. El esquema completo de la gobernanza se puede encontrar en el Anexo *B*.

[1] https://github.com/opencanarias/taple-tools/tree/release-0.1

Para los elementos de la red de la prueba de concepto se ha planteado el siguiente diseño de la red figura 6.1. Se ha representado con cuadrados de colores, los microledgers de distintos sujetos, con el siguiente código de colores:

- Carteras de puntos: Verde oscuro y azul oscuro.
- Identidad del ciudadano: Verde claro y azul claro.
- Trazas de reciclaje: Amarillo claro y rojo claro.

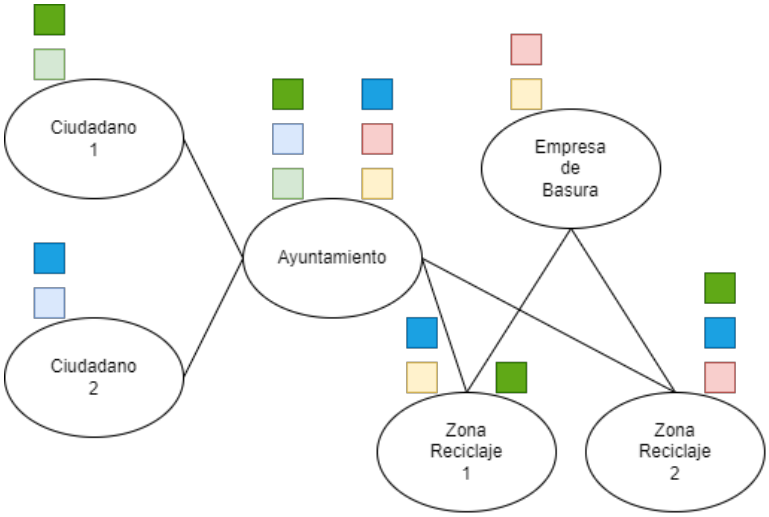

Figura 6.1: Red TAPLE del Caso Estudio

La gobernanza se ha definido siguiendo las instrucciones de la página de desarrollo en [1]. Primero se han añadido los miembros que forman parte de la prueba de concepto, luego se han especificado los tipos de sujetos y el esquema. Por último se han escrito las políticas sobre los sujetos.

Miembros

Se han añadido todos los miembros que formaran parte de la red, rellenando el campo `key` con el correspondiente `CONTROLLER ID` obtenido del material criptográfico generado en la sección *Generación de material criptográfico*.

Esquemas

En esta sección de la gobernanza se han especificado los esquemas que compondrán los sujetos, para el `id` de cada esquema se ha usado el nombre del modelo de datos.

Políticas

Por último, se han redactado las políticas, destacando la importancia de elaborar una política para la gobernanza en sí misma. En el caso de la gobernanza, se ha designado a todos los participantes como verificadores (debido a la falta de testigos disponibles), lo que permite al nodo que la inicializó realizar cambios de manera unilateral. En nuestro caso, la responsabilidad de la gobernanza recae siempre en el ayuntamiento.

Para el ciudadano solamente se añade al ayuntamiento como verificador y aprobador, requiriendo su intervención para verificar los cambios. Por otro lado el ciudadano puede iniciar cambios sobre sus datos.

Las operaciones de reciclaje tienen como verificadores al ayuntamiento y la empresa de reciclaje, para permitir que puedan ver el progreso del reciclaje. Además las zonas de reciclaje pueden generar operaciones de reciclaje de forma unilateral, es decir sin verificar las operaciones de reciclaje.

Para terminar, las transacciones de puntos tienen como verificadores al ayuntamiento y a las zonas de reciclaje. El ayuntamiento tiene la función de verificar las operaciones, mientras que las zonas de reciclaje tienen acceso a la cartera para añadir puntos. Los ciudadanos requieren la verificación del ayuntamiento al canjear puntos, por lo tanto, se asigna un aforo de verificaciones del 10%. Por otro lado, las zonas de reciclaje pueden añadir puntos a la cartera de forma unilateral, sin necesidad de verificación.

Es importante destacar que en la versión 0.1 de TAPLE, los eventos de creación de sujetos no pueden ser aprobados o verificados. Por lo tanto, el ayuntamiento tiene la capacidad de aceptar o rechazar las operaciones de tipo *state (cambio)* realizadas por los ciudadanos, pero no puede impedir la creación de nuevos ciudadanos o carteras de puntos.

6.1.4. Despliegue de la red

Los nodos TAPLE se despliegan en contenedores utilizando *Docker*, utilizando la imagen *opencanarias/taple-client:0.1.4*, que es la última revisión de la versión 0.1. Para facilitar el despliegue, se utilizan siete scripts. Se asigna un script a cada uno de los participantes, y se utiliza un último script para secuenciar el despliegue de los anteriores.

