Leveraging Ontochains for Distributed Public Transit Ticketing

Preece, Joseph; Morris, Chris; Easton, John

DOI:
10.1049/blc2.12071

License:
Creative Commons: Attribution (CC BY)

Document Version
Publisher's PDF, also known as Version of record

Citation for published version (Harvard):
Preece, J, Morris, C & Easton, J 2024, 'Leveraging Ontochains for Distributed Public Transit Ticketing: An Investigation with the System for Ticketing Ubiquity with Blockchains', *IET Blockchain*. https://doi.org/10.1049/blc2.12071

Received: 27 June 2023 | Revised: 27 February 2024 | Accepted: 18 March 2024

DOI: 10.1049/blc2.12071

ORIGINAL RESEARCH

Leveraging ontochains for distributed public transit ticketing: An investigation with the system for ticketing ubiquity with blockchains

Joseph D. Preece | Christopher R. Morris | John M. Easton

Birmingham Centre for Railway Research and Education, University of Birmingham, Birmingham, UK

Correspondence
Joseph D. Preece, Birmingham Centre for Railway Research and Education, University of Birmingham, Birmingham B15 2TT, UK.
Email: j.d.preece@bham.ac.uk

Funding information
TruBlo Project (Third Open Call subgrant agreement), under European Union's Horizon 2020 Research and Innovation Program, Grant/Award Number: 957228

Abstract
Transport ticketing systems are crucial for enabling seamless, efficient, and sustainable mobility. However, traditional ticketing systems face limitations such as ticket fraud, lack of interoperability, and the inability to adapt to changes in the dynamic transport networks they issue tickets for. This paper presents new approaches to the system for ticketing ubiquity with blockchains (STUB), a novel smart transport ticketing solution that employs ontochains, a hybrid data structure combining blockchains and ontologies to form a type of distributed knowledge graph. STUB aims to address these limitations by providing a secure, transparent, and flexible platform for ticket issuance, validation, and management. We describe the key components and workflow of the STUB system, highlighting the use of transport network ontologies for modelling complex relationships within transportation systems and blockchain technologies for transport network ontology's state. Additionally, the implementation of Merkle proofs for efficient and secure validation between on-chain and off-chain ontological data is discussed. Finally, a simulated toy example is used as a lightweight proof-of-concept to demonstrate these capabilities. The proposed STUB system has the potential to significantly impact the future of transportation ticketing by offering a more seamless, interoperable, and user-friendly experience whilst addressing the challenges associated with traditional ticketing systems.

1 | INTRODUCTION

Transportation systems around the world are becoming increasingly interconnected, driven by the need for seamless, efficient, and sustainable mobility. Ticketing systems play a crucial role in facilitating smooth and reliable access to various transportation services, and enhancing access to mobility for customers across the world. However, traditional ticketing systems often suffer from issues such as a lack of interoperability, and limited flexibility in fare options. These challenges have led to growing interest in the development of smart transport ticketing solutions that harness emerging technologies to overcome the limitations of traditional systems.

One promising approach to smart transport ticketing involves the integration of blockchains, which offers a distributed, transparent, and tamper-proof platform for secure transactions. Blockchains have been widely recognized for their potential applications in various industries due to their inherent ability to provide trust and security. However, previous approaches have only dealt with tickets stored as tokens within the blockchain, failing to address the complexities of validation in multi-modal settings [1, 2]. One such approach is System for Ticketing Ubiquity with Blockchains (STUB), built on the Hyperledger Fabric platform [1].

This paper presents modifications to STUB that leverage ontochains, a nascent data structure that combines the advantageous properties of blockchains and ontologies to form a type of distributed knowledge graph (DKG). STUB 2.0 aims to address the limitations of traditional ticketing systems by providing a secure, transparent, and flexible solution for ticket issuance, validation, and management. By integrating ontochains for transport network representation and blockchains for secure transaction processing, STUB offers a comprehensive framework for the next generation of transport ticketing systems.

This paper is structured as follows:

- Section 2 provides a comprehensive literature review on the role of blockchains, and ontologies within transportation, and details existing transport ticketing systems;
- Section 3 addresses the shortcomings of previous implementations of STUB;
- Section 4 outlines the Transport Network Ontology (TOnNe), used to construct an ontochain about the transport network;
- Section 5 describes the architecture and workflow of the STUB system, elaborates on the use of Merkle proofs for ontochain validation within STUB, and describes the toy environment used to validate the methodologies and mechanisms explained in Section 5;
- Section 6 describes the economic feasibility of the platform;
- Section 7 concludes the paper, presenting a discussion of the main contributions, potential challenges, and comparison to other proposed solutions, and suggests directions for future research.

2 | BACKGROUND

2.1 | Blockchains

A blockchain (blockchain) is a distributed, digital ledger that records transactions in a secure, transparent and tamper-evident manner. Each "block" contains a portion of these transactions and a cryptographic link to the previous block, creating an unbreakable "chain" of data. The use of specialized consensus algorithms ensures that all users on the network agree on the latest state of the ledger, making it nearly impossible to alter or manipulate past transactions.

Blockchains have a variety of potential applications within the transport sector. Namiot et al. [3] discuss the use of blockchain in storing information related to the transport industry, such as vehicle operating conditions, which can be used in insurance telematics applications. Callefi et al. [4] identify seven capabilities enabled by blockchain in the context of transportation operations, including reliable data, data privacy, and decentralized data control. Eremina et al. [5] present a study on the use of blockchain in transport management, specifically in creating a decentralized network of road lane sharing in real-time. Astarita et al. [6] provide a literature review on the application of blockchain-based systems in transportation, highlighting its potential in various fields such as food track and trace, regulatory compliance, and smart vehicles' security.

With regards to ticketing specifically, the majority of literature addresses the issues within the events and entertainment sectors [7–9]. Ferrick [8] argues that blockchain can eliminate fraud, improve consumer sentiment, and reduce costs for exchanges, brokers, and concert-goers. Cha et al. [9] proposes a privacy-preserving blockchain-based ticketing service that can ensure the authenticity of purchased tickets and protect user privacy. Previous work of the author [1, 2, 10, 11] has identified the potential for blockchain within the transport ticketing sector, storing tickets as tokens transacted on a blockchain network to enable transparent proof-of-ownership (PoO) and proof-of-validity (PoV) on a multi-modal transport network (tn), alongside Nguyen et al.[12], who propose a "decentralized network in which the transportation providers can verify and confirm their tickets through a blockchain containing smart contracts as tickets".

2.2 | Ontologies

An ontology (ontology)[1] is a formal, graphical representation of knowledge within a particular domain, which includes a set of concepts and categories, and the relationships between them. ontologys enable machines to understand the meaning and context of data, by providing a common vocabulary that can be used to annotate and classify information. They are often used in artificial intelligence and the semantic web to enable intelligent reasoning and decision making. In computer science, the accepted definition of an ontology is that given by Gruber [13], namely that "an ontology is an explicit specification of a conceptualization" in which a conceptualization is defined as "the objects, concepts, and other entities that are presumed to exist in some area of interest and the relationships that hold them".

Previous work, notably the InteGRail (integrail) European project [14] which started work in 2005, investigated the use of ontology to integrate data in the rail domain. This study found it to be advantageous as a means of data interchange and produced a simple ontology. The advantages of an ontology approach have been discussed elsewhere, in the context of passenger information by Verstichel et al. [15] or remote condition monitoring by Tutcher et al. [16]. These papers find significant benefits to all stakeholders from the ease with which data can be exchanged. Within other industries, significantly more progress has been made, with biomedical research and medical data exchange have long been leaders in this field. The gene ontology from that domain, introduced by Smith and Kumar [17], has been in use since 2004, allowing researchers in the field of genetics to share information more easily. Building on that previous work this project will encode information about public transport systems to an ontology and that is what will be stored on the ledger.

