LEDGER
ledgerjournal.org

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

RESEARCH ARTICLE

Are Smart Contracts and Blockchains Suitable for Decentralized Railway Control?

Michael Kuperberg,[*†] Daniel Kindler,[‡] Sabina Jeschke[§]

Abstract. Conventional railway operations employ specialized software and hardware to ensure safe and secure train operations. Track occupation and signaling are governed by central control offices, while trains (and their drivers) receive instructions. To make this setup more dynamic, the train operations can be decentralized by enabling the trains to find routes and make decisions which are safeguarded and protocolled in an auditable manner. In this paper, we present the case study findings of a first-of-its-kind blockchain-based prototype implementation for railway control, based on decentralization but also ensuring that the overall system state remains conflict-free and safe. We also show how a blockchain-based approach simplifies usage billing and enables a train-to-train/machine-to-machine economy. Finally, first ideas addressing the use of blockchain technology as a life-cycle approach for condition-based monitoring and predictive maintenance in train operations are outlined.

1. Introduction

Distributed ledger technologies (DLTs) and blockchains are well-suited for trust networks without a single, all-powerful central authority. When it comes to auditability and tamper resistance, the underlying cryptography promises an integrated approach that includes traceability, fault-tolerant consensus mechanisms, and verifiable timestamped signatures. Additional functionality is provided by chain-stored executable rules, known as smart contracts, which can be executed by network participants. The results of smart contract execution are again stored on-chain, providing a seamless framework for distributed applications.

When collaborative decision-making of multiple parties is used to form consensus, decentralization is achieved: state changes are agreed upon directly by the involved and affected parties. The objective of decentralization is to minimize the use of monopolizing intermediaries and to avoid undesired asymmetric power settings. Decentralization mechanisms provide competing (or mutually non-trusting) players with a cooperation platform where trust is established through auditability and transparent consensus.

Legacy IT solutions for resource control and scheduling often suffer from a lack of transparency and trust, and they are based on fragmented architectures with lacking access to data. Additionally, these solutions often do not provide built-in facilities for metering and billing, and they are built for human operators rather than for self-aware (or automated/autonomous) devices and vehicles.

[*] 0xB543aA9AE3f8e1369D150d4Df660cdbA6840097E
[†] Michael Kuperberg (michael.kuperberg@deutschebahn.com) is the Chief Blockchain Architect of DB Systel GmbH, the IT provider of Deutsche Bahn AG.
[‡] Daniel Kindler (daniel.kindler@deutschebahn.com) is the Managing Partner Blockchain & DLT Solutions – DB Group Business Segment Infrastructure of DB Systel GmbH.
[§] Sabina Jeschke (sabina.s.jeschke@deutschebahn.com) is a Member of the Management Board Digitalization & Technology (T) of Deutsche Bahn AG.

DLTs and blockchains are candidate technologies to remedy some of these shortcomings. The business domain of railway operations constitutes an example of resource control and scheduling where (for historical reasons) the centralized solutions are often based on multiple information sources, as the overall complexity could not be handled in a single data source. Evolving the solution design for this domain must account for the changing organizational circumstances: for example, many countries are shifting away from a vertically-integrated monopolies and towards a close-knit network of independent companies, including infrastructure providers, mutually-competing train operators, and independent oversight agencies (governmental and international).

However, the suitability of DLTs for reliably *replacing* traditional, centralized top-down rail control systems with a large number of participants is still an open research topic. This topic is relevant to those DLT researchers who are interested in the specific real-life challenges of decentralized control systems, where individual goals and systemwide goals may conflict. Specifically for the safety-sensitive domain of railway control, a trusted distributed datastore with consensus-based (decentralized) process execution has a significant potential for improving flexibility and cost efficiency when compared to traditional implementations based on point-to-point integration, assuming that the conflicting interests are properly balanced. For the railway domain, integrating trains and infrastructure as first-class participants (without human intermediaries) can help streamline the operations. While a lot of research for DLTs is focused on business-to-business (B2B) scenarios (such as supply chains, elimination of middlemen, micropayments, etc.), machine-to-machine (M2M) scenarios are mostly focused on "Internet of Things" (IoT) and edge device economy such as monetization of sensor data. Application of DLTs to multi-agent systems has been studied as well, but no research has been undertaken to integrate safety and security aspects (such as interlocking in train operations).[1-3]

The relevance of decentralized traffic control reaches beyond railways: it can provide benefits to other transportation modes (such as unmanned vehicles for goods and passengers), including water transportation and warehouse logistics. Application of DLTs to platooning and car economy (beyond simple billing of energy and services) is emerging,[4] and the long-term vision of decentralized applications (DApps) will benefit from previous research in multi-agent systems and autonomous/self-learning systems.

Before a fully-fledged decentralized implementation of a given control system is started, the overall viability of decentralization for a specific scenario needs to be studied and prototyped. Some limitations of the DLTs are well-known (such as throughput, constraints from the CAP theorem, data growth, and the need for oracles), while others are only uncovered by implementing a use case as close as possible to the real-world situation.

The contribution of this paper is a systematic case study of the applicability of decentralization to a real-life control system, exemplified by the railway domain and backed by a prototype implementation. We derive research questions and a research hypothesis, based on high-level requirements. To the best of our knowledge, this paper provides the very first analysis of such a decentralized control system.

The remainder of this paper is structured as follows: Section 2 describes the context and the foundations of rail control systems. Section 3 presents our work goals and research setting, leading to the requirements and the problem statement in Section 4. Section 5 discusses our research hypothesis and how it can be validated. Section 6 analyzes related work and how it compares to our results.

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

Section 7 presents the architecture of the solution and explains the choice of the used technologies (including the use of the Ethereum blockchain stack). Section 8 describes some important implementation aspects and the “lessons learned” that we gathered during the proof of concept. Section 9 concludes and provides an outlook, together with the planned next steps.

2. Foundations

Unlike car traffic and streetcars (trams), most mainline railway operations have strict access control measures to prevent accidents such as train collisions. To safeguard operations, railways utilize technical frameworks to constantly enforce strict safety procedures. These frameworks are designed and implemented to withstand an operator’s failure or even an operator’s death.

While the scope of the frameworks differs between countries, the state-of-the-art implementations such as ETCS include the constant upkeep of “safe blocks” (to prevent collisions in case one of the trains comes to an unexpected stop or even derails),[5] emergency braking if a red signal is passed (or if the human operator does not react within a specified time), detection of train decomposition, and variable speed control. Trainside and lineside IT components work together to achieve these goals. The resulting functionality is often called Automatic Train Protection (ATP), and it consists of several hardware and software parts. Figure 1 shows a high-level view of these core components.

Beyond the safety-guaranteeing frameworks, railway operation requires live dispatching: in addition to schedule-based passenger/freight trains, dispatching must accommodate ad-hoc traffic, deviations, construction-caused alterations, equipment failures, and so on. Despite advances in conventional and AI-supported decision making, a lot of this work is still performed by humans, *i.e.* experienced dispatchers. Dispatching and safety frameworks are usually partially decomposed for large railway networks: they are split into regions so that size and complexity are manageable—very similar to air traffic control operations. Over time, such operations have been electrified, electronic equipment has been introduced, and the newest equipment generation relies on digital, semi-detached infrastructure elements (switches, signals, occupancy sensors) as well as on in-cab signaling. [6]

Dispatching and safety frameworks are complex, heterogeneous and very costly (with an invest of between 100,000 and 300,000€ per km),[7] developed over many years and with lifecycles of several decades. Despite attempts at standardization, both interoperability and vendor lock-in pose an ongoing challenge. Additionally, such frameworks have historically been nation-specific; international standards such as the European Train Control System (ETCS) require substantial investment in hardware and software during a transition phase. Ultimately, this should help overcome the heterogeneous patchwork of country-specific standards in place throughout the EU: cross-border train operation often requires additional training, multi-system vehicles (at a higher cost), or changing trains/staff at the border.

Still, railway traffic management (which includes timetabling, capacity management and other concerns beyond dispatching and safety) is not part of ETCS; such functionality is being designed as part of the European Rail Traffic Management System (ERTMS),[8] which is the overall initiative that encompasses ETCS and the international wireless communications standard GSM-R. Infrastructure utilization is the key to lower operating costs, and customer satisfaction is strongly correlated with punctuality and density of service. Thus, some progress

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

has been made on improving track utilization (*e.g.* using flexible-length “moving blocks”),[9] in addition to improvements in safety. Despite these advances, the principles of railway operations remain largely the same: control center is the “authority” and the train is the “subject.” In most mainline railway control implementations, train-to-train coordination is only possible for the staff in the control center; train drivers act as human intermediaries that operate the train controls.