Los scripts de cada nodo siguen una estructura similar, cambiando la clave y los valor de los puertos:

Código 6.8: Despliegue ayuntamiento

```
72 docker run -d -e TAPLE_HTTPPORT=3000 \
73     -e TAPLE_NETWORK_ADDR=/ip4/0.0.0.0/tcp \
74     -e TAPLE_NETWORK_P2PPORT=40000 \
75     -e TAPLE_NODE_SECRETKEY=2↩
         ↪ daf002ddcf793fe73aa987fbf4aeddfca94bc89164d7a054c9bdb0b25407250↩
         ↪  \
```

```
    -e RUST_LOG=info \
    -p 3000:3000 \
    -p 40000:40000 \
    --name="city_hall" \
    opencanarias/taple-client:0.1.4
```

La opción `-d` permite que el contenedor se inicialice en segundo plano, `-name` para indicar el nombre el contenedor y `-e` para las variables de entorno. Las variables de entorno usadas vienen especificadas en el apartado de desarrollo de [1]

Para el resto se ha ido incrementando en uno el valor de los puertos y además incorporan una linea extra en el caso de la zonas de reciclaje. Esta linea extra es para indicar al nodo que existen al menos otro nodo en la red en una dirección IP determinada, necesario para conectar los nodos entre sí. Para las zonas de zonas de reciclaje el nodo conocido es la empresa de reciclaje, en el resto de casos en ayuntamiento. En esta línea se usa el `PeerID`.

Código 6.9: Variable entorno adicional

```
    -e TAPLE_NETWORK_KNOWNNODES=/ip4/172.17.0.2/tcp/40000/p2p/12↩
       ↪ D3KooWKiBurMyAHiJ2UkBAy1ZoCwxXxpZHVjTA6D9YLT8cmdny \
```

Para secuenciar el despliegue se usa el script a continuación, el script además de levantar los nodos, borra la persistencia de los clientes, inicializa la red con la gobernanza y actualiza el entorno con la nueva *id* de la gobernanza.

Código 6.10: Despliegue secuenciado

```
./city_hall.sh
echo "City Hall started"
sleep 1

./citizen_1.sh
echo "Citizen 1 started"
sleep 1

./citizen_2.sh
echo "Citizen 2 started"
sleep 1

./rec_company.sh
echo "Recycle Company started"
sleep 1

./container_1.sh
echo "Container 1 started"
sleep 1

```

```
./container_2.sh
echo "Container 2 started"
sleep 1

cd ..

rm -rf data/*

python -m scripts.load_governance
echo "Governance loaded"

echo "Deploy finished"
```

6.1.5. Cliente

Cada participante podrá llevar a cabo las operaciones definidas en los casos de estudio utilizando un cliente de línea de comandos (*shell*). Dependiendo del rol que desempeñe el participante, el cliente permitirá realizar diferentes tipos de operaciones, además de incorporar lógica adicional según sea necesario.

Todos los clientes utilizan *LevelDB* para almacenar datos auxiliares, como las asociaciones entre `subject_id` e identificadores digitales de ciudadanos, para almacenar solicitudes pendientes de aprobación o validación. Para la comunicación con el nodo correspondiente, se utiliza un *Adaptador REST* encargado de transformar los modelos de Pydantic en solicitudes REST y convertir las respuestas en modelos más operables. El cliente sigue una estructura de Vista-Controlador, donde una clase llamada `Shell` se encarga de la interacción con el usuario, mientras que el `Client` se ocupa de llevar a cabo las operaciones correspondientes.

Además se hace uso de clases utilitarias varias:

- `Subject_Factory`: Transforma un sujeto en su correspondiente modelo de datos.
- `Persistence`: Facilita el uso de LevelDB por parte del controlador.

Shell

La clase `Shell` se encarga de las operaciones de entrada y salida con el usuario. Esta clase se apoya en la librería *shlex* para segmentar en tokens la entrada y la librería *rich* para mostrar una salida mejor formateada.

Esta clase incorpora los comandos básicos: *exit*, *exit* y *clear*. Para llevar a cabo operaciones más complejas, se utiliza instancias de la clase `Command`, la cual representa las posibles acciones que el cliente puede realizar. Además, en caso de ser necesario, puede solicitar datos adicionales antes de llamar al controlador y contiene información sobre el comando correspondiente.

Un `Shell` tiene un diccionario en el que se utiliza el nombre del comando como clave y el comando como valor. Cada cliente especificará los comandos necesarios, además de uno inicializar información del cliente.