The core data to be modelled within the context of STUB captures the current state of a transport network. Transport network data is well suited to graph-based representation, and this approach is widely used for modelling transport networks of all types. In STUB, we propose to go further than a simple graph representation, by using a linked data approach to

[1] In our paper, we use the term "ontology" to encompass both traditional ontologies and knowledge graphs, acknowledging the broader scope within the field of information science and technology.

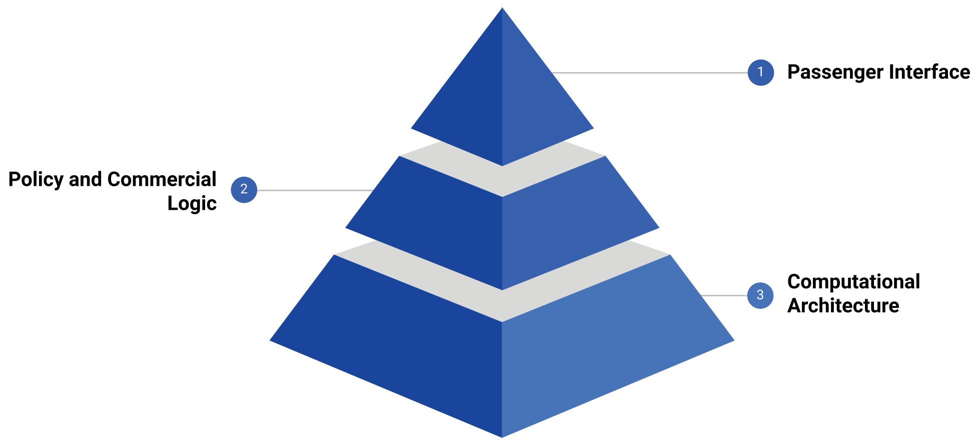

FIGURE 1 Ticketing infrastructure in the UK.

capture not just the raw data but, but also derived contextual information; the relationship between data, information and inferred knowledge, as first presented by Sharma [18].

2.3 | Ontochains

Next Generation Internet (NGI)['s] ONTOCHAIN (OC) is a software framework that combines ontologys and blockchains to provide a DKG for managing and sharing data [19, 20]. The ontology provides a common framework for defining and representing concepts and relationships within a domain, whilst the blockchain provides a tamper-evident and immutable ledger for recording and verifying changes to the ontology. Together, they enable the creation of distributed applications that can share and reason about complex data in a secure and transparent manner, without the need for a centralized authority.

OC comprises a novel protocol suite grouped into high-level application protocols and lower-level core protocols. Papaioannou and Stamoulis [21] studies the business model of OC, and establishes that open source licensing enables the joint exploitation of decentralized software frameworks. The OC blockchain-based software framework is being built by many companies, each having its own business agenda [22–29]. The work of STUB has been developed alongside from the OC research framework, and henceforth these data structures will be referred to as ontochains (ontochains), without necessarily linking to the work of OC directly.

2.4 | Ticketing

2.4.1 | Architecture

In the current ticketing infrastructure within the United Kingdom (UK)['s] railway system, the operational framework is divided into three distinct layers each serving a particular set of functions. These layers, developed and put into operation during the early stages of rail privatization, present a traditional (yet now increasingly inadequate) structure that is illustrated in Figure 1.

Layer 1: Passenger interface
This layer is the most visible and directly interactive component of the ticketing system. It encompasses all passenger-facing elements such as ticketing applications on smartphones and other devices, ticket offices where travellers can interact with sales staff, and ticket machines that provide self-service options. Layer 1 is designed to be user-friendly and accessible, aiming to facilitate the ticket purchase and validation process for passengers.

Layer 2: Policy and commercial logic
The intermediary layer, Layer 2, acts as the essential link that bridges Layers 1 and 3. It is responsible for executing the policy, business, and commercial logic that informs the fare structures displayed and offered to passengers. This layer translates the complex regulations and pricing strategies into understandable and applicable rules for ticket vending.

Layer 3: Computational architecture
Layer 3 represents the computational backbone of the ticketing system. It includes the servers, databases, and processing algorithms that handle the vast amounts of data and transactions necessary for ticket issuance and management. This layer is currently bogged down by legacy technologies and concepts that were introduced during the initial phase of rail privatization. As a result, it often lacks the capacity to adapt swiftly to new technologies or changes in transport policy, ultimately affecting the overall agility of the ticketing infrastructure.

A significant challenge in the current structure is the deep entanglement of Layers 2 and 3. This interweaving means that efforts to update or modify the ticketing system often encounter substantial resistance, as changes to policy or business logic (Layer 2) are not easily decoupled from the computational processes (Layer 3). This complexity renders it difficult to enhance one layer without necessitating extensive, intricate modifications to the other.

Without a clear separability, the system is resistant to change, lacks the necessary flexibility to rapidly evolve, and restricts adaptive responses to new market conditions or technological advancements.

2.4.2 | Standards

Previous literature collectively suggests that smart ticketing standards for transport already exist and have been implemented in various ways, although problems still remain. Blythe [30, 31] discuss the use of smart cards for payment and access to transport services, with the latter paper highlighting the emergence of interoperability standards through ITSO. Turner and Wilson [32] discuss the UK government's vision for a seamless transport ticketing infrastructure by 2020, built on smart card ticketing technologies, and the challenges of integration. (However, we note this has not yet been achieved). Gnoni et al. [33] propose an Integrated Mobility System (IMS) that uses Radio Frequency Identification (RFID) technology to improve ticketing management in a public transport network.

In more recent times, the European Union (EU) declared 2018 as the year of modality. Finger et al. [34] state that "the attainment of seamless multi-modal door-to-door mobility has emerged as a clear priority on the EU policy agenda", and that "different approaches to ticketing and payment systems have been observed to date across the different EU member states, and, in some instances, even across different regions of the same country". However, they proceed to state that "it is becoming increasingly clear, however, that an overarching EU framework may be needed for the successful implementation of multi-modal transport especially in cross-border contexts".

Prior to this, the EU published Council Regulation No. 2017/1926 (eu2017), requiring all transport service providers (TSPs) operating within the bounds of the EU to provide access to their data in specific formats. However, Scrocca et al. [35] note that "[TSPs] have a poor knowledge of the EU regulation", and that they "deem the conversion of their data to these standards as technically complex". As time has progressed, projects have started to produce tangible results, notably projects within the Shift2Maas, Ride2Rail, and IP4MaaS under the Innovation Programme 4 (IP4) of Europe's Rail. However, widespread usage remains limited, likely due to the difficulties raised by Scrocca et al. [35]. We note these difficulties, and have decided to pursue the initial concepts of STUB with the restrictions of such standards, allowing it to develop more organically.

Aside from ticketing standards, the General Transit Feed Specification (GTFS) is an open data format that empowers public transit agencies to publish their transit schedules and associated geographic information in a machine-readable format [36]. Initially devised by Google in partnership with the Portland TriMet transit agency, GTFS has evolved into a global standard used by thousands of public transport providers. GTFS data enables the integration of transit information into a wide array of applications, notably in journey planning software and mobility apps, thereby greatly enhancing the accessibility and user experience for transit riders. The specification encompasses information about routes, stops, trip schedules, and various transit modes, making public transportation data uniformly available and fostering innovative solutions for urban mobility. With its widespread adoption, GTFS has been instrumental in streamlining how riders access information on transit systems worldwide, contributing to more efficient and user-friendly public transportation networks.