This top-down, hub-and-spoke pattern remains in place even for cutting-edge “autonomous train operation.”[10] Most mainstream upgrade programs also follow the “slow evolution” path, mandated by the backward compliance in large networks but also by the intrinsic interests of the manufacturers and investors. However, there are situations where the “authority-subject hub-and-spoke” pattern presents a bottleneck and becomes too restrictive, and where seamless train-to-infrastructure and train-to-train contracting would lead to improvements: trains could dynamically negotiate and “sell” a timeslot, automating redispatching in a rational, market-driven way. Potentially, passengers can request unscheduled stops (and bid/pool for them), and unpredicted construction or extension of maintenance shutdown periods could be propagated across the network. Additionally, trains could self-report operation-impacting defects (such as overheated axle bearings or derailments).

Another potential for improvements exists in the “back office” area: many national rail networks provide “open access” to competing passenger and freight railways, which pay regulated fees for infrastructure usage (tracks, stations, energy supply). Likewise, the “back office” sells ahead-of-time access rights since network access is strictly controlled to enable timetabled train operations to maintain their quality of service. Using a blockchain, *both* the sale/allocation of access rights/usage rights *and* the actual payment for them would happen transactionally and instantly, in one system rather than in several.

In the area of maintenance and servicing, the current state-of-the-art is to design maintenance windows, maintenance intervals, and maintenance plans in such a way that the safety of the system is ensured. The use of blockchain technology (coupled with the continuously evolving sensor technology of trains or infrastructure components) can be seen as the basis of modern maintenance, in which the individual components independently register their requirements. The rollout of such on-demand maintenance then includes for example also the automatic ordering of spare parts or the provision of special teams.

In Figure 1, we show the most essential parts of this railway “ecosystem.” Each shown part exists in several instances: there are sovereign rail networks both at the national and regional levels. In reality, there are additional layers (*e.g.* procurement, malus/bonus processing, HR, maintenance planning and management, construction, governmental oversight, insurance) which we do not detail here.

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

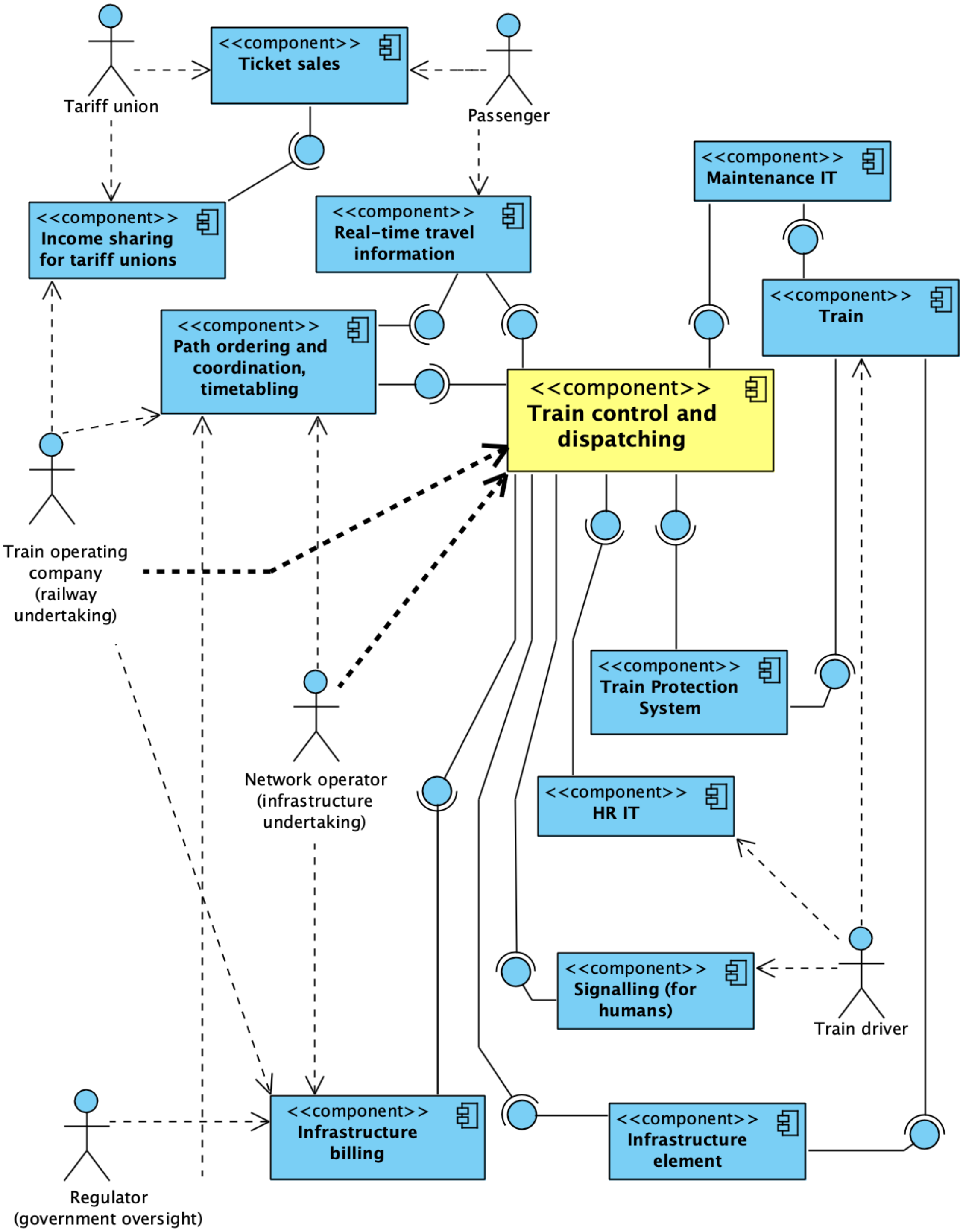
Tariff union
<<component>> Ticket sales
Passenger
<<component>> Income sharing for tariff unions
<<component>> Real-time travel information
<<component>> Maintenance IT
<<component>> Train
<<component>> Path ordering and coordination, timetabling
<<component>> Train control and dispatching
Train operating company (railway undertaking)
<<component>> Train Protection System
Network operator (infrastructure undertaking)
<<component>> HR IT
<<component>> Signalling (for humans)
Train driver
<<component>> Infrastructure billing
<<component>> Infrastructure element
Regulator (government oversight)

Fig. 1. Train control and dispatching IT with adjacent IT systems and actors.

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

3. Research Setting and Work Goals

As our long-term goal, we are working towards the *vision* of a modular, API-driven, multi-modal and multi-player "Transportation Operating System" with the central requirement to improve overall system performance in terms of capacity, efficiency, availability, and punctuality. One key property of the envisioned system is the systematic management of business processes and data, specifically in the way that all interactions (from ticket purchases to subcontractor payments) are handled on the basis of a consistent, trusted, resilient, and replicated "data base."

Thus, the core technological requirement for the underpinning of this vision is to provide a platform where competitors can collaborate, and where decisions are made by finding consensus among participants. This also means that the platform itself is run in a collaborative way, and no single company can monopolize the operations or the decision making. The design of the platform shall also enable government oversight and intervention where appropriate.

As our *mid-term work goal*, we want to build a core module out of the envisioned "Transportation Operating System": a novel control system for railway operations with "open-access" operations, *i.e.* for settings with multiple competing train operators. The control system shall include safety control logic (reusing existing infrastructure where appropriate) and dispatching logic (replacing legacy, centralized control centers). Train localization technology and train integrity checks (detection of decomposition) that are needed for safe operations are assumed to be available; there exist established technologies for this (such as trackside equipment, GNSS, Differential GPS, and axle counters).[11, 12]

This first core module shall be built and operated in the same collaborative manner as the overall platform and shall also lead to better metrics in the operations phase. We believe that building such a novel railway control system will also positively affect the technology support networks (especially maintenance and privately-owned infrastructure) and improve the efficiencies and the cash flow between the involved parties. To automate the cash flow, machine-to-machine (M2M) payments shall be supported, using IoT identities and wallets.

The potential for disruption includes opening the railway infrastructure to non-rail vehicles by including them into the safety frameworks of the railway control: for example, low-traffic track stretches in congested areas could be equipped with a road surface and emergency vehicles or construction equipment could co-use these stretches, provided that they have software and hardware to become part of the decentralized control system and of the "automatic vehicle protection" (which would be built similarly to the existing "automatic train protection" patterns). As the on-board IT of road vehicles becomes more powerful, cars are beginning to feature reliable duplex communication channels; this enables binding safety rules such as "vehicle cannot enter an occupied block." Also, maintenance staff could devise safety measures (such as speed reductions or temporary traffic halts) directly and "on the spot," visible to all vehicles.

Our make-or-buy analysis has shown that there is no product on the market which implements the business logic of the aforementioned control system, and it has led to the decision to design such a system in-house, using technology and components from the market. One of the core design decisions is the selection of the underlying runtime platform, which is also subject to the requirements in the following section.