Client

`Client` actúa como controlador y se encarga de comunicarse con el nodo TAPLE y LevelDB. Además, aplica lógica de aplicación adicional, gestiona errores y hace uso de `Persistence` para abstraer el uso de LevelDB, `Subject_Factory` para convertir los sujetos recibidos en modelos específicos y `RestConnector` para la comunicación con el nodo. Existe un cliente para cada tipo de actor:

- *Empresa de Reciclaje*: Es el cliente más simple, permite listar las zonas de reciclaje, ver el historial de reciclaje de todas las zonas o zonas individuales.
- *Zona de Reciclaje*: El cliente permite registrar la entrada de residuos al mismo tiempo que añade al ciudadano los puntos correspondientes.
- *Ayuntamiento*: Incluye la funcionalidad de la empresa de reciclaje, además de poder consultar la información de un ciudadano, listar los ciudadanos, validar las solicitudes de actualización de información de un ciudadano y confirmar el canjeo de recompensas a cambio de los puntos.
- *Ciudadano*: El cliente permite dar de alta al ciudadano si no existe, solicitar actualizar su propia información, ver sus puntos actuales, ver su historial de puntos y canjear puntos por recompensas en la tienda del ayuntamiento.

Uso del cliente

Existen seis clientes, uno pora cada nodo: *client_citizen[1,2]*, *client_recycle_zone[1,2]*, *client_recycle_company* y *client_city_hall*. La ejecución de un cliente es la siguiente:

Código 6.11: Ejecución del cliente

```
114 python client_<actor>
```

Una vez que se inicie el programa escribiendo el comando "`help`", se mostrarán los comandos disponibles. Para obtener información adicional sobre cada comando, se puede escribir "`<command> help`", lo que mostrará información adicional sobre el comando. El primer comando que se debe ejecutar es "`init`" para inicializar el controlador con la información necesaria.

Adaptador REST

La clase `RestConnector` es responsable de establecer la comunicación HTTP con los nodos utilizando la biblioteca *requests*. Cuenta con una función para cada entrada de

la Interfaz de Programación de Aplicaciones (API), lo que facilita la abstracción del controlador. Internamente, convierte las respuestas de la API en modelos correspondientes o en errores. Además, adapta los datos de entrada al formato esperado por los nodos.

7. Conclusiones y vías futuras

El trabajo descrito en este documento explora la tecnología DLT, en particular las plataformas blockchain. Creemos que esta tecnología jugará un papel muy importante en el futuro inmediato como lo prueban las iniciativas gubernamentales como la *Blockchain Strategy* de la UE y el interés de las grandes empresas de software y consultoras, por ejemplo IBM y Accenture. Las aplicaciones de blockchain en empresas, instituciones y para el beneficio de los ciudadanos en general son cruciales para la nueva sociedad que está emergiendo con la actual transformación digital: gestión de identidades digitales, servicios financieros, trazabilidad de procesos/activos y gestión de historiales médicos, entre otras.

El TFG presentado en esta Memoria ha explorado las posibilidades de la blockchain a través de un caso de estudio implementado con la recién lanzada plataforma blockchain TAPLE. Por tanto se trata de una de las primeras pruebas de concepto de dicha plataforma. Hemos sido capaces de vislumbrar las posibilidades de TAPLE para crear aplicaciones en los dominios para los cuales se ha creado, aunque la plataforma todavía está en una fase inicial y no hemos podido probar características como los testigos, borrado de un microledger o los smart contracts dado que todavía no están disponibles. Por otro lado, en cuanto al uso de blockchain creemos que deberá superar limitaciones actuales como la necesidad de un nivel de participación suficiente dentro de la red para garantizar la integridad y seguridad, el consumo computacional de los algoritmos de consenso y la necesidad de adoptar una nueva infraestructura.

De la experiencia vivida con nuestro trabajo, hemos observado la necesidad de una mayor disponibilidad de casos de uso bien documentados de blockchain y de DLT en general. Este podría ser un trabajo futuro interesante como parte de la colaboración, en ese ámbito, entre la universidad y las empresas: la experimentación con el desarrollo de aplicaciones blockchain y su documentación en informes qué describan el diseó e implementación, así como los beneficios e inconvenientes encontrados. El TFG aaquí presentado puede servir como un trabajo inicial en esa dirección y en el contexto de la plataforma TAPLE. También, se ha echado en falta el disponer de libros que introduzcan a la tecnología DLT y blockchain. La publicación de casos de estudios y de buenos libros será clave para la adopción de estas tecnologías. Cabe señalar, que el grupo ModelUM continuará colaborando con la empresa Open Canarias y está prevista la definición de un trabajo para una tesis de máster que se realizaría en el marco del máster de Ingeniería del Software que se oferta por primera vez el próximo curso en la Facultad de Informática de la Universidad de Murcia.