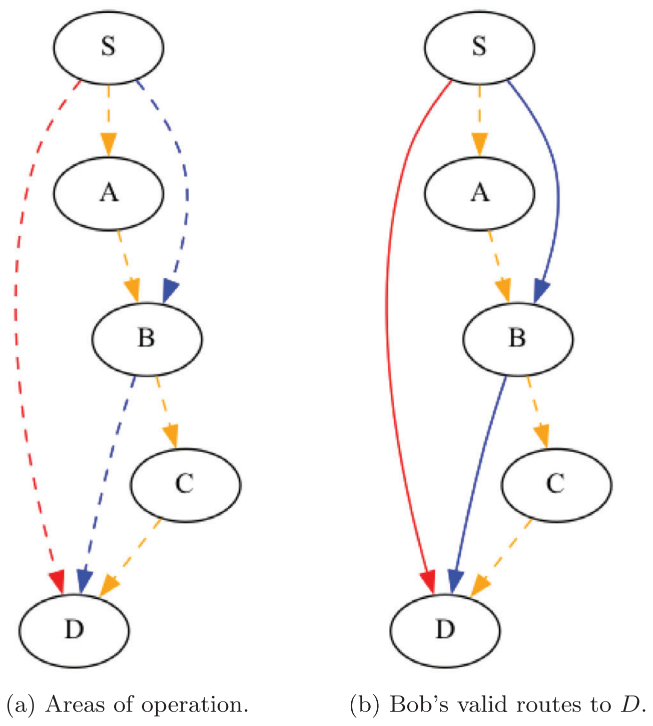

FIGURE 2 A toy example of a simplified tn.

3 | ADDRESSING THE LIMITATIONS OF STUB 1.0

Whilst the initial version of STUB made significant strides towards creating a distibuted, secure, and transparent ticketing system [1], it had notable limitations, primarily stemming from the lack of a consensus on the tn itself. This absence of a structured knowledge representation model for the transport network led to several challenges in validating tickets and coordinating between different transportation modes and TSPs.

To help understand the issue, let us establish a toy tn. Figure 2 illustrates a simple tn with five stations: the starting station S; the destination station D; and three interim stations A, B, and C. Three TSPs operate within this network, named after their illustrated colours of red, yellow, and blue. Each TSP represents various styles of transport service; for example, red operate a direct service $S \to D$ much like a high-speed rail line, whilst yellow operate a local-stopping service $S \to A \to B \to C \to D$ much like a bus would. We assume that red and blue have a pre-agreed partnership to allow passengers to use a single ticket between their services.

Now assume a customer (customer) (named Bob) buys a ticket from the red TSP to travel $S \to D$. This ticket should detail important information about his journey, including the starting stop, destination stop, the day of travel, and any network, TSP, or service restrictions to prevent Bob from using transport he does not have permission to use. In this example, the ticket allows him to travel from S to D, on the red service $S \to D$, or the blue service $S \to B \to D$, as red and blue have reached a prior agreement to allow this. Services offered by yellow are therefore invalid. Figure 2(b) illustrates the valid routes in solid lines. Bob will need to prove the validity of this ticket when travelling, in some cases to TSPs that did not vend the original ticket (blue).

Capturing this information within a blockchain is trivial, and STUB 1.0 used a smart contract architecture to capture journey metadata that was pre-agreed by the TSPs. This was an adequate solution for the small testing networks used during the development of the proof-of-concept (PoC). However, proving the validity for a complex transport network required extensive preagreements and knowledge about the tn prior to engagement. Each participating TSP had to establish explicit agreements with other TSPs, resulting in a complex web of interdependencies that hindered the efficient and timely processing of ticket validation. This complexity also increased the potential for errors and inconsistencies, as the system relied heavily on manual coordination and communication between parties.

Furthermore, the absence of knowledge of the tn hindered the system's flexibility and adaptability. As tns continuously evolve due to the introduction of new services, infrastructure changes, and policy updates, a system without a robust knowledge model struggles to keep up with this dynamic environment. This lack of adaptability may lead to outdated information, reduced efficiency, and an increased potential for discrepancies in ticket validation processes. With the blockchain already achieving the PoO, it provides the ideal framework to share this knowledge amongst the stakeholders. Yet blockchains are not well-suited to querying such complex data structures in an efficient manner; rather, they excel in sharing the data and ensuring its provenance and ownership.

4 | THE TRANSPORT NETWORK ONTOLOGY

To address these limitations, STUB 2.0 incorporates the TOnNe, enabling a more structured representation of the underlying transportation system. This addition enhances the ticket validation process, improves the passenger experience by providing a unified view of transportation options, and allows the system to adapt more readily to changes in the transport network. By integrating the TOnNe with the secure transaction capabilities of blockchain technology, STUB 2.0 offers a more comprehensive and efficient solution for transport ticketing. This section focuses on the construction of the TOnNe itself; please refer to Section 5 to see how it is integrated within the STUB ecosystem as a DKG.

Table 1 establishes the terminology used for a typical tn.

Consider a tn with one TSP named *TSP1*. Within this tn, there are two stops (stops): North Terminal and South Terminal. There is one service (service) named Southbound, which makes two planned stops (planned stops). Figure 3 illustrates the TOnNe that represents the tn.

We see that the TOnNe is made of a number of vertices that represent the subjects and objects, connected by edges that represent predicates. These subjects, predicates, and objects are the basis of triples, which is how graph databases store this information before parsing it into graphs. Figure 4 demonstrates how this information can be stored within the Turtle format.

TABLE 1 The terminology used by a typical transport network.

Transport Service Provider	A company that offers transportation services to individuals, businesses, or organisations. TSPs can provide a range of transportation options and are responsible for planning, scheduling, and providing transportation services, as well as managing vehicles and infrastructure.
Stop	A physical location where a transport service stops to pick up or drop off passengers, such as bus stops, train stations, and airports.
Service	A scheduled or on-demand service that moves people, goods, or vehicles from one location to another. This can include services like buses, trains, and subways
Planned stop	A pre-determined location where a transport service will stop to pick up or drop off passengers. These stops are scheduled in advance and may be listed on a transport service's route map or timetable. Examples of planned stops on a bus route might include designated bus stops or rail stations.

5 | PIECING THE PUZZLE TOGETHER: STUB 2.0

5.1 | System architecture

The STUB system is designed with a modular architecture that consists of two main components, the blockchain and ontology (implemented within a DKG), to form the ontochain. Table 2 details the components and their interactions within the STUB system architecture. Figure 5 illustrates the system architecture for STUB.

By integrating these components, the STUB system architecture offers a robust and flexible solution for smart transport ticketing. The combination of blockchain technology and ontology-based modelling enables STUB to provide a secure, transparent, and interoperable platform for ticket issuance, validation, and management, addressing the limitations of traditional ticketing systems and STUB 1.0.

5.1.1 | A note on decentralization and distribution

While the terms 'decentralized' and 'distributed' are often used interchangeably, it is important to distinguish their

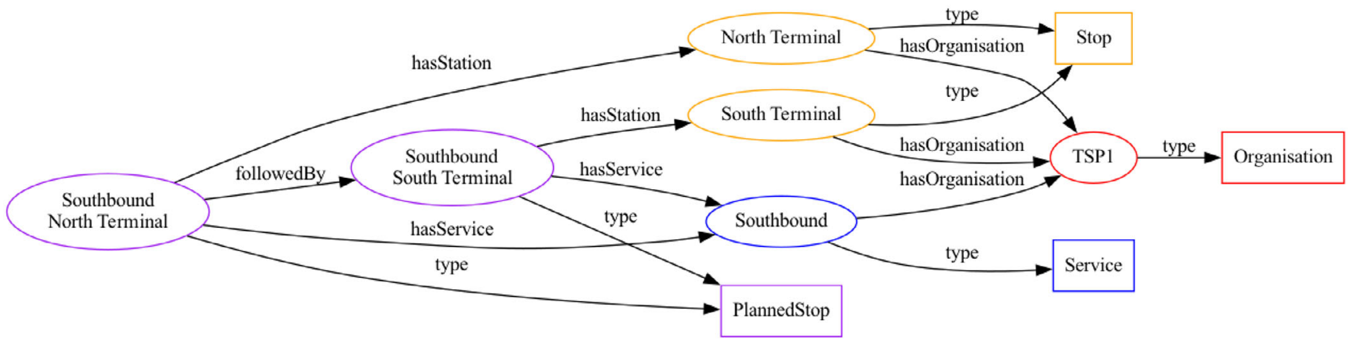

FIGURE 3 A toy example of a TOnNe.

```
    @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
    @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
    @prefix ex: <http://example.org/> .

    ex:Organisation a rdfs:Class .
    ex:Stop a rdfs:Class .
    ex:Service a rdfs:Class .
    ex:PlannedStop a rdfs:Class .

    ex:TSP1 a ex:Organisation ;
        ex:hasStops ex:NorthTerminal, ex:SouthTerminal ;
        ex:hasService ex:Southbound .

    ex:NorthTerminal a ex:Stop ;
        ex:type ex:TSP1 ;
        ex:hasStation ex:SouthboundNorthTerminal .

    ex:SouthTerminal a ex:Stop ;
        ex:type ex:TSP1 ;
        ex:hasStation ex:SouthboundSouthTerminal .

    ex:Southbound a ex:Service ;
        ex:type ex:TSP1 ;
        ex:hasService ex:SouthboundNorthTerminal, ex:SouthboundSouthTerminal .

    ex:SouthboundNorthTerminal a ex:PlannedStop ;
        ex:type ex:Southbound ;
        ex:hasStation ex:NorthTerminal ;
        ex:hasService ex:Southbound ;
        ex:followedBy ex:SouthboundSouthTerminal .

    ex:SouthboundSouthTerminal a ex:PlannedStop ;
        ex:type ex:Southbound ;
        ex:hasStation ex:SouthTerminal ;
        ex:hasService ex:Southbound .