Our *research objective* is to create a decentralized control system with the above properties and to enable trains (and train operators) as well as infrastructure elements to be first-level, *active*, and self-aware participants in railway control systems. Active participation by vehicles includes wayfinding (both ad-hoc and in-advance), booking, and actual usage of the infrastructure, as well as interaction with lineside equipment (*e.g.* setting a switch into the correct position) and with other vehicles. Such active participation builds on controlled access to a trustworthy, up-to-date view of the network use—past, current, and planned. As part of our objective, we want to establish an authoritative data repository that is audit-proof/tamper-proof (for example through write-once-read-many semantics, also known as WORM) and which relies on open-source, security-assessed, consumer, off-the-shelf (COTS) software components rather than on proprietary technology.

4. Requirements, Assumptions and Constraints

Safety-wise, the proposed architecture shall thus be at least on par with the current situation:

- The interlocking logic (algorithms+data) as mandated by the railway operating rules shall be kept (but they would run on a new platform, and may be reimplemented in another programming language): trains must not "overwrite" data that is the base for enforcing safety measures.
- The trackside infrastructure (*e.g.* switches, magnets/balises) that obeys the interlocking logic shall be kept (but a new generation of it might be capable of receiving direct payments, see Section 5).
- The trainside infrastructure which obeys the "orders" from the trackside infrastructure (*e.g.* speed controls, blocked section, etc.) shall remain in place, unless it is replaced by communication-based train control (CBTC).
- Trains, train-operating personnel, and train-operating companies shall gain insight into overall system state (current and future), enabling them to adjust their behavior and to react to emergencies (for example, a train can use "heartbeats" and other trains can subscribe to the "heartbeat feeds" of other trains in proximity, so that derailments or pending head-on collisions can be detected quickly).

As our approach is targeting efficiency improvements rather than safety improvements, it shall also support the "moving block" paradigm which increases train density and track usage.

To simplify safety assessments and certification, the system implementation shall be modularized, with well-defined interfaces and protocols between the functional building blocks–but still building on a single, uniform, and authoritative data layer. Certification is especially important in the context of hardware failures, and EU laws mandate the corresponding norms such as EN 50126[13,14] (IEC 62278[15-18]), DIN/EN 50128,[19-21] DIN/EN 50129,[22] DIN/EN 50159,[23] and DIN/EN 61508.[24]

With regards to our scope, we assume that the trainside safety aspects of the train operation are maintained (whether a train is operated automatically or by humans). For example, trainside emergency stop shall be auto-activated if a train enters an occupied block. However, the information such mechanisms rely upon is not provided by the centralized control center, but rather by the new control solution. Trainside safety measures (such as a forced emergency stop when trying to enter an occupied block) would be based on the state information (*e.g.* block occupancy) stored in the proposed, new system. Currently, such information is transmitted by trackside equipment/IT (*e.g.* LZB, magnets in PZB/Indusi) or by GSM-R (as in ETCS Level 3).

Therefore, our control system assumes that the physical reality and the IT representation are matched: the infrastructure elements and the trains have "IoT digital twins." It is imperative that a physical element and its "digital twin" are "mutually reliable" for both state representation and state changes. For example, the physical switch must be reliably "locked" except during state changes, and may not change its position without having been instructed by the digital twin to do so. If a switch position is modified by brute force (*e.g.* through sabotage), the malfunction must be detected (*e.g.* by an appropriate circuitry) and the disruption must be represented in the IoT digital twin, preventing an accident. We assume in the following that infrastructure elements and the synchronization between the IT representation and the physical entity is available and reliable, *i.e.* solved outside of our work.

5. Initial Research Hypothesis and How It Is Being Verified

We believe that modern, enterprise-grade implementation of distributed ledger technology (*e.g.* decentralized blockchains) is most suitable for the requirements in Sections 3 and 4, because blockchains/DLTs bring the desired platform, consensus functionality, and quality attributes out-of-the-box. We also believe that DLT/blockchains and software modularization (separation of concerns, *e.g.* through microservices) are complementing each other, and do not conflict. Therefore, the work presented in this paper seeks to support the following research hypothesis: *the complex resource, conditions, and constraints of railway control systems can be mapped to blockchain-level assets and smart contracts in such a way that decentralized bookings of resources can be performed while the safe and secure state of the entire system is maintained at any step*.

Note that proving this hypothesis does not impose a qualitative or a quantitative evaluation as a precondition. A quantitative comparison between the proposed approach and existing (centralized) approaches/implementations can only be done once the new system is implemented beyond a prototype. However, this paper's scope is to describe the prototype implementation; therefore, we do not yet have suitable data that could be used for such a quantitative comparison. Consequently, we leave both the comparison and the formulation of the underlying metrics to future work.

Technically, a DLT becomes the trusted "single truth" for both the current state of the networks (today, this is done in the control layer) and for the "future infrastructure reservations" which are the results of pre-scheduling and ad-hoc planning (today, this is done within separate systems, in the context of the traffic management layer). The ledger data is replicated across ledger nodes, providing fault tolerance and reliability. Furthermore, smart contracts serve as "gatekeepers" to the state changes (*e.g.* new reservations or cancellations) and the participating nodes cross-verify these changes, implementing a consensus mechanism that ensures consistency with specified rules across nodes.

It is a part of our approach that a train can pay the infrastructure usage directly to the infrastructure element (*e.g.* to a switch, or to a gated railway crossing). This means that the infrastructure element has a way to receive payments, and its owner can administrate those payments. Payments can also be bonuses (or fines), or "back office" tasks as in Figure 1. To enable such a M2M economy, we propose to use blockchain wallets (which are effectively PKI

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

keypairs, *cf.* Ethereum), where the balance of a wallet is stored on the blockchain ledger. In this paper, we do not discuss how the payment is integrated into the planning/management/control processes.

A major implementation aspect for testing this hypothesis is to restrict the write access to the data repository: it is to be protected by "gatekeepers," which ensure that only secure entries can be inserted (*e.g.* no two trains are located in the same block at the same time). The WORM semantics mean that data cannot be overwritten (not even by consensus). If an entry needs to be corrected, a new transaction ("compensation") is written and it shows the latest state. Some nodes may choose to store only a certain, regulation-imposed backlog of past data to keep the data amount manageable.

To test the hypothesis, we implemented a prototype of the control system and tested it on a real-world dispatcher training facility, *i.e.* under conditions which are closely paralleling those of full-scale operations. For the initial stage, we do not perform formal verification of the implementation. The implementation case study is described in the following section.

Some of the further functionality of the "Transportation Operating System" has already been showcased by us in previous works: *e.g.* blockchain-based station usage billing and blockchain-based revenue sharing.[25, 26] We have also studied how network partitioning can be detected and avoided at the consensus level in blockchain implementation. Throughout these modules, privacy, reliability, and performance (throughput, latency) remain an ongoing challenge.

6. Related Work and Literature Review

Decentralization in railway control has been addressed in multi-agent research, including comparisons of performance between centralized and decentralized scenarios.[27, 28] However, these approaches have neither used a tamper-proof, transparent ruleset (as the Ethereum smart contracts that we used), nor did they use a tamper-proof "full history" approach (as we do with the Ethereum-based distributed ledger/blockchain). Beyond the concepts, none of these approaches has been validated in a real-world training facility.

Outside the railway industry, autonomous vehicles are currently not able to pay for their infrastructure usage (such as tolled highways, bridges, or parking facilities) "on their own" (in an unsupervised manner). While there are proofs-of-concepts and interest groups with the focus of car-to-infrastructure payments, they are neither targeting ahead-of-time reservations nor do they cover the kind of limited-access, restricted-capacity infrastructure that is the cornerstone of railway operations. [29]

Our approach includes enabling trains to establish trade relationships with each other, *e.g.* to enable unsupervised monetary compensation from a delayed express train to a freight train when the latter yields its priority so that the express train can reduce its delay. The communication part of this vision can be compared to "car-to-car communication," [30] which relies on technologies such as 5G. In contrast to such work, our contribution focuses on the protocols and on the contents of such M2M communication; strictly speaking, the blockchain and its consensus introduces an intermediary layer.

Using blockchains (or distributed ledgers) for machine-to-machine payments has been studied and demonstrated on several occasions. This is encouraged by the native (crypto-) currency capabilities and token concepts of the underlying technologies, such as ERC20 in the

Ethereum ecosystem or NEP-5 in NEO.[31] Still, none of these approaches proposes or even implements the solution to the systemic constraints of mainline railway operations.

Non-DLT solutions for WORM data storage include dedicated hardware and software/cloud designs,[32, 33] modifications of existing file systems,[34] or data archiving.[35] Some database products support replication and "hot standby"/"cold standby" modes,[36] usually with a cluster-based distribution. In a similar way, Master Data Management systems are concerned with data replication and synchronization.[37] These solution types incur a vendor lock-in (due to proprietary software) and are subject to the "deleted by the master administrator" problem. Additionally, they do not scale across enterprises, across thousands of nodes, and do not provide the auditability of the "rule source code" and "rule execution" as is the case with blockchains/DLTs.

Our vision includes "sovereign" infrastructure elements such as switches which are adhering to the ledger rules and events (from smart contracts). There are different approaches to build railway IT infrastructure in a more networked/peer-to-peer way.[38] However, none of them covers M2M payments and train-to-train economy.