En resumen el uso de la tecnología blockchain es muy prometedor y revolucionara

parte de la industria y el internet tal como lo conocemos, abriendo puertas a nuevos modelos de aplicaciones y la descentralización del control de los datos. Cabe destacar que la concepción popular sobre la blockchain es muy cerrada y sus aplicaciones se extienden mucho más allá de las criptomonedas o los NFTs.

Bibliografía

[1] TAPLE Documentation | TAPLE Documentation, February 2023. URL https://www.taple.es. [Last Online; accessed 4. May 2023].

[2] MongoDB. *Blockchain, Ledgers, and Databases :A Guide to Navigate the Confusion*. MongoDB, 1633 Broadway, 38th Floor, New York, NY, 10019 United State, 2022. URL https://webassets.mongodb.com/_com_assets/collateral/mongodb_blockchain.pdf.

[3] Contributors to Wikimedia projects. Distributed ledger - Wikipedia, April 2023. URL https://en.wikipedia.org/w/index.php?title=Distributed_ledger&oldid=1152463870. [Online; accessed 13. Jun. 2023].

[4] Manav Gupta. *Blockchain for dummies, 2nd IBM Limited Edition* . John Wiley & Sons, Inc., Hoboken, NJ 07030-5774, 2018. URL https://www.ibm.com/downloads/cas/36KBMBOG.

[5] Contributors to Wikimedia projects. Blockchain - Wikipedia, May 2023. URL https://en.wikipedia.org/w/index.php?title=Blockchain&oldid=1153640161. [Online; accessed 16. May 2023].

[6] Contributors to Wikimedia projects. Byzantine fault - Wikipedia, March 2023. URL https://en.wikipedia.org/w/index.php?title=Byzantine_fault&oldid=1142263534. [Online; accessed 13. Jun. 2023].

[7] Colaboradores de los proyectos Wikimedia. Pbft - Wikipedia, la enciclopedia libre, June 2023. URL https://es.wikipedia.org/w/index.php?title=Pbft&oldid=151692566. [Online; accessed 13. Jun. 2023].

[8] Contributors to Wikimedia projects. Proof of authority - Wikipedia, April 2023. URL https://en.wikipedia.org/w/index.php?title=Proof_of_authority&oldid=1150778360. [Online; accessed 17. May 2023].

[9] Contributors to Wikimedia projects. Proof of work - Wikipedia, May 2023. URL https://en.wikipedia.org/w/index.php?title=Proof_of_work&oldid=1154532265. [Online; accessed 17. May 2023].

[10] Anh Ngoc Quang Huynh, Duy Duong, Tobias Burggraf, Hien Thi Thu Luong, and Nam Huu Bui. Energy Consumption and Bitcoin Market. *Asia-Pac. Financ. Markets*, 29(1):79–93, March 2022. ISSN 1573-6946. doi: 10.1007/s10690-021-09338-4.

[11] Contributors to Wikimedia projects. Proof of stake - Wikipedia, May 2023. URL https://en.wikipedia.org/w/index.php?title=Proof_of_stake&oldid=1153534723. [Online; accessed 17. May 2023].

[12] Contributors to Wikimedia projects. Proof of personhood - Wikipedia, May 2023. URL https://en.wikipedia.org/w/index.php?title=Proof_of_personhood&oldid=1153537436. [Online; accessed 17. May 2023].

[13] Contributors to Wikimedia projects. Smart contract - Wikipedia, May 2023. URL https://en.wikipedia.org/w/index.php?title=Smart_contract&oldid=1152983560. [Online; accessed 13. Jun. 2023].

[14] Ayesha Shahnaz, Usman Qamar, and Ayesha Khalid. Using Blockchain for Electronic Health Records. *IEEE Access*, 7:147782–147795, October 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2946373.

[15] SmithSamuelM. Keri WhitePaper, June 2023. URL https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/KERI_WP_2.x.web.pdf. [Online; accessed 7. Jun. 2023].

[16] Lewis Tseng, Xinyu Yao, Safa Otoum, Moayad Aloqaily, and Yaser Jararweh. Blockchain-based database in an IoT environment: challenges, opportunities, and analysis. *Cluster Comput.*, 23(3):2151–2165, September 2020. ISSN 1573-7543. doi: 10.1007/s10586-020-03138-7.

[17] Contributors to Wikimedia projects. Non-fungible token - Wikipedia, May 2023. URL https://en.wikipedia.org/w/index.php?title=Non-fungible_token&oldid=1154874719. [Online; accessed 18. May 2023].