```

FIGURE 4 The example TOnNe, represented in the Turtle format.

specific meanings, particularly in the context of technological platforms. STUB is architected with a distributed design, which enables a network of TSPs to seamlessly connect and partake in a shared data ecosystem. In this distributed network, data and resources can be spread across multiple nodes to enhance performance and reliability. However, it is crucial to highlight that unlike a truly decentralized system, which operates without a single control centre, STUB maintains a degree of centralized governance. There exist designated network administrators who assert control, oversee the network's function, and manage access—an essential distinction that plays a pivotal role in ensuring the security and

TABLE 2 The components and their interactions within the STUB system architecture.

Blockchain		STUB utilizes a Hyperledger Besu implementation as its underlying blockchain platform. The blockchain component consists of the following sub-components:
	Smart contracts	These programmable, self-executing agreements reside on the blockchain and govern the rules for ticket issuance and management. They ensure that transactions adhere to predefined conditions and facilitate secure, automated processing of the ticketing data.
	Transactions	Transactions represent various actions within the system, such as ticket purchases, validation events, and updates to the transport network information. They are stored immutably on the blockchain, providing a transparent and tamper-proof record of all activities.
Ontology		The ontology component models the complex relationships within the transport network, allowing for a more structured representation of the underlying system. It consists of the following sub-components:
	Triple store	STUB employs Stardog as its Triple Store, which stores the transport network information in the form of subject-predicate-object triples. This storage structure enables efficient querying and retrieval of the ontological data.
	Reasoning engine	This sub-component is responsible for processing the ontological data, validating its consistency, and performing reasoning tasks on the Stardog triple store. It allows the system to infer new knowledge and make informed decisions based on the existing transport network information.
Middleware		The middleware serves as the connecting layer between the blockchain and the ontology components, forming the basis of the ontochain. It extracts data from the smart contracts on the blockchain and constructs it within the Triple Store, allowing the ontological data to be reasoned on and utilized by other parts of the system.
API		An application programming interface (API) interacts with both the ontology parser and the smart contracts, enabling external applications to query and interact with the STUB system. This API serves as the primary interface for TSPs, administrators, and customers to access the ticketing system and leverage its functionality.

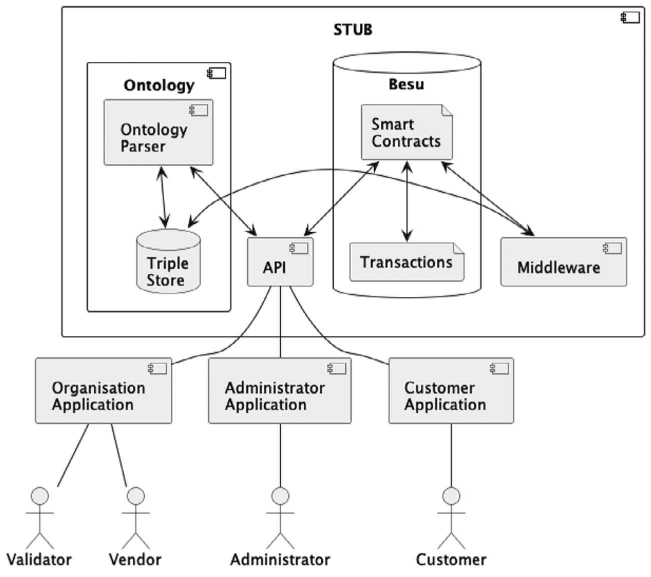

FIGURE 5 The deployment diagram for STUB.

integrity of the data exchanged within the platform's architecture.

5.2 | Smart contract architecture

The STUB system's smart contract architecture is designed to facilitate secure and efficient management of transport ticketing processes, representing the ontological data within the smart contracts to ensure the ownership and validation of the tn data within.

The relationships between these components, as depicted in Figure 6, demonstrate the modular and interconnected nature of the STUB smart contract architecture. By organizing the contracts in this manner, the system enables efficient querying, updating, and management of transport ticketing processes and on-chain TOnNe data, whilst maintaining a high level of security and transparency. Table 3 provides descriptions of these smart contracts in greater detail. For clarity, we have abstracted the arrays and mappings within each class to the typical representation in a Unified Modelling Language (UML) diagram, where:

- 1 denotes a relationship where one instance of a class is associated with exactly one instance of another class;
- * denotes a relationship where one instance of a class is associated with zero or more instances of another class.

Furthermore, we have only detailed the functions that modify state; there are additional classes that ensure function invocations are only carried out if requested by a valid administrator.

5.3 | Proof-of-validity

As the ontology is a representation of the TOnNe, TSPs need to ensure that their local version of the triple store matches the on-chain version, as this is the trusted version. To achieve this,