More modern paradigms to switch controls and automated dispatching are to be found in "turnkey" systems for urban/suburban networks,[39] often with CBTC (communication-based train control).[40] These networks have modularized management facilities which control both trains and infrastructure in an integrated way, including peak traffic management and ad-hoc addition of trains. However, so far, no such system has been applied as an "in-place upgrade" to a mainline network, as this would require large-scale adaptation of trains and infrastructure and also the (costly) coexistence of the "old" and "new" systems, as larger networks cannot be upgraded in one "big bang" step.

A certain overlap with our approach can be found in newer signaling and control systems such as ETCS,[5] where the train is more "aware" of its surroundings. However, even with ETCS, the physical train is obtaining certain constraints (for example, maximum speed) in a *passive* way, and for safety reasons the human train driver can only operate within those constraints.

From the economist's perspective, our proposal can be classified as a "decentralized transparent free market economy with strict rule enforcement," whereas the existing mainline approach is more of a "centralized market economy with central resource allocation in a non-discriminatory manner" (in Germany, the deregulated railway ecosystem includes a regulatory/oversight authority which is tasked with ensuring the open-access policy set forth by law). There is a substantial body of work investigating the (dis-)advantages of central coordination in comparison with decentralized, "peer-to-peer" trading. However, such research is rarely applicable to the allocation of scarce resources (as it is the case in mainline railways) in combination with safety constraints that are inherently complex and challenging to include in a market model/simulation.

7. Case Study Architecture and Employed Technologies

Conventional train control systems are usually centralized in two ways: logically and technically. Logically, there is just one business entity (the "infrastructure operator") running the control system, and it is the only party that has full access rights (including writing permissions); it may or may not allow read-only access for train operators at its own discretion. Technically, the control system is usually centralized because cost factors do not encourage a multi-node/multi-location setup. Additionally, a “hot standby” or even “active-active” setup means that data must be replicated successfully and completely *before* it can be considered as “written through”; this may increase latency and strain the data links between the locations.

Increasingly, infrastructure malfunctions are leading to compensation claims from the train operators, which themselves have to pay compensations for delays to customers. This forms a monetary driver for further fault tolerance in railways operations; additionally, the ongoing progress in hardware performance-to-cost ratio encourages the “design for failover” approach with additional hardware elements for fallback, even in the light of the additional implementation costs compared to the simple setup.

Therefore, our architecture (as shown in Figure 2) is primarily based on *technical decentralization*, *i.e.* on multiple nodes. The blockchain identity of the train is the actor which orders paths (*i.e.* performs reservations of an infrastructure element for a given timeframe); it interacts with the blockchain identity of the infrastructure element via the smart contract which

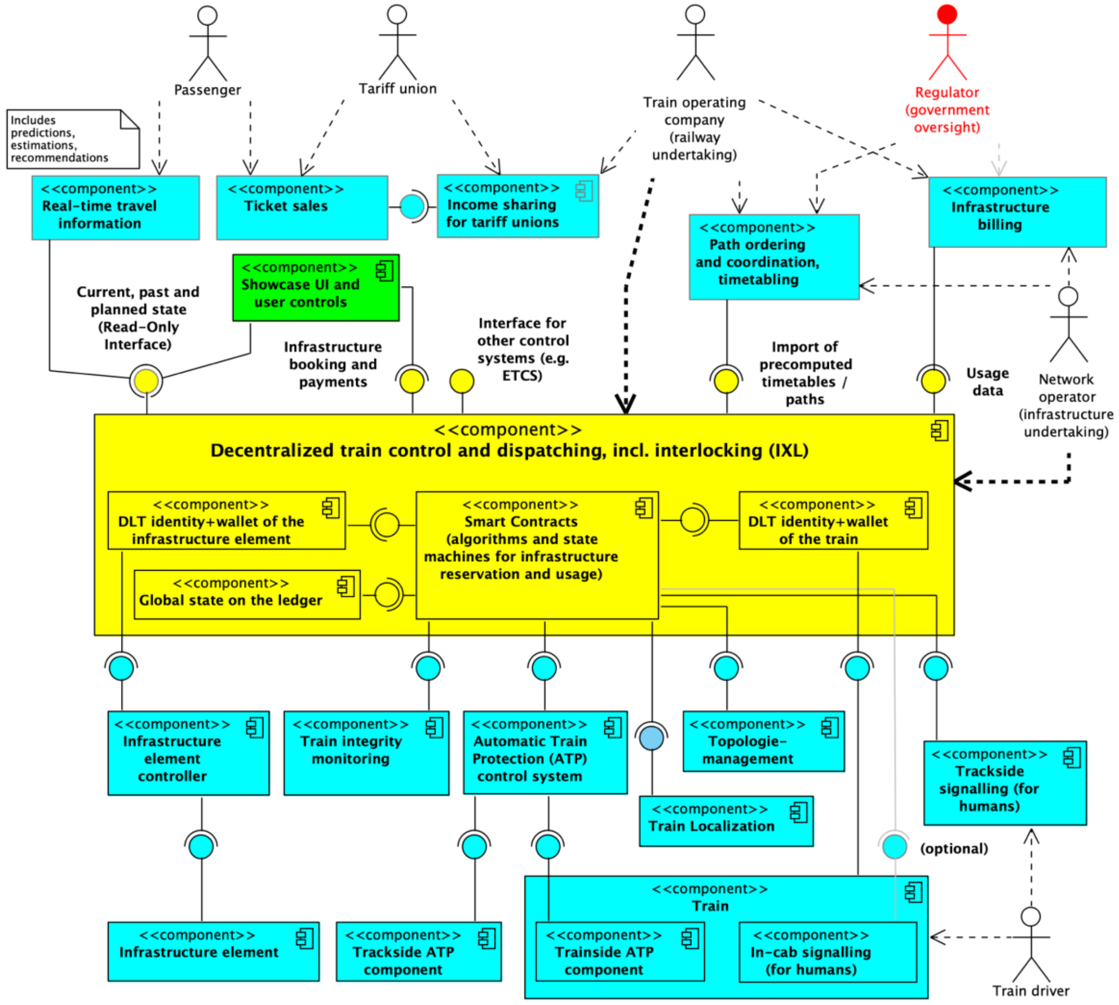
Passenger
Tariff union
Train operating company (railway undertaking)
Regulator (government oversight)
Includes predictions, estimations, recommendations
<<component>> Real-time travel information
<<component>> Ticket sales
<<component>> Income sharing for tariff unions
<<component>> Path ordering and coordination, timetabling
<<component>> Infrastructure billing
<<component>> Showcase UI and user controls
Current, past and planned state (Read-Only Interface)
Infrastructure booking and payments
Interface for other control systems (e.g. ETCS)
Import of precomputed timetables / paths
Usage data
Network operator (infrastructure undertaking)
<<component>> Decentralized train control and dispatching, incl. interlocking (IXL)
<<component>> DLT identity+wallet of the infrastructure element
<<component>> Smart Contracts (algorithms and state machines for infrastructure reservation and usage)
<<component>> DLT identity+wallet of the train
<<component>> Global state on the ledger
<<component>> Infrastructure element controller
<<component>> Train integrity monitoring
<<component>> Automatic Train Protection (ATP) control system
<<component>> Topologie-management
<<component>> Trackside signalling (for humans)
<<component>> Train Localization
(optional)
<<component>> Infrastructure element
<<component>> Trackside ATP component
<<component>> Train
<<component>> Trainside ATP component
<<component>> In-cab signalling (for humans)
Train driver

Fig. 2. Case study architecture (partially implemented by the prototype).

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

is the gatekeeper (ensuring consistency and a safe global state). The smart contract is the entity which changes the "should be" part of global state. When it comes to the "is" global state, the physical world is the "leading truth"; the state on the blockchain is mirroring the "physical truth." The train protection component does rely on the "is-state" but consults the required ("should be") future state as well.

When it comes to logical decentralization, the situation is more intricate. Logical decentralization means that the nodes belong to multiple parties. This immediately poses the questions of authority, agreements, responsibility, and liability. When it comes to safety and human life, strong authorities are the traditional choice. Implicitly, a single authority means a single (central) responsibility. In aviation, the pilot and the co-pilot are an example where strict rules of authority in a multi-party setup are used to prevent a stalemate (standoff), as there is no arbiter to act as an intermediary between the parties.

Logical decentralization is inherently more complex than logical centralization, as it needs to address the situation with failing/unreachable nodes, the meaning of dissenting minorities, party splitting and unstable behavior (*cf.* the "Byzantine generals" problem and the associated body of research). At the same time, large-scale networks with decentralized decision-making have appeared and maintained operation, *e.g.* the public Ethereum and Bitcoin blockchains. Such networks succeed in horizontal scaling, a working set of rules for consensus (which is a systematic decision-making using defined majority rules), and in fault tolerance. At the same time, achieving suitable quality of service and performance (latency, throughput, predictability) while maintaining scalability remains challenging in decentralized DLTs and blockchains.