[18] From Farm to Blockchain: Walmart Tracks Its Lettuce, September 2018. URL https://web.archive.org/web/20181205103719/https://www.nytimes.com/2018/09/24/business/walmart-blockchain-lettuce.html. [Online; accessed 3. Jul. 2023].

[19] Senthil Nathan, Chander Govindarajan, Adarsh Saraf, Manish Sethi, and Praveen Jayachandran. Blockchain Meets Database: Design and Implementation of a Blockchain Relational Database. *arXiv*, March 2019. doi: 10.48550/arXiv.1903.01919.

[20] Contributors to Wikimedia projects. Blockchain-based database - Wikipedia, January 2023. URL https://en.wikipedia.org/w/index.php?title=Blockchain-based_database&oldid=1136115938. [Online; accessed 31. May 2023].

[21] Hyperledger Sawtooth, April 2023. URL https://sawtooth.hyperledger.org. [Online; accessed 5. Jun. 2023].

[22] hyperledger. Github Sawtooth, June 2023. URL `https://github.com/hyperledger/sawtooth-core`. [Online; accessed 5. Jun. 2023].

[23] Hyperledger – Open Source Blockchain Technologies, June 2023. URL `https://www.hyperledger.org`. [Online; accessed 5. Jun. 2023].

[24] Hyperledger Sawtooth Documentation, April 2023. URL `https://sawtooth.hyperledger.org/docs/1.2`. [Online; accessed 5. Jun. 2023].

[25] Case Studies – Hyperledger Foundation, June 2023. URL `https://www.hyperledger.org/learn/case-studies`. [Online; accessed 5. Jun. 2023].

[26] hyperledger. Github Fabric, June 2023. URL `https://github.com/hyperledger/fabric/tags`. [Online; accessed 5. Jun. 2023].

[27] Introduction — hyperledger-fabricdocs main documentation, June 2023. URL `https://hyperledger-fabric.readthedocs.io/en/release-2.5/whatis.html`. [Online; accessed 5. Jun. 2023].

[28] Features & Use Cases • • BigchainDB, June 2020. URL `https://www.bigchaindb.com/features`. [Online; accessed 7. Jun. 2023].

[29] Germany BigchainDB GmbH, Berlin. BigchainDB 2.0: The Blockchain Database. May 2018. URL `https://www.bigchaindb.com/whitepaper/bigchaindb-whitepaper.pdf`.

[30] Tracking (Autonomous) Provenance and Lifecycle Events (TAPLE), February 2023. URL `https://www.taple.es/whitepaper/taple-whitepaper.pdf`. [Last Online; accessed 4. May 2023].

[31] Alex Preukschat and Drummond Reed. *Self-Sovereign Identity: Decentralized Digital Identity and Verifiable Credentials*. Simon and Schuster, Riverside, NJ, USA, June 2021. ISBN 978-1-61729659-8. URL `https://books.google.es/books/about/Self_Sovereign_Identity.html?id=Nh4uEAAAQBAJ&redir_esc=y`.

[32] Open Canarias. Github: taple-client, June 2023. URL `https://github.com/opencanarias/taple-client`. [Online; accessed 14. Jun. 2023].

[33] Pydantic, June 2023. URL `https://docs.pydantic.dev/latest`. [Online; accessed 14. Jun. 2023].

[34] Peter Gonczol, Panagiota Katsikouli, Lasse Herskind, and Nicola Dragoni. Blockchain Implementations and Use Cases for Supply Chains-A Survey. *IEEE Access*, 8:11856–11871, January 2020. ISSN 2169-3536. doi: 10.1109/ACCESS.2020.2964880.

Lista de Acrónimos y Abreviaturas

API	Interfaz de Programación de Aplicaciones.
DAG	Grafos Acíclicos Dirigidos.
DLT	Distributed Ledger Technology.
DNI	Documento Nacional de Identidad.
IoT	Internet Of Things.
IP	Internet Protocol.
KERI	Key Event Receipt Infrastructure.
NIE	Número de Identidad de Extranjero.
NoSQL	Bases de datos no relacionales.
PBFT	Practical Byzantine Fault Tolerance.
PoA	Proof of Authority.
PoS	Proof of Stake.
PoW	Proof of Work.
REST	Transferencia de Estado Representacional.
SDK	Kit de Desarrollo de Software.
TAPLE	Tracking (Autonomous) of Provenance and Lifecycle Events.
TFG	Trabajo Final de Grado.
TFM	Trabajo Final de Máster.