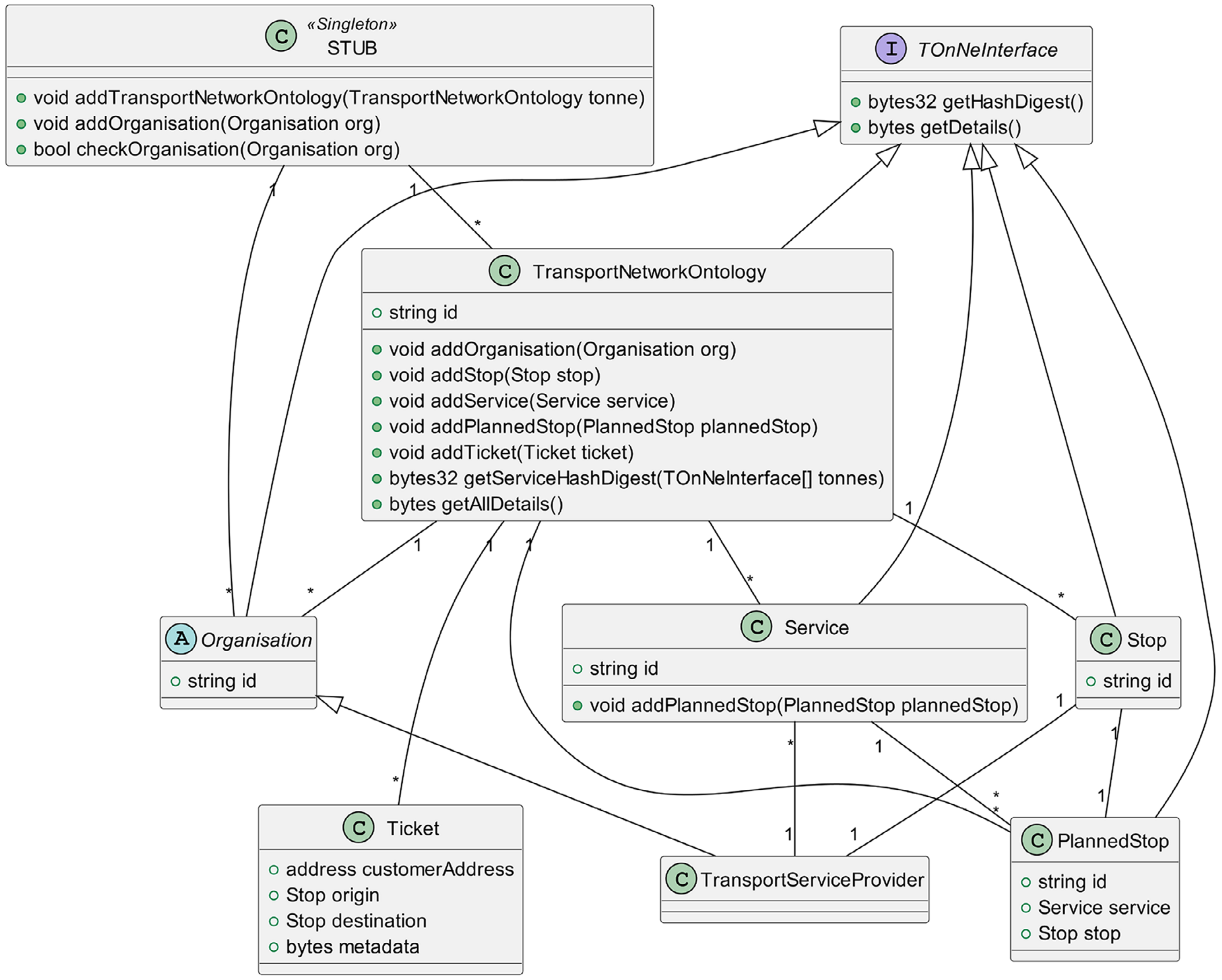
«Singleton»
STUB
void addTransportNetworkOntology(TransportNetworkOntology tonne)
void addOrganisation(Organisation org)
bool checkOrganisation(Organisation org)
TOnNeInterface
bytes32 getHashDigest()
bytes getDetails()
TransportNetworkOntology
string id
void addOrganisation(Organisation org)
void addStop(Stop stop)
void addService(Service service)
void addPlannedStop(PlannedStop plannedStop)
void addTicket(Ticket ticket)
bytes32 getServiceHashDigest(TOnNeInterface[] tonnes)
bytes getAllDetails()
Organisation
string id
Service
string id
void addPlannedStop(PlannedStop plannedStop)
Stop
string id
Ticket
address customerAddress
Stop origin
Stop destination
bytes metadata
TransportServiceProvider
PlannedStop
string id
Service service
Stop stop

FIGURE 6 The smart contract architecture for STUB.

the hash digests of the objects are checked down a hierarchy recursively, until a mismatched hash digest is found. This is principle of Merkle proofs, This simple concept, though complex through the implementation within the STUB ecosystem, is what drives the PoV. The trust between TSPs is on the Hyperledger Besu (Besu) ledger, which is maintained and consented by the participating TSPs. Therefore, the only trusted version of the TOnNe is the copy that resides within the world state of the blockchain, meaning no modifications can happen without consensus of all TSPs.

A Merkle tree, also known as a hash tree, is a data structure to efficiently verify the integrity of large amounts of data Merkle [37]. Figure 7 illustrates a Merkle tree. Merkle trees work by breaking down a large amount of data into smaller blocks, and then creating a hash of each block using a cryptographic hash function. These hashes are then organized into a binary tree structure, with each leaf node representing a data block and each non-leaf node representing the hash of its child nodes.

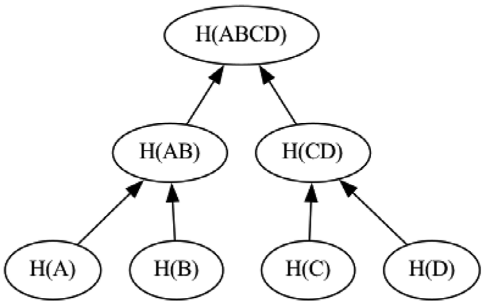

FIGURE 7 A Merkle tree.

The process starts with creating leaf nodes for each data block and then combining pairs of leaf nodes to create their parent nodes. The process is repeated until a single node is left, which is the Merkle root. This root is a hash of all the data blocks and serves as a digital fingerprint of the entire data set. To verify the integrity of a data set, one only needs to compare

TABLE 3 The smart contracts used by the STUB platform.

`STUB` (singleton)	This is the main contract that serves as the central entry point for the system. It acts as a registry and controller for other smart contracts within the architecture, managing the relationships between tickets, TOnNes, and organizations. There are a variety of methods to allow STUB administrator addresses to add new administrators or create new objects.
`TransportNetworkOntology`	This contract represents the transport network's overall structure, including stops, services, and planned stops. It is responsible for maintaining relationships between these elements, as well as the associated tickets and organizations.
`TOnNeInterface`	This contract represents an interface for objects part of the TOnNe to ensure the overriding of two functions: **`getDetails`**, which provides a JSON (JSON) representation of the details of the implementing class; and **`getHashDigest`** which returns the top-level hash digest of each implementing class. This enables the off-chain version to check the Merkle root of the instances (explained in Section 5.3), and determine if any changes have occurred.
`Organisation`	This abstract contract represents the various entities involved in the transport ticketing ecosystem, such as TSPs, government agencies, and other stakeholders.
`TransportServiceProvider`	A concrete implementation of the Organisation contract, representing individual TSPs within the system.
`Service`	A contract representing the various transportation services offered by the TSPs, such as bus routes, train lines, or ferry services. This can be built upon further by subclasses.
`PlannedStop`	This contract represents planned stops within the transport network, defining the schedule and location of stops for each service. This can be built upon further by subclasses.
`Stop`	A contract representing physical stops within the transport network, such as bus stations, train stations, or ferry terminals. This can be built upon further by subclasses.
`Ticket`	This contract represents individual tickets issued within the STUB system. It maintains relationships with the associated services and planned stops, ensuring secure and accurate ticket validation.

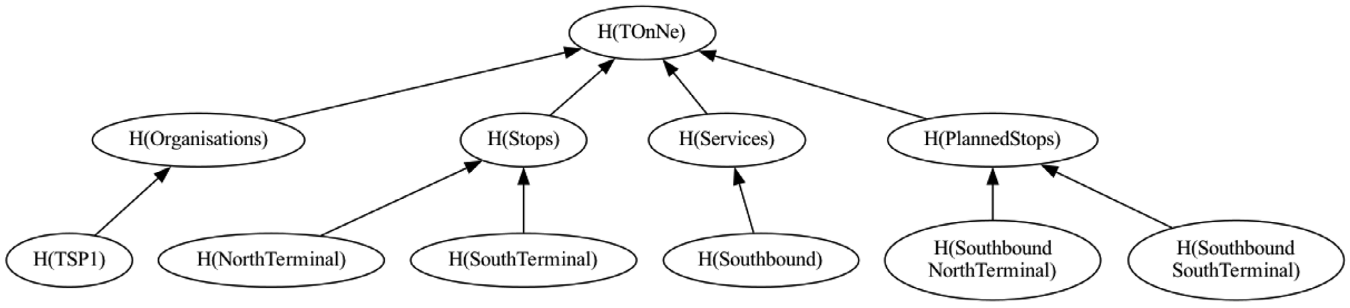

FIGURE 8 The Merkle proof tree for the TOnNe.

the Merkle root of the data set with a previously stored value. If the values match, the data has not been tampered with. However, if the values do not match, the data has been modified or tampered with. Additionally, Merkle Trees also allow for efficient verification of partial data sets by providing a mechanism for "Merkle proofs" which are small pieces of data that can be used to prove that a specific data block is included in the data set without revealing the entire data set.