Ultimately, our solution architecture is technically suitable for *both* logically-centralized *and* logically-decentralized setups: since we use a *private-consortial* (non-public) blockchain as the "engine" to run our algorithms and store our data, additional nodes can be added, and additional parties can be onboarded. Note that for our architecture, there is no need for separate "oracles" that supply externally-sourced information. For timer-triggered events (*e.g.* checking whether a reserved element can be released since the reservation has expired and the train has vacated the element), reliable solutions are rather straightforward to implement.

It is important to stress that reservations (and payments) are handled in a peer-to-peer fashion between trains and infrastructure elements (such as switches); the IoT twin of the infrastructure element is in control of the element's blockchain wallet. A blockchain, on the other side, is the event bus and the recording ledger, but it is *not* a first-level, self-aware, individually-acting entity with own interest. At the same time, the participants of the blockchain network safeguard the outcome of the peer-to-peer transactions, because these transactions (*e.g.* admitting a train into a track section) are safety-relevant and affect all peers—not just two.

State changes on a blockchain consist of several steps. Independently of the technology used, the three core steps on the "happy path" (in the absence of errors) are transaction proposal, transaction validation, and the replication of the validated transaction; there might also be technology-specific steps such as ordering of the transactions. The transaction validation is the most intricate step: it is where "mining" could be included to combat spamming and to introduce incentives to "compute" the block. In our situation (private network), transaction validation is the trust-intensive step. The important design question to answer here is: how many network participants have to vote in favor of a transaction to validate it? The possible answers are "at least n" (n≥1), "at least 51% of all participants," etc. Our architecture is very flexible with regard to the validation algorithm, and we have used Proof-of-Authority while also experimenting and

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

adjusting the consensus thresholds with Proof-of-Work (with minimal complexity) and also Proof-of-Stake.

For the initial implementation, we have chosen Solidity as the programming language for smart contracts and the open-source Ethereum blockchain (geth and Parity) as the DLT product, deployed in a private/consortial setup. As our work progresses beyond prototyping and as other ledger technologies (*e.g.* Quorum, R3 Corda, Hedera Hashgraph, Hyperledger Fabric, Hyperledger Burrow, IOTA, Neo, etc.) improve, we will re-evaluate this choice. It is outside the scope of this paper to provide another extensive comparison of Ethereum with other blockchain technologies; for such work, see *e.g.* Kapsammer *et al.* "The Blockchain Muddle," (2018), and Valenta and Sandner, "Comparison of Ethereum, Hyperledger Fabric and Corda," (2017).[41, 42]

The following arguments have led us to the choice of Ethereum over other technologies. In our view, none of the competing products is as good as (or better than) Ethereum for *all* of the following:

- Ethereum is an open-source set of technologies, based on a community-driven ecosystem and powered by a large active audience of developers and researchers.
- There are *multiple* ready-to-use implementations of Ethereum (*e.g.* geth, Parity,[43] Hyperledger Burrow,[44] Quorum,[45] and others) which support Proof-of-Authority (PoA), Proof-of-Stake (PoS), and Proof-of-Work (PoW) consensus algorithms.
- The core Ethereum technology (EVM, P2P networking, cryptographic underpinnings) has matured significantly and powers a large, self-stabilizing unpermissioned public network (mainnet) with different implementations.
- Ethereum networks can be both unpermissioned (by default) and permissioned (using the aforementioned Quorum implementation).
- There exists a large codebase of security-checked contracts and templates written in Solidity (*e.g.* OpenZeppelin),[46] and automated tooling to check Solidity code for errors and bugs.
- The built-in wallet functionality, the established "wallet contract" mechanisms, and the token contracts lay a solid foundation for M2M payments and contracting.
- There is an abundance of libraries to interface Ethereum networks/nodes and deployed contracts from a variety of programming languages (*e.g.* from JavaScript using the official web3.js API/library, from Java using web3j,[47] from .NET using Nethereum,[48] etc.), in addition to native RPC/JSON communication.
- So-called "light nodes" can interact with the network without downloading the entire blockchain; [49] Ethereum implementations can run on very constrained hardware.
- The Ethereum Virtual Machine (EVM) and the EVM bytecode are subject to intensive research into formal verification,[50,51] theorem proving,[52,53] and security analysis.[54]
- The by-design ability to write smart contracts not only in Solidity, but in any language for which a compiler to the EVM bytecode exists (*e.g.* in Vyper or Serpent)[55, 56] offers a similar potential as the Java VM/bytecode: beyond Java itself, other programming languages (*e.g.* Kotlin, Scala) can step in and integrate.
- Using Quorum, it is possible to utilize an equivalent to the privacy-preserving concepts of "channels" and "private data" in Hyperledger Fabric.

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

- The concept of "gas" to pay for processing of transactions, which incentivizes economical and performance-oriented code and ensures that the execution duration is bounded.

At the same time, it should be mentioned that blockchain-native tokens and assets *can* be added to most ledger implementations, though with the risk of "reinventing the wheel." Other products may also provide more choices with regard to mainstream programming languages for smart contracts (*e.g.* Hyperledger Fabric, which supports Go, Java, and JavaScript), while Solidity is complex and the dominant smart contract programming language available in Ethereum, so it has to be learned no matter which other languages are already known to a person.

In this context, an often-cited disadvantage of using Ethereum is that, "out of the box," it is an *unpermissioned* (permissionless) blockchain: *theoretically*, everyone (within the consortium) is free to set up one or several nodes and to participate and *potentially* every participant (and every node) is equal-righted. Technically, there is no PKI-based authentication and authorization in Ethereum (*e.g.* unlike in Hyperledger Fabric), leading to anonymity/pseudonymity. However, there are two levels of protection in place for our implementation of the rail control prototype: at the network level, protection consists of restricting access to nodes based on a whitelist. Additionally, at the level of smart contracts and assets, Ethereum's concepts and the Solidity language for smart contracts provide built-in mechanisms of ownership, delegation, and custom-defined permissions for custom-defined assets and contracts.

8. The Prototype Implementation of the Blockchain-Based Control Core

To validate our approach, we looked for a physical system that would be as close as possible to a real-life mainline railway. At the proof-of-concept stage, using a real railway would incur risks that could affect human lives, and a "secluded" full-scale test setup was not available. As a replacement, the Darmstadt training facility for infrastructure operators ("Eisenbahnbetriebsfeld Darmstadt" near Frankfurt am Main, known as EBD, *cf.* Figure 3) was a very good opportunity:

- The facility is a long-standing joint venture between a research university (Technische Universität Darmstadt) and the Deutsche Bahn AG (through its DB Netz AG subsidiary).
- EBD is actively being used for academic teaching and railway research, having been employed in a significant number of projects.[57]
- The EBD includes different generations of railway control equipment (from mechanical to electronic), so it is clear which functionality of it our approach would replace.
- The facility includes a large, complex model railway layout that is fully digital and includes a variety of train material, working signals, and switches, as well as "section occupation" information and "vehicle location" functionality (both can be used through documented APIs and over established network protocols).
- The existing localization software and hardware in the EBD, as well as the existing implementation of train protection aspects (both trainside and trackside), would remain as-is for our undertaking.

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

- The train control, dispatching, and interlocking were to be engineered using blockchain technology, following the architecture described above and controlling the train protection elements of the EBD.

We have decided in favor of the ready-to-use EBD and against a custom-made scale model railway even though the model railway would be transportable and thus better suitable for on-premise demonstrations. The reason to start with the EBD is that the reproducibility and trustworthiness of our results from the perspective of domain specialists (both non-IT and IT) is strengthened when using a validated, established third-party setup as a foundation. Our implementation supplies train control and dispatching functionality that works “on top of” the EBD-provided interfaces. Our contribution includes additional detailed checks not performed by the EBD software itself: *e.g.* we ensure that switches are in the correct position and locked also for the “converging” direction of travel (and not only in the “diverging” direction of travel, which is naturally mandatory), at any speed of the train.

In particular, our implementation takes care of ensuring safety: it controls switches and signaling, and ensures that a train’s right of way is secured against the effects of other trains. While our system controls the trackside equipment of the EBD, it is the EBD’s proven interplay between trackside and trainside hardware and software which ensures that an EBD train is brought to a halt if the train were to try to violate safety rules (*e.g.* by trying to enter a section blocked by other train). In other words, our implementation controls the ATP provided by the EBD.