A. Anexo I

A.1. Ciudadano

Esquema JSON de un ciudadano

Código A.1: Esquema de Ciudadano

```
{
   "definitions": {
      "ViaType": {
         "type": "string"
      },
      "Identidad": {
         "type": "object",
         "properties": {
            "nombre": {
               "type": "string",
               "description": "Nombre del ciudadano"
            },
            "apellidos": {
               "type": "string",
               "description": "Primer apellido del ciudadano"
            },
            "ni": {
               "type": "string",
               "description": "DNI del ciudadano",
               "minLength": 9,
               "maxLength": 9
            }
         },
         "required": [
            "nombre",
            "apellidos",
            "ni"
         ],
         "additionalProperties": false
      },
      "Direcion": {
         "type": "object",
```

```
        "properties": {
            "type": {
                "$ref": "#/definitions/ViaType",
                "description": "Tipo de vía"
            },
            "direccion": {
                "type": "string",
                "description": "Dirección"
            },
            "codigo_postal": {
                "type": "string",
                "description": "Código Postal",
                "minLength": 5,
                "maxLength": 5
            },
            "comunidad": {
                "type": "string",
                "description": "Comunidad/Región/Provincia"
            },
            "ciudad": {
                "type": "string",
                "description": "Ciudad"
            }
        },
        "required": [
            "type",
            "direccion",
            "codigo_postal",
            "comunidad",
            "ciudad"
        ],
        "additionalProperties": false
    },
    "Citizen": {
        "type": "object",
        "properties": {
            "identidad": {
                "$ref": "#/definitions/Identidad",
                "description": "Identidad del ciudadano"
            },
            "direccion": {
                "$ref": "#/definitions/Direcion",
                "description": "Dirección del ciudadano"
            },
            "cuid": {
```

```
                "type": "string",
                "description": "Código de identificación único del ciudadano",
                "minLength": 36,
                "maxLength": 36,
                "default": ""
            }
        },
        "required": [
            "identidad",
            "direccion",
            "cuid"
        ],
        "additionalProperties": false
    }
},
"$ref": "#/definitions/Citizen"
}
```

A.2. Traza de Reciclaje

Esquema JSON de una traza de reciclaje

Código A.2: Esquema de traza de reciclaje

```
{
    "type": "object",
    "title": "RecycleOperation",
    "properties": {
        "type": {
            "type": "string",
            "description": "Tipo de residuo"
        },
        "peso": {
            "type": "number",
            "description": "Peso de los residuos, en gramos",
            "exclusiveMinimum": 0
        },
        "container": {
            "type": "string",
            "description": "Id del contenedor",
            "minLength": 36,
            "maxLength": 36
        }
```

```
    },
    "required": [
        "type",
        "peso",
        "container"
    ],
    "additionalProperties": false
}
```

A.3. Transacción de Puntos

Esquema JSON de una transacción de puntos

Código A.3: Esquema de Transacción de Puntos

```
{
    "type": "object",
    "title": "PointTransaction",
    "properties": {
        "cuid": {
            "type": "string",
            "description": "Codigo de identificacion unico del ciudadano↩
                ↪ ",
            "minLength": 36,
            "maxLength": 36
        },
        "value": {
            "type": "integer",
            "description": "Valor de la operacion"
        },
        "motivo": {
            "type": "string",
            "description": "Motivo de la operacion"
        },
        "balance": {
            "type": "integer",
            "description": "Balance actual de puntos",
            "default": 0
        }
    },
    "required": [
        "cuid",
        "value",
        "motivo"
```

```
    ],
    "additionalProperties": false
}
```