We use the concept of Merkle proofs to validate the TOnNe, illustrated in Figure 8. Here, the TOnNe follows a hierarchical object-oriented programming (OOP) structure, with a superclass entity at the root (the `TransportNetworkGraph`), and various subclasses and member classes. The single entity for the TOnNe that will persist within the blockchain also stores a reference to each child entity within a mapping. To check whether the on-chain and off-chain TOnNes are identical, the hash digest of the root entity is computed from the off-chain TOnNe. If the hash digest matches the on-chain hash digest, we can immediately verify that the on-chain and off-chain TOnNes are identical, and that no updates to the off-chain version are required. However, if the has digest is different, then there has been a change to the on-chain version, and it must be determined where.

A crude way to do this would be to parse the entire structure again, but this is computationally infeasible, especially as the size of the TOnNe begins to increase. As such, the next layer of entities are checked. For example, there exists a mapping structure that contains all of the planned stop entities. By computing the hash digest of all of the off-chain planned stop entities and comparing it to the on-chain hash digest, we reveal whether the stopping points have changed. If so, we can continue to move down in levels of granularity (sorted by TSP, region etc.) until we reach the specific changes, without having to perform an entire search.

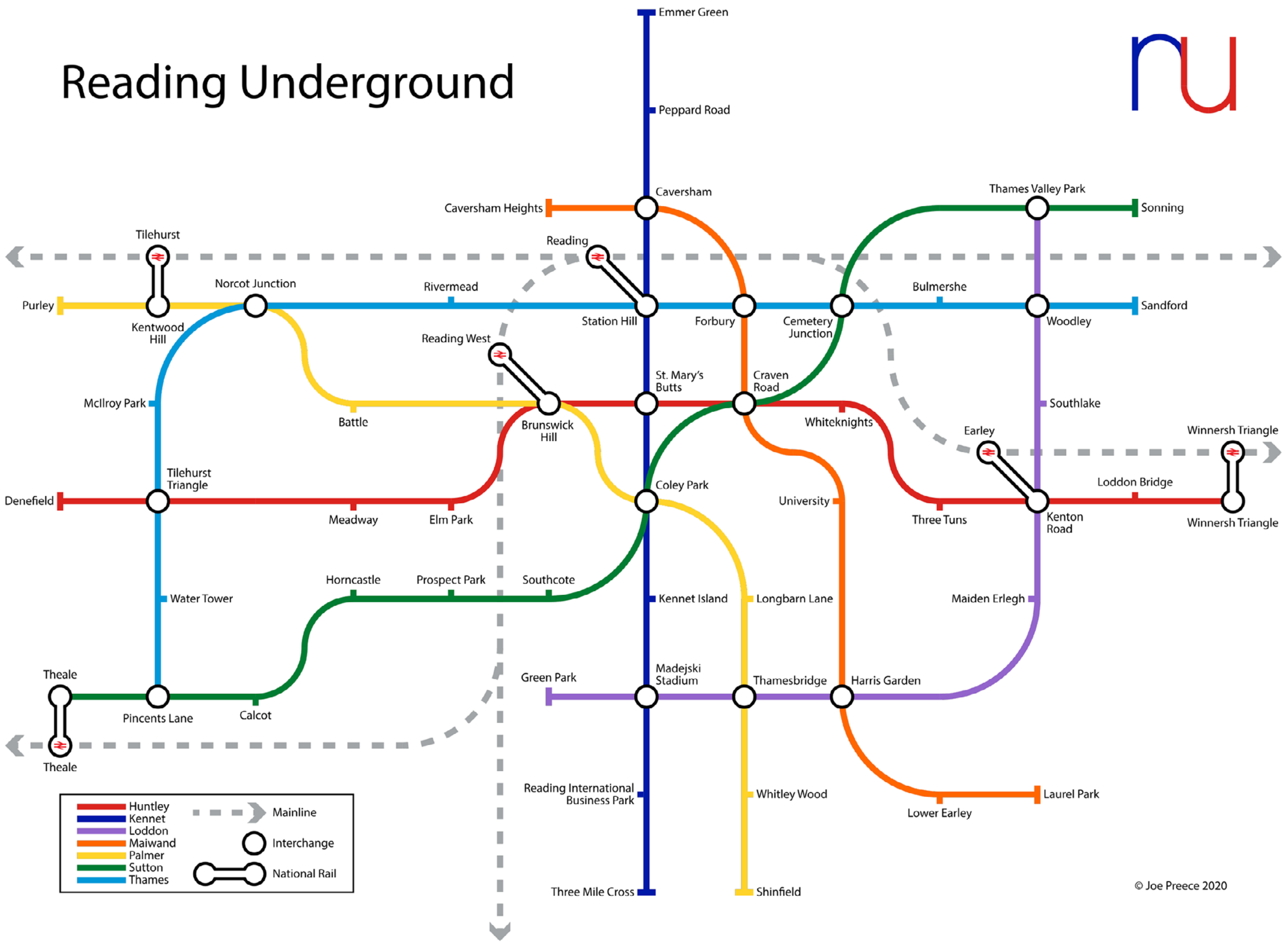

FIGURE 9 The fictional Reading Underground network used to test the capabilities of STUB.

5.4 | Proof of journey

Returning to the example from Section 3, assume that Bob decides to use yellow, for which his ticket is not valid. The yellow validator (validator) would take the ticket information, ensure they have the latest version of the TOnNe by checking the Merkle root values of the on-chain version and the off-chain version, and validate the ticket. Because the ticket is invalid, yellow can reject the ticket and issue Bob with a fine, or ask him to leave the transport at the next available convenience. However, tickets (tickets) which are valid are marked as such, updating the ticket on the ledger with identifying information about who and where the ticket was validated. This data is propagated via the ledger, allowing any future validators to track the journey of a customer via their shared ticketing data on the ledger.

5.5 | Toy model proof-of-concept

The Reading Underground, a toy transport network illustrated in Figure 9, served as a suitable test environment for the STUB. With seven TSPs offering 14 services across 52 stops, it provided ample size to indicate the durability of the STUB design. Utilizing the Truffle Suite as a testing environment and ingesting the provided data, we assessed the STUB[’s] ability to read the latest changes from the blockchain and update its own state using Merkle proofs and the architectures described in Section 5.

5.5.1 | Testing environments

The testing suite incorporates a multi-faceted approach, merging Stardog and Hyperledger Besu for ontology and smart contract management respectively. We deployed a Stardog cloud server as the centre of our ontology management strategy. This facilitated efficient data storage, querying, and inference, enabling seamless integration of ontological data within the platform’s operations.

For the smart contract functionality and testing within the Hyperledger Besu network, we integrated the Truffle testing suite. This suite provided precomputed addresses, enabling the

TABLE 4 The STUB unit tests.

	Test	Description
1	Create a STUB singleton.	Create a STUB smart contract via the smart contract constructor, owned by a STUB administrator address.
2	Create a tn.	Use a STUB administrator account create a new instance of `TransportNetworkOntology`, and then add it to the `STUB` instance using the `addTransportNetwork()` function. Only the `STUB` administrator should be able to achieved this.
3a	Create the `TransportServiceProvider` smart contract instances.	For each of the seven TSPs, create a smart contract instance with a separate administrator address for each instance. A STUB administrator then adds each instance to the `TransportNetworkOntology` instance via the `addOrganisation()` function.
3b	Ensure `TransportServiceProvider` instances are added to the ontology.	We call the `getAllDetails()` function to return the top-level hash digest of the entire `TransportNetworkOntology`, and compare it with the top-level hash digest from the Stardog ontology. As the values will be different, we then look through the next layer of hash digests in the Merkle tree. The only value of difference will be that of the TSPs, which can then be updated.
4a	Create the `Stop` instances.	For each of the stops, create a smart contract instance with the relevant TSP administrator. The same TSP administrator then adds each instance to the `TransportNetworkOntology` instance via the `addStop()` function.
4b	Ensure `Stop` instances are added to the ontology.	Same process as 3b.
5a	Create the `Service` instances.	For each of the services, create a smart contract instance with the relevant TSP administrator. The same TSP administrator then adds each instance to the `TransportNetworkOntology` instance via the `addService()` function.
5b	Ensure `Service` instances are added to the ontology.	Same process as 3b.
6a	Create the `PlannedStop` instances.	For each of the planned stops, create a smart contract instance with the relevant TSP administrator. The same TSP administrator then adds each instance to the `TransportNetworkOntology` instance via the `addPlannedService()` function.
6b	Ensure `PlannedStop` instances are added to the ontology.	Same process as 3b.
7	Change a variable.	Use an administrator account to change the details of an instance of a class, and see this updated on the off-chain version via the required functions.
8a	Issue a ticket	Use a TSP administrator to issue a ticket to a customer address, between two stops.
8b	Validate a ticket	Direct from the off-chain ontology, check the ticket details and validate the ticket by determining the trip is on the correct line.

invocation of smart contracts within a suitable test environment. This ensured that smart contracts underwent rigorous testing, guaranteeing their reliability and correctness within the Hyperledger Besu blockchain network.