As the EBD is a scale model of the current, centralized mode of railway operations, the EBD’s model locomotives cannot be active participants of the blockchain-based approach: they do not have computing power or communication facilities, even for lightweight API-based access to blockchain nodes. In fact, even the trainside safety aspects (such as emergency full-

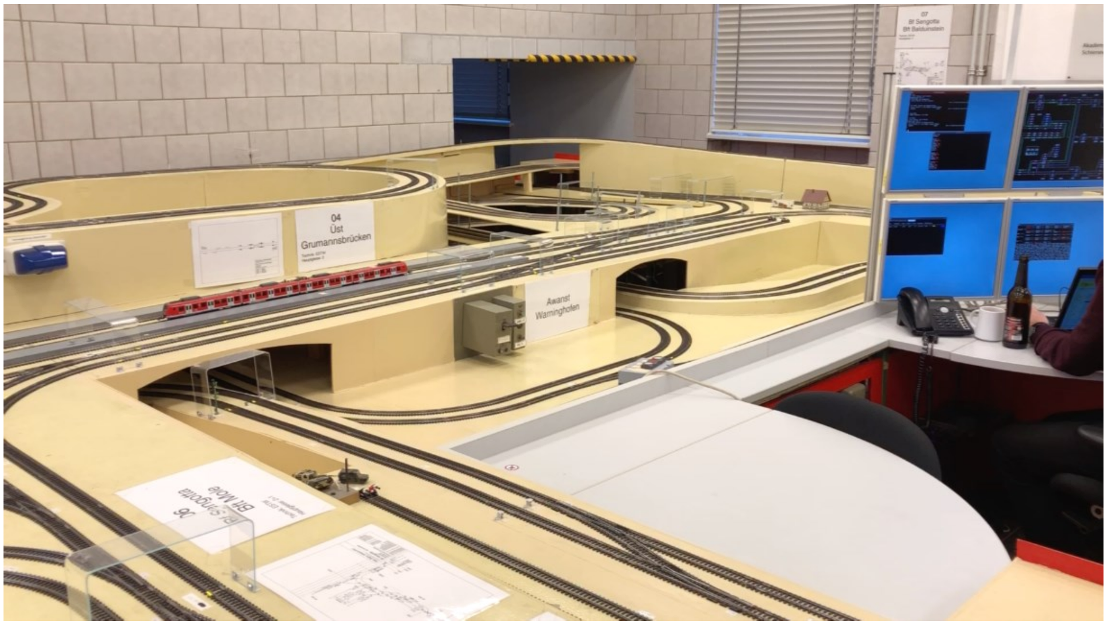

Fig. 3. EBD validation setup (excerpt of the railway layout).

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

stop if a red signal is passed) are virtualized inside the EBD by the central control logic of the model railway. To enable humans (*i.e.* train drivers) to "reserve" paths (*i.e.* necessary sequences of infrastructure elements for the specified time), we provide a graphical UI which visualizes every step, from wayfinding over payment to the actual usage. For automated operation, the underlying input (departure and arrival points and times) may be provided by passengers or by operators from the train-owning railway undertaking.

The UML sequence diagram in Figure 4 shows the steps involved in a successful reservation of a multi-resource travel path. The release of the infrastructure elements can be "as quick as possible" (as soon as the train has passed them, which increases availability), or can be triggered later (including "implicit release" when the reservation expires, provided that there is no active usage of the infrastructure element).

The route-finding on throughput-constrained, weighted/priced and time-aware graphs is already solved by several libraries and products. In our approach, the current and future reservations are stored on a distributed ledger; at the time of writing, there were no libraries that would expose such route-finding functionality directly on ledger-stored graphs. Thus, we have used a rather simple JavaScript library for finding potential routes, but JGraphT is one of the more elaborate candidates.[58] *Available* routes (a subset of *potentially possible* routes) are then determined on the basis of the ledger-stored reservations.

Both the graph search and the subsequent reservation are subject to transactional concerns such as atomicity, consistency, isolation, and optimistic/pessimistic locking. For example, if the set of the potential routes (which are found by a train) is not locked, there is a probability that another train will book a part of that potential route. Then, by the time that the initial train decides to book a potential route, it may have become unavailable.

Such problems are not DLT-specific and are solved in different ways (optimistic, speculative, pessimistic, time-constrained locking) in existing systems. To keep our prototype implementation simple, we did not implement pessimistic transaction locking but instead act in an optimistic way: we accept the possibility of a "booking failure" if the selected potential route has become unavailable in between (some or all segments, that is). In such a case, the wayfinding and reservation attempts are simply repeated.

Likewise, for the initial implementation iteration, we chose to live with possible side effects of non-atomic behavior, when a route with multiple resources is booked: if the first resource booking(s) succeed but a following resource booking fails, the already-booked segments of the chosen route are "released" (un-reserved). Then, the train has to try again.

Including locking and transactionality into the booking contract is of course possible, but we decided to do it once the technology stack for the next version of the approach has been re-evaluated: we are investigating ledgers such as Quorum, Hyperledger Fabric, R3 Corda and others because they offer sub-groups ("channels") so that not all information is broadcast to every node in the network.

Railway-operating software is subject to strict norms (*e.g.* EN 50128, 50129) and safety integrity level (SIL) procedures. In addition to rigorous testing, formal verification may be

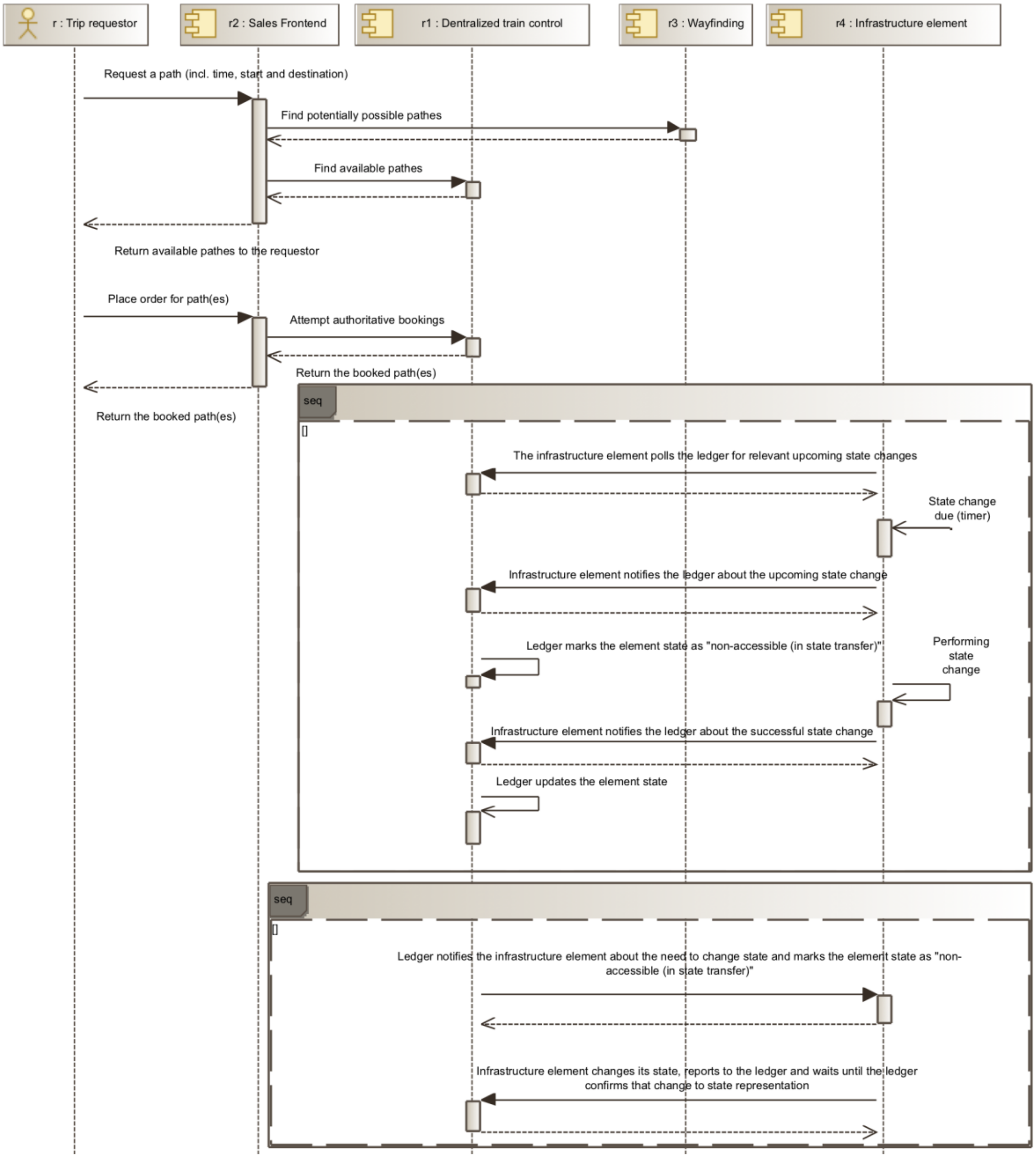
r : Trip requestor
r2 : Sales Frontend
r1 : Dentralized train control
r3 : Wayfinding
r4 : Infrastructure element
Request a path (incl. time, start and destination)
Find potentially possible pathes
Find available pathes
Return available pathes to the requestor
Place order for path(es)
Attempt authoritative bookings
Return the booked path(es)
Return the booked path(es)
seq
[]
The infrastructure element polls the ledger for relevant upcoming state changes
State change due (timer)
Infrastructure element notifies the ledger about the upcoming state change
Ledger marks the element state as "non-accessible (in state transfer)"
Performing state change
Infrastructure element notifies the ledger about the successful state change
Ledger updates the element state
seq
[]
Ledger notifies the infrastructure element about the need to change state and marks the element state as "non-accessible (in state transfer)"
Infrastructure element changes its state, reports to the ledger and waits until the ledger confirms that change to state representation

Fig. 4. Sequence diagram of the path reservation logic (excerpt, simplified view).

required to prove the suitability of both platform components and application components, as a prerequisite to certification. These tasks will be part of our future work.