B. Anexo II

Código B.1: Fichero de Governanza

```
{
   "members": [
      {
         "id": "b11a2304-c1a0-42b4-bafe-f22c73812051",
         "tags": {},
         "description": "Citizen 1",
         "key": "E6y61Ras_rC1DV_4obUnxGepq5U9S3OMlu9NL5gBkQbA"
      },
      {
         "id": "49038391-f970-4fae-90c7-dece4db3c131",
         "tags": {},
         "description": "Citizen 2",
         "key": "E5OTk3oqVy2-gyRXOJYRChrngV57E8TWeqR-kI0-yesU"
      },
      {
         "id": "City Hall",
         "tags": {},
         "description": "City Hall",
         "key": "Ekv_Je7a79pE7CUY2BHaXWnnrSc5B1RFw9zYeJlbyo-o"
      },
      {
         "id": "32c069c5-6ed7-4193-988c-0fc1e029552c",
         "tags": {},
         "description": "Container 1",
         "key": "ESGawmqW9bigN76XiRpW4fXhmcCLAsN9Lx-yBQW4o83c"
      },
      {
         "id": "abf385e8-fbdd-433d-80f6-1e6e16e42c87",
         "tags": {},
         "description": "Container 2",
         "key": "Ey3wRMJ22dcKRAYmQckOztDxA-V_JEUtbQ7ICyKHN3x8"
      },
      {
         "id": "Recycle Company",
         "tags": {},
```

```
36          "description": "Trash Company",
37          "key": "EQDMqaSKaY1F0gdIzeFozHeWiHorsV7sSecsXYOxGGAE"
38       }
39    ],
40    "schemas": [
41       {
42          "id": "RecycleOperation",
43          "tags": {},
44          "content": {
45             "type": "object",
46             "title": "RecycleOperation",
47             "properties": {
48                "type": {
49                   "type": "string",
50                   "description": "Tipo de residuo"
51                },
52                "peso": {
53                   "type": "number",
54                   "description": "Peso de los residuos, en gramos",
55                   "exclusiveMinimum": 0
56                },
57                "container": {
58                   "type": "string",
59                   "description": "Id del contenedor",
60                   "minLength": 36,
61                   "maxLength": 36
62                }
63             },
64             "required": [
65                "type",
66                "peso",
67                "container"
68             ],
69             "additionalProperties": false
70          }
71       },
72       {
73          "id": "PointTransaction",
74          "tags": {},
75          "content": {
76             "type": "object",
77             "title": "PointTransaction",
78             "properties": {
79                "cuid": {
80                   "type": "string",
```

```
                    "description": "Codigo de identificacion unico del ciudadano",
                    "minLength": 36,
                    "maxLength": 36
                },
                "value": {
                    "type": "integer",
                    "description": "Valor de la operacion"
                },
                "motivo": {
                    "type": "string",
                    "description": "Motivo de la operacion"
                },
                "balance": {
                    "type": "integer",
                    "description": "Balance actual de puntos",
                    "default": 0
                }
            },
            "required": [
                "cuid",
                "value",
                "motivo"
            ],
            "additionalProperties": false
        }
    },
    {
        "id": "Citizen",
        "tags": {},
        "content": {
            "definitions": {
                "ViaType": {
                    "type": "string"
                },
                "Identidad": {
                    "type": "object",
                    "properties": {
                        "nombre": {
                            "type": "string",
                            "description": "Nombre del ciudadano"
                        },
                        "apellidos": {
                            "type": "string",
```

```
                        "description": "Primer apellido del ciudadano"
                    },
                    "ni": {
                        "type": "string",
                        "description": "DNI del ciudadano",
                        "minLength": 9,
                        "maxLength": 9
                    }
                },
                "required": [
                    "nombre",
                    "apellidos",
                    "ni"
                ],
                "additionalProperties": false
            },
            "Direcion": {
                "type": "object",
                "properties": {
                    "type": {
                        "$ref": "#/definitions/ViaType",
                        "description": "Tipo de vía"
                    },
                    "direccion": {
                        "type": "string",
                        "description": "Dirección"
                    },
                    "codigo_postal": {
                        "type": "string",
                        "description": "Código Postal",
                        "minLength": 5,
                        "maxLength": 5
                    },
                    "comunidad": {
                        "type": "string",
                        "description": "Comunidad/Región/Provincia"
                    },
                    "ciudad": {
                        "type": "string",
                        "description": "Ciudad"
                    }
                },
                "required": [
                    "type",
```

```
                        "direccion",
                        "codigo_postal",
                        "comunidad",
                        "ciudad"
                    ],
                    "additionalProperties": false
                },
                "Citizen": {
                    "type": "object",
                    "properties": {
                        "identidad": {