To enhance the interoperability and flexibility of these technologies, we developed specific JavaScript files. The `Stardog Handler` script enabled data transactions to be executed directly from within JavaScript, capitalizing on Truffle's native language capabilities. Additionally, the `Middleware` script facilitated the conversion of Reading Underground information from comma separated value (CSV) files into Turtle format, which could then be transmitted to the Stardog server.

5.5.2 | Testing suite

In our pursuit to evaluate the capabilities of the Merkle proof concept, we conducted a series of comprehensive tests. Table 4 describes each test in detail. Using the Reading Underground as a transport network through smart contracts, we consistently monitored the state of the transport network across both the off-chain and on-chain versions. This systematic examination enabled us to effectively ascertain the validity and efficacy of the Merkle proof concept within the context of smart contract-based transport networks. Additionally, we tested lightweight approaches for validating tickets within the off-chain version, thus broadening our assessment and understanding of the Merkle proof concept's practical application within the domain of ticket validation and transport network management.

6 | ECONOMIC FEASIBILITY

Precise cost estimates are rarely accurate on projects of this scale, even when the scope has been precisely defined and the project is in the process of implementation, as such we will attempt only to illustrate very approximate costings and thus the economic feasibility of the solution.

First we must consider the hosting cost of the Besu part of the system. Assuming for the purposes of a coarse estimate that the system will be hosted in the cloud, in particular using Amazon Web Services (AWS) (a not unreasonable assumption, given the low cost of the service), we can calculate what hardware would be needed and thus what the cost would be. Using the work in Gonçalves et al. [38] it is possible to ascertain that a medium AWS instance, available for approximately $20 per month, would be adequate for the blockchain hosting.

Moving on the ontology hosting, taking Stardog as an example triple store and assuming we propose to host it in the cloud, then we can use the Stardog capacity planning guide[2] to ascertain how big of a server we would need. If we assume that each of the 4,294,504 timing points on the UK rail network needs to be described and that we use four triples for each then we need to 17,178,016 triples. Whilst this is a limited estimation (some will need more than four triples, some will not be needed at all), the minimum number of triples Stardog considers for planning is 100,000, 000. Such a server could be had for under $300 per month from AWS. Whilst these numbers are approximate, they could increase by several orders of magnitude and still be entirely economically feasible for a system taking in £2.6 billion [39] in revenue.

7 | CONCLUSION

This paper introduced STUB 2.0, an improvement over its predecessor that leverages ontochains, a hybrid data structure combining the strengths of blockchains and ontologies. Through the integration of these two technologies, STUB aims to address the limitations of traditional ticketing systems by providing a secure, transparent, and flexible framework for ticket issuance, validation, and management.

We have outlined the key components and workflow of the STUB system, demonstrating how TOnNes can effectively model complex relationships within transportation systems, while blockchain technologies ensure secure, distributed, and tamper-proof transactions that relate to the TOnNe and tickets. Furthermore, we discussed the implementation of Merkle proofs for efficient and secure validation of ontological data within STUB.

When compared to other existing or proposed smart transport ticketing systems, STUB demonstrates several advantages. By utilizing ontochains, STUB facilitates improved interoperability between different transportation modes and TSPs, enabling a more seamless and user-friendly experience for passengers. Furthermore, the incorporation of blockchain technology ensures a high level of security and transparency, mitigating the risks of data tampering.

However, the implementation of STUB in real-world scenarios may encounter potential challenges and limitations. Scalability remains a concern, as the system would need to handle a large number of transactions and maintain performance levels in the face of growing demand. Techniques for improving blockchain scalability, such as sharding or off-chain solutions, should be considered in future iterations of STUB.

[2] https://docs.stardog.com/operating-stardog/server-administration/capacity-planning

Data privacy is another important aspect to consider, as the transparent nature of blockchain could raise concerns regarding the protection of sensitive user information. Employing privacy-preserving techniques, such as zero-knowledge proofs or confidential transactions, could help address these concerns while maintaining the benefits of a transparent and secure system.

Integrating STUB with existing transportation infrastructure may also present challenges, as it requires collaboration among various stakeholders, including TSPs, government agencies, and payment processors. Developing standardized interfaces and protocols for data exchange could facilitate smoother integration and encourage widespread adoption of STUB.

Thus far, STUB has only been tested on toy models, that are removed from the nuances of real-world systems. In the future, we intend to explore the applicability of the mechanisms of STUB with real world data. This could better be facilitated with an improved version of the TOnNe that utilizes the preexisting structures and standards of the GTFS, widely used by TSPs, enabling easier integration with data streams and existing platforms.

In conclusion, STUB represents an innovative step towards the next generation of smart transport ticketing systems. By harnessing the power of ontochains, STUB provides a robust and adaptable framework that has the potential to transform the way we access and utilize transportation services in the future.

NOMENCLATURE

API	application programming interface
AWS	Amazon Web Services
CSV	comma separated value
DKG	distributed knowledge graph
EU	European Union
GTFS	general transit feed specification
IMS	Integrated mobility system
IP4	Innovation Programme
JSON	JavaScript Object Notation
NGI	Next Generation Internet
OC	ONTOCHAIN
OOP	object-oriented programming
PoC	proof-of-concept
PoO	proof-of-ownership
PoV	proof-of-validity
RFID	radio frequency identification
STUB	system for ticketing ubiquity with blockchains
TOnNe	Transport network ontology
TSP	transport service provider
UK	United Kingdom
UML	unified modelling language

AUTHOR CONTRIBUTIONS

Dr Joe Preece: conceptualisation; methodology; visualisations; writing—draft; software. **Dr Christopher Morris**: software; analysis; writing—draft. **Dr John Easton**: writing—reviewing and editing.

ACKNOWLEDGMENTS

During the preparation of this work the authors used Bearly AI in order to draft and improve the language used within the paper. At no point was Generative AI used to conduct automated research or provide insights; all original research was undertaken by the authors. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication. This project received funding from the TruBlo project the third Open Call (subgrant agreement), under European Union's Horizon 2020 Research and Innovation program under the Grant Agreement Number 957228.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Joseph D. Preece https://orcid.org/0000-0002-1854-3578