Some Blockchain implementations do not prohibit or (as Ethereum) even explicitly support *forks*: branches which make the ledger a non-linear, tree-like structure. A fork marks a "split" where state changes are non-serialized and two branches can contain two conflicting statements about the same item. Forks can be intentional,[59] but fork-like situations can also happen if the network splits in two parts with no interconnections between them: each network part develops

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

its own version of the ledger. Obviously, forks (or a nonlinear blockchain *without* consistency checks) bring the risk of ambiguity—something that must be avoided in train control.

When we chose Ethereum for the prototype implementation, we were aware of the possibility of forks on the *public* ledger—and even as we are using a consortial, non-public setup, network partitioning cannot be fully excluded. As part of our future work, we plan to conduct a deeper evaluation of other DLT technologies and products, and avoidance of forks as well as the detection/avoidance of network partitions will be a key evaluation criterion.

While the Ethereum blockchain has the "replicate everything on each full node" principle, it is possible to reduce the data load (*e.g.* through *sharding*)[60] and participants can use asymmetric cryptography to encrypt private information passed over an unencrypted medium (DLT), given that the public keys (for the encryption part) are already available as part of the wallet. Other enterprise-grade DLT/Blockchain stacks (such as Hyperledger Fabric from the Linux Foundation) offer further facilities for privacy scoping, *e.g.* channels.

9. Conclusion and Future Work

In this paper, we have introduced blockchain-based decentralized railway control and the case study of its prototype implementation, which combines traditional safety mechanisms (train control and protection, interlocking) with cutting-edge IT. As a result, trains can determine possible routes and book them directly (both ad-hoc and in advance), based on transparent and binding smart contracts which ensure conflict-free resource booking. At the same time, our approach coexists with centralized batch planning that produces long-term timetables, and which integrates pre-planned construction sites across railway undertakings. For our case study, the prototype implementation has proven that the concept is viable, as demonstrated in the EBD training facility control center in Darmstadt.

As the ledger lists all binding recordings of the current status and of the advance reservations, a train-to-train economy without the risk of double-spending, deniability, or repudiation can be established. The trains, infrastructure elements, and other vehicles can participate in the asset exchange in an autonomous way, or using "human proxies" from the train-operating companies, train drivers and administrative staff; the machines become proactive members of the transportation network. Essential parts of the specialized hardware, software and infrastructure—interlocking, signals, and control centers—can be streamlined, slimmed, merged, and redesigned to be more fault-tolerant through replication and failover.

In combination with the ledger immutability and the unambiguous assignment of actions and assets to an identity, a virtual but trusted identity and a trusted curriculum vitae (CV) can be created and traced. Every part of the ecosystem (whether static or moving) becomes identifiable and possesses a history. Such data opens up new opportunities, *e.g.* in digital and predictive maintenance and in cross-enterprise asset exchange. Also, certain necessary administrative processes can be greatly simplified by using the virtual wallet for immediate monetary transactions. Vehicle usage and infrastructure services can be billed automatically; charges and reservation fees are paid seamlessly step by step, in the same system. Switches and other track elements as participants of the blockchain can receive payments.

Future research work will perform a quantitative comparison between the classic (centralized) and the novel (decentralized) solutions. The approach and the case study presented in this paper are not yet fully featured or fully implemented to executed comparative experiments and to measure metrics.

To enable a quantitative comparison, we will first scale our feasibility study to real-world railway operations, starting with a “shadow” mode (non-authoritative, passive mode) in a constrained operation scope—we target small branch railways with legacy control technology, which are due for upgrade but where the costs and complexity of contemporary, conventional, and centralized technology are prohibitive.

At this stage, it is not possible to quantify cost savings, or to predict a cost-benefit ratio. As the work progresses, we will address this question, and calculate business cases both for low-volume lines with legacy technology and for larger networks with intensive traffic. We hope that an open architecture and a collaborative approach (including vendors and railway operators) will spread the development costs over many shoulders.

From a scientific point of view, we plan to employ the blockchain in combination with multi-agent systems (MAS) and IoT middleware to study the *effectiveness* of our blockchain-based, autonomic peer-to-peer economy for railway operations. Efficiency, flexibility aspects such as the trading of priorities, additional stops for ensuring trip connections and passenger transfer, and surge pricing or seamless on-demand offers in passenger and freight transport are on our agenda, up to macroeconomic effects (*e.g.* modal split shift).

From the implementation point of view, we will start with overcoming the limitations of Ethereum: other frameworks and consensus mechanisms will be evaluated and, if necessary, integrated based on specific requirements (such as robustness against network partitioning and against forks). Also, we plan to study whether using sidechains or state channels (to reduce the workload on the chain) would be suitable without compromising safety, security, and reliability.[61,62]

From the deployment point of view, we are considering a trial with full-scale real-life equipment, *e.g.* in marshalling yards with remote-controllable unmanned mini-shunters or on secondary lines with manageable complexity and little traffic. The focus would be to bring the entire approach into an operational state and to investigate scalability, resilience and performance (latency, throughput, resource usage). Likewise, interaction and coexistence with ETCS and earlier signaling/safety systems would be studied. For this scenario, we aim at a close interaction and collaboration with the government-devised regulation authorities.

In parallel to the use of blockchain for decentralized control models for dispatching, the power of the method in railway operation lies in the field of maintenance and service. These tasks are highly labor-intensive, cost-intensive, and time-consuming. A huge number of existing delays and other shortcomings are directly or indirectly attributable to this area. A comprehensive digitization of this field includes—in addition to the sensory recording of the states of all components in real time—the proof of the execution of repairs and maintenance as well as the documentation and traceability of the individual steps. The digital twin of a train or the digital twin of a component of the infrastructure is therefore time-dependent. Its dynamics can be understood as an autogenerated CV of the corresponding component realized by blockchain technology.

Acknowledgements

Malte Plescher provided improvement suggestions and contributed to several rounds of extensive paper reviews. Michael Leining, Michael Ziemek, and Christian Klug provided valuable specialist feedback and helped set up the project. Karl Schöpf organized the funding for the project and contributed to the project scoping. Holger Kötting and Jan Niklas Dornbach were very helpful in making use of the software and hardware of the *Eisenbahnbetriebsfeld Darmstadt* (EBD) and also provided us with documentation and code samples. Niklas Armbruster, Simon Dosch, Christoph-Samuel Pitter, and Frank Polster contributed to the implementation. Steffen Ortolf contributed to resource planning and project management. Patrick Charrier challenged the design and provided valuable assistance at different stages.

Author Contributions

MK designed the architecture, led the implementation, researched related work and prepared the manuscript (80%). DK supplied additional domain knowledge and research material (10%). SJ contributed the aspects of maintenance and trusted CVs for the trains and their parts (10%).

References

[1] Calvaresi, D., Dubovitskaya, A., Calbimonte, J. P., Taveter, K., Schumacher, M. "Multi-Agent Systems and Blockchain: Results from a Systematic Literature Review." In Demazeau, Y., An, B., Bajo, J., Fernández-Caballero, A. (Eds.) *Advances in Practical Applications of Agents, Multi-Agent Systems, and Complexity: The PAAMS Collection. PAAMS 2018.* 110-126 (2018) https://doi.org/10.1007/978-3-319-94580-4_9.

[2] Kapitonov, A., Lonshakov, S., Krupenkin, A., Berman, I."Blockchain-Based Protocol of Autonomous Business Activity for Multi-Agent Systems Consisting of UAVs." In *Proceedings of the 2017 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)* 84-89 (2017) https://doi.org/10.1109/RED-UAS.2017.8101648.

[3] Afanasyev, I., Kolotov, A., Rezin, R., Danilov, K., Kashevnik, A., Jotsov, V. "Blockchain Solutions for Multi-Agent Robotic Systems: Related Work and Open Questions." *ArXiv* (accessed 14 March 2020) https://arxiv.org/abs/1903.11041.

[4] Chen, C., Xiao, T., Qui, T., Lv, N., Pei, Q. "Smart-Contract-Based Economical Platooning in Blockchain-Enabled Urban Internet of Vehicles." *IEEE Transactions on Industrial Informatics* **16.6** 4122-4133 (2020) https://doi.org/10.1109/TII.2019.2954213.