                            "$ref": "#/definitions/Identidad",
                            "description": "Identidad del ciudadano"
                        },
                        "direccion": {
                            "$ref": "#/definitions/Direcion",
                            "description": "Dirección del ciudadano"
                        },
                        "cuid": {
                            "type": "string",
                            "description": "Código de identificación ú↩
                                ↪ nico del ciudadano",
                            "minLength": 36,
                            "maxLength": 36,
                            "default": ""
                        }
                    },
                    "required": [
                        "identidad",
                        "direccion",
                        "cuid"
                    ],
                    "additionalProperties": false
                }
            },
            "$ref": "#/definitions/Citizen"
        }
    }
],
"policies": [
    {
        "id": "governance",
        "validation": {
            "quorum": 0.1,
            "validators": [
```

```
                "Ekv_Je7a79pE7CUY2BHaXWnnrSc5B1RFw9zYeJlbyo-o",
                "E5OTk3oqVy2-gyRXOJYRChrngV57E8TWeqR-kI0-yesU",
                "E6y61Ras_rC1DV_4obUnxGepq5U9S3OMlu9NL5gBkQbA",
                "ESGawmqW9bigN76XiRpW4fXhmcCLAsN9Lx-yBQW4o83c",
                "Ey3wRMJ22dcKRAYmQckOztDxA-V_JEUtbQ7ICyKHN3x8",
                "EQDMqaSKaY1F0gdIzeFozHeWiHorsV7sSecsXYOxGGAE"
            ]
        },
        "approval": {
            "quorum": 0.1,
            "approvers": [
                "Ekv_Je7a79pE7CUY2BHaXWnnrSc5B1RFw9zYeJlbyo-o",
                "E5OTk3oqVy2-gyRXOJYRChrngV57E8TWeqR-kI0-yesU",
                "E6y61Ras_rC1DV_4obUnxGepq5U9S3OMlu9NL5gBkQbA",
                "ESGawmqW9bigN76XiRpW4fXhmcCLAsN9Lx-yBQW4o83c",
                "Ey3wRMJ22dcKRAYmQckOztDxA-V_JEUtbQ7ICyKHN3x8",
                "EQDMqaSKaY1F0gdIzeFozHeWiHorsV7sSecsXYOxGGAE"
            ]
        },
        "invokation": {
            "owner": {
                "allowance": true,
                "approvalRequired": false
            },
            "set": {
                "allowance": false,
                "approvalRequired": true,
                "invokers": []
            },
            "all": {
                "allowance": false,
                "approvalRequired": true
            },
            "external": {
                "allowance": false,
                "approvalRequired": true
            }
        }
    },
    {
        "id": "Citizen",
        "validation": {
            "quorum": 1.0,
            "validators": [
                "Ekv_Je7a79pE7CUY2BHaXWnnrSc5B1RFw9zYeJlbyo-o"
```

```
            ]
        },
        "approval": {
            "quorum": 1.0,
            "approvers": [
                "Ekv_Je7a79pE7CUY2BHaXWnnrSc5B1RFw9zYeJlbyo-o"
            ]
        },
        "invokation": {
            "owner": {
                "allowance": true,
                "approvalRequired": true
            },
            "set": {
                "allowance": false,
                "approvalRequired": true,
                "invokers": []
            },
            "all": {
                "allowance": false,
                "approvalRequired": true
            },
            "external": {
                "allowance": false,
                "approvalRequired": true
            }
        }
    },
    {
        "id": "RecycleOperation",
        "validation": {
            "quorum": 0.1,
            "validators": [
                "EQDMqaSKaY1F0gdIzeFozHeWiHorsV7sSecsXYOxGGAE",
                "Ekv_Je7a79pE7CUY2BHaXWnnrSc5B1RFw9zYeJlbyo-o"
            ]
        },
        "approval": {
            "quorum": 0.1,
            "approvers": [
                "EQDMqaSKaY1F0gdIzeFozHeWiHorsV7sSecsXYOxGGAE"
            ]
        },
        "invokation": {
            "owner": {
```

```
                "allowance": true,
                "approvalRequired": false
            },
            "set": {
                "allowance": false,
                "approvalRequired": false,
                "invokers": [
                ]
            },
            "all": {
                "allowance": false,
                "approvalRequired": true
            },
            "external": {
                "allowance": false,
                "approvalRequired": true
            }
        }
    },
    {
        "id": "PointTransaction",
        "validation": {
            "quorum": 0.1,
            "validators": [
                "ESGawmqW9bigN76XiRpW4fXhmcCLAsN9Lx-yBQW4o83c",
                "Ey3wRMJ22dcKRAYmQckOztDxA-V_JEUtbQ7ICyKHN3x8",
                "Ekv_Je7a79pE7CUY2BHaXWnnrSc5B1RFw9zYeJlbyo-o"
            ]
        },
        "approval": {
            "quorum": 0.1,
            "approvers": [
                "ESGawmqW9bigN76XiRpW4fXhmcCLAsN9Lx-yBQW4o83c",
                "Ey3wRMJ22dcKRAYmQckOztDxA-V_JEUtbQ7ICyKHN3x8",
                "Ekv_Je7a79pE7CUY2BHaXWnnrSc5B1RFw9zYeJlbyo-o"
            ]
        },
        "invokation": {
            "owner": {
                "allowance": true,
                "approvalRequired": true
            },
            "set": {
                "allowance": true,
                "approvalRequired": false,
```

```
                    "invokers": [
                        "ESGawmqW9bigN76XiRpW4fXhmcCLAsN9Lx-yBQW4o83c",
                        "Ey3wRMJ22dcKRAYmQckOztDxA-V_JEUtbQ7ICyKHN3x8"
                    ]
                },
                "all": {
                    "allowance": false,
                    "approvalRequired": true
                },
                "external": {
                    "allowance": false,
                    "approvalRequired": true
                }
            }
        }
    ]
}
```