REFERENCES

1. Preece, J.D., Easton, J.M.: Blockchain technology as a mechanism for digital railway ticketing. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 3599–3606. IEEE, Piscataway, NJ (2019)
2. Preece, J.D.: Ticket to Ride: An Investigation into the Use of Blockchain Technology in the Rail Industry. University of Birmingham, Birmingham (2020)
3. Namiot, D., Pokusaev, O., Kupriyanovsky, V., Akimov, A.: Blockchain applications for transport industry. Inte. J. Open Inf. Technol. 5(12), 130–134 (2017)
4. Callefi, M.H.B.M., Tavares, T.M., Ganga, G.M.D., Godinho Filho, M.: Blockchain-enabled capabilities in transport operations: an overview of the literature. Indep. J. Manage. Prod. 12(9), 728–740 (2021)
5. Eremina, L., Mamoiko, A., Bingzhang, L.: Use of blockchain technology in planning and management of transport systems. E3S Web Confe. 157, 04014 (2020)
6. Astarita, V., Giofrè, V.P., Mirabelli, G., Solina, V.: A review of blockchain-based systems in transportation. Information 11(1), 21 (2019)
7. Lima, M.B., Banakar, R.M.: Fifa ticketing system using blockchain hyperledger sawtooth platform. Int. Jo. Eng. Adv. Technol. 9(4), 1031–3035 (2020). https://doi.org/10.35940/ijeat.d7789.049420
8. Ferrick, B.: The times they are a-changin': incorporating blockchain networks into the event ticket industry. SSRN Electron. J. (2019). https://doi.org/10.2139/ssrn.3318703
9. Cha, S.C., Peng, W.C., Hsu, T.Y., Chang, C.L., Li, S.W.: A blockchain-based privacy preserving ticketing service. In: 2018 IEEE 7th Global Conference on Consumer Electronics (GCCE), pp. 585–587. IEEE, Piscataway, NJ (2018)
10. Preece, J., Easton, J.: A review of prospective applications of blockchain technology in the railway industry. Int. J. Railw. Technol. (2018)
11. Preece, J., Morris, C., Easton, J.: Towards STUB 2.0: Using graph-based world states in hyperledger Besu to facilitate distributed transport ticketing. In: 2022 IEEE International Conference on Big Data (Big Data), pp. 3838–3844. IEEE, Piscataway, NJ (2022)
12. Nguyen, T.H., Partala, J., Pirttikangas, S.: Blockchain-based mobility-as-a-service. In: 2019 28th International Conference on Computer Communication and Networks (ICCCN), pp. 1–6. IEEE, Piscataway, NJ (2019)
13. Gruber, T.R.: A translation approach to portable ontology specifications. Knowl. Acquis. 5(2), 199–220 (1993)
14. Köpf, H., et al.: Integrail–publishable final activity report. Intelligent Integration of Railway Systems no. FP6–012526, (2010)
15. Verstichel, S., Ongenae, F., Loeve, L., Vermeulen, F., Dings, P., Dhoedt, B., Dhaene, T., De Turck, F.: Efficient data integration in the railway domain through an ontology-based methodology. Transp. Res. Part C: Emerg. Technolo. 19(4), 617–643 (2011)
16. Tutcher, J., Easton, J.M., Roberts, C.: Enabling data integration in the rail industry using RDF and owl: the racoon ontology. ASCE-ASME J. Risk Uncertainty Eng. Syst., Part A: Civ. Eng. 3(2), 4015001 (2017)
17. Smith, B., Kumar, A.: Controlled vocabularies in bioinformatics: a case study in the gene ontology. Drug Discovery Today: BIOSILICO 2(6), 246–252 (2004)
18. Sharma, N.: The origin of data information knowledge wisdom (DIKW) hierarchy. Preuzeto 25, 2021 (2008)
19. Papaioannou, T.G., Kochovski, P., Shkembi, K., Barelle, C., Simonet-Boulogne, A., Ciaramella, M., Ciaramella, A., Stankovski, V.: A blockchain-based, semantically-enriched software framework for trustworthy decentralized applications.
20. Shkembi, K., Kochovski, P., Papaioannou, T.G., Simonet-Boulogne, A., Ciaramella, M., Barelle, C., Stankovski, V.: The semantic web and blockchain at a meeting point.
21. Papaioannou, T., Stamoulis, G.D.: A business model for multi-tiered decentralized software frameworks: The case of ontochain.
22. Norta, A., Kormiltsyn, A., Udokwu, C., Dwivedi, V., Aroh, S., Nikolajev, I.: A blockchain implementation for configurable multi-factor challenge-set self-sovereign identity authentication. In: 2022 IEEE International Conference on Blockchain (Blockchain), pp. 455–461. IEEE, Piscataway, NJ (2022)
23. García, R., Cediel, A., Teixidó, M., Gil, R.: Copyrightly: Blockchain and semantic web for decentralised copyright management. In: Proceedings of the 18th International Conference on Economics of Grids, Clouds, Systems, and Services, GECON 2021, pp. 199–206. Springer, Cham (2021)
24. Bella, G., Cantone, D., Longo, C., Nicolosi Asmundo, M., Santamaria, D.F.: Blockchains through ontologies: the case study of the ethereum ERC721 standard in oasis. In: Intelligent Distributed Computing XIV, pp. 249–259. Springer, Cham (2022)
25. Kalafatelis, A., Panagos, K., Giannopoulos, A.E., Spantideas, S.T., Kapsalis, N.C., Touloupou, M., Kapassa, E., Katelaris, L., Christodoulou, P., Christodoulou, K., et al.: Island: an interlinked semantically-enriched blockchain data framework. In: Proceedings of the 18th International Conference on Economics of Grids, Clouds, Systems, and Services, GECON 2021, pp. 207–214. Springer, Cham (2021)
26. Bella, G., Cantone, D., Longo, C., Nicolosi Asmundo, M., Santamaria, D.F.: Semantic representation as a key enabler for blockchain-based commerce. In: Proceedings of the 18th International Conference on Economics of Grids, Clouds, Systems, and Services, GECON 2021, pp. 191–198. Springer, Cham (2021)
27. Arshad, J., Azad, M.A., Prince, A., Ali, J., Papaioannou, T.G.: Reputable–a decentralized reputation system for blockchain-based ecosystems. IEEE Access 10, 79948–79961 (2022)
28. Kochovski, P., Stankovski, V., Gec, S., Faticanti, F., Savi, M., Siracusa, D., Kum, S.: Smart contracts for service-level agreements in edge-to-cloud computing. J. Grid Comput. 18, 673–690 (2020)
29. Sopek, M., Tomaszuk, D., Głąb, S., Turoboś, F., Zieliński, I., Kuziński, D., Olejnik, R., Łuniewski, P., Grądzki, P.: Technological foundations of ontological ecosystems on the 3rd generation blockchains. IEEE Access 10, 12487–12502 (2022)
30. Blythe, P.: Integrated ticketing smart cards in transport. In: IEE Seminar on Using ITS in Public Transport and in Emergency Services (Ref. No. 1998/524), pp. 1–20. IET, London (1998)
31. Blythe, P.T.: Improving public transport ticketing through smart cards. In: Proceedings of the Institution of Civil Engineers - Municipal Engineer, vol. 157, pp. 47–54. Thomas Telford Ltd., London (2004)
32. Turner, M., Wilson, R.: Smart and integrated ticketing in the UK: piecing together the jigsaw. Comput. Law Secur. Rev. 26(2), 170–177 (2010)
33. Gnoni, M.G., Rollo, A., Tundo, P.: A smart model for urban ticketing based on RFID applications. In: 2009 IEEE International Conference on

Industrial Engineering and Engineering Management, pp. 2353–2357. IEEE, Piscataway, NJ (2009)
34. Finger, M., MONTERO-PASCUAL, J.J., Serafimova, T.: Towards EU-wide multimodal ticketing and payment systems (2019)
35. Scrocca, M., Comerio, M., Carenini, A., Celino, I.: Turning transport data to comply with eu standards while enabling a multimodal transport knowledge graph. In: Proceedings of the 19th International Semantic Web Conference–ISWC 2020, pp. 411–429. Springer, Cham (2020)
36. McHugh, B.: Pioneering open data standards: The GTFS story. Beyond Transparency: Open Data and the Future of Civic Innovation, pp. 125–135. America Press, San Francisco, CA (2013)
37. Merkle, R.C.: A digital signature based on a conventional encryption function. In: Proceedings of the Advances in Cryptology–CRYPTO'87, pp. 369–378. Springer, Berlin, Heidelberg (1988)
38. Gonçalves, I., González Silva, P.H., Mendonça, D.: Estimating transaction cost for cloud-based private ethereum blockchains. (2021)
39. Office of Rail and Road. Passenger rail usage. https://dataportal.orr.gov.uk/statistics/usage/passenger-rail-usage/

How to cite this article: Preece, J.D., Morris, C.R., Easton, J.M.: Leveraging ontochains for distributed public transit ticketing: An investigation with the system for ticketing ubiquity with blockchains. IET Blockchain 1–14 (2024). https://doi.org/10.1049/blc2.12071