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

[5] The specifics of these regulations can be found in the European Union Agency for Railways's European Train Control System (ETCS) Technical specifications, published as part of the Control Command and Signalling (CCS) Technical Specification for Interoperability (TSI) within the European Rail Traffic Management System (ERTMS) documentation, information about which can be found at https://www.era.europa.eu/activities/european-rail-traffic-management-system-ertms_en, with the current versions of the documents (as of 17 June 2020) listed under "CCS TSI Annex A – Mandatory Specifications," at https://www.era.europa.eu/content/ccs-tsi-annex-mandatory-specifications.

[6] No Author. "Stellwerk in der Cloud." Deutchse Bahn AG (press release) (2018) https://www.deutschebahn.com/de/konzern/im_blickpunkt/Stellwerk-in-der-Cloud-1717666. (See also the PDF factsheet: https://www.deutschebahn.com/resource/blob/1659046/f2af362033d98d6097c3770d92223415/DF_DSTW-Annaberg-Buchholz-data.pdf).

[7] No Author. "The ERTMS in 10 Questions." *European Commission* (2005) http://europa.eu/rapid/press-release_MEMO-05-235_en.htm (See especially "3. How much does the ETCS cost?").

[8] No Author. "European Rail Traffic Management System." *European Union Agency for Railways* (accessed 14 March 2020) https://www.era.europa.eu/activities/european-rail-traffic-management-system-ertms_en.

[9] On moving blocks as part of ETCS level 3, see: No Author. "Sending the Right Signals." *Thales* (2017) https://www.thalesgroup.com/en/united-kingdom/news/sending-right-signals.

[10] ATO (Automated Train Operation) definition in: No Author. "Press Kit: Metro Automation Facts, Figures and Trends." *Union Internationale des Transports Publics* (2013) https://www.uitp.org/sites/default/files/Metro%20automation%20-%20facts%20and%20figures.pdf.

[11] For a description of Differential GPS, see: No Author. "Augmentation Systems." *National Coordination Office for Space-Based Positioning, Navigation, and Timing* (accessed 14 March 2020) https://www.gps.gov/systems/augmentations/.

[12] Rosenberger, M. "Future Challenges to Axle Counting Systems." *ASPECT 2012 conference of the Institute of Railway Signal Engineers (IRSE)* (accessed 14 March 2020). Available via the Internet Archive at: https://web.archive.org/web/20181104152246/http://www.irse.org:80/knowledge/publicdocuments/3.08%20Rosenberger%20-%20Future%20challenges%20of%20axle%20counting.pdf.

[13] No Author. "Railway Applications – The Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS) Part 1: Generic RAMS Process." *VDE*

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

Verlag EN 50126-1 (2018) `https://www.vde-verlag.de/normen/0100488/din-en-50126-1-vde-0115-103-1-2018-10.html`.

[14] No Author. "Railway Applications – The Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS) Part 2: Systems Approach to Safety." *VDE-Verlag* EN 50126-2 (2018) `https://www.vde-verlag.de/normen/0100487/din-en-50126-2-vde-0115-103-2-2018-10.html`.

[15] No Author. "IEC 62278:2002: Railway Applications - Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS)." *International Electrotechnical Commission* (2002) `https://webstore.iec.ch/publication/6747`. (Note: equivalent to EN 50126, see notes 13-14.)

[16] No Author. "IEC TR 62267-2:2011: Railway Applications - Automated Urban Guided Transport (AUGT) - Safety Requirements - Part 2: Hazard Analysis at Top System Level." *International Electrotechnical Commission* (2011) `https://webstore.iec.ch /publication/6680`.

[17] No Author. "IEC TR 62278-3:2010: Railway Applications - Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS) - Part 3: Guide to the Application of IEC 62278 for Rolling Stock RAM." *International Electrotechnical Commission* (2010) `https://webstore.iec.ch/publication/6746`.

[18] No Author. "IEC TR 62278-4:2016: Railway Applications - Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS) - Part 4: RAM Risk and RAM Life Cycle Aspects." *International Electrotechnical Commission* (2016) `https://webstore.iec.ch/publication/29621`.

[19] No Author. "Railway Applications - Communication, Signalling and Processing Systems – Software for Railway Control and Protection Systems." *VDE Verlag* EN 50128/A1 (2019) `https://www.vde-verlag.de/normen/1800505/e-din-en-50128-a1-vde-0831-128-a1-2019-09.html`.

[20] No Author. "Railway Applications - Communication, Signalling and Processing Systems – Software for Railway Control and Protection Systems; Supplement 1: Additional Information for the Application of DIN EN 50128 (VDE 0831-128)." *VDE Verlag* EN 50128 VDE 0831-128 Beiblatt 1 (2016) `https://www.vde-verlag.de/normen/0800324/din-en-50128-vde-0831-128-beiblatt-1-2016-07.html`.

[21] No Author. "Railway Applications – Communication, Signalling and Processing Systems Software for Railway Control and Protection Systems." *VDE Verlag* EN 50128 (2012) `https://www.vde-verlag.de/normen/0831033/din-en-50128-vde-0831-128-2012-03.html`.

[22] No Author. "Railway Applications - Communication, Signalling and Processing Systems – Safety Related Electronic Systems for Signalling." *VDE Verlag* EN 50129 (2019)

https://www.vde-verlag.de/normen/0800576/din-en-50129-vde-0831-129-2019-06.html.

[23] No Author. “Railway Applications – Communication, Signalling and Processing Systems – Safety-Related Communication in Transmission Systems.” *VDE Verlag* EN 50159 (2011) https://www.vde-verlag.de/normen/0831028/din-en-50159-vde-0831-159-2011-04.html.

[24] No Author. “Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems – Part 1: General Requirements.” *VDE Verlag* EN 61508-1 (2011) https://www.vde-verlag.de/normen/0803012/din-en-61508-1-vde-0803-1-2011-02.html.

[25] Kuperberg, M., Sandner, P., Felder, M.“Blockchain-basierte Abrechnung der IoT-registrierten Stationshalte: ein Proof-of-Concept auf Basis von Ethereum.” *Frankfurt School Blockchain Center* (working paper) (2018). Available at: https://medium.com/@philippsandner/blockchain-basierte-abrechnung-der-iot-registrierten-stationshalte-ein-proof-of-concept-auf-basis-4534dfa9d47d.

[26] For an example of dynamic revenue splitting powered by smart contracts see: No Author. “DB Systel and IBM Reinvent Mobility by Using IBM Blockchain Technology.” *IBM* (accessed 14 March 2020) https://www.ibm.com/case-studies/db-systel-and-ibm.

[27] Böcker, J., Lind, J., Zirkler, B. “Using a Multi-Agent Approach to Optimise the Train Coupling and Sharing System.” *European Journal of Operational Research* **131.2** 242-252 (2001) https://doi.org/10.1016/S0377-2217(00)00124-7.

[28] Törnquist, J., Davidsson, P. “A Multi-Agent System Approach to Train Delay Handling.” In Timm, I. J., Schleiffer, R., Davidsson, P., Kirn, S. (Eds.) *Agent Technologies in Logistics, Proceedings of the ECAI-02 Workshop, July 23, 2002* 50-53 (2002). Available via the Internet Archive at: https://web.archive.org/web/20130120014907/http://www.ide.bth.se/~pdv/Papers/proceedingsECAI-02-WS-Logistics.pdf.

[29] For proofs-of-concept see: Gross, J. “High-Tech Blockchain Paves the Way for Cars to Pay.” *IBM Client Success Field Notes* (accessed 14 March 2020) https://www.ibm.com/blogs/client-voices/blockchain-paves-way-for-cars-to-pay/; and No Author. “Say Hello to the First Automotive Blockchain.” *Car eWallet* (accessed 14 March 2020) https://car-ewallet.zf.com/site/carewallet/en/car_ewallet.html. For an example of an interest group see: No Author. “Mobility Open Blockchain Initiative.” *MOBI* (accessed 14 March 2020) https://dlt.mobi.

[30] No Author. “Car-2-Car Communication Consortium.” *Car-2-Car* (accessed 14 March 2020) https://www.car-2-car.org/.

[31] See the NEP-5 Token Standard’s GitHub repository: Adams, T., Fong, A., luodanwg, tanyuan. “NEP 5 Token Standard.” *The Neo Project* (GitHub Repository) (accessed 14 March

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

2020) https://github.com/neo-project/proposals/blob/master/nep-5.mediawiki.

[32] No Author. “WORM Storage.” *NetApp ONTAP 9 Documentation Center* (accessed 14 March 2020) https://docs.netapp.com/ontap-9/index.jsp?topic=%2F com.netapp.doc.dot-cm-concepts%2FGUID-AE1A76A0-8B20-4A63-B391-66B3EEC896DE.html.

[33] See Archive2Azure: No Author. “Archive360 Open Archive.” *Archive360* (accessed 14 March 2020) https://www.archive360.com/products/archive2azure.

[34] See GRAU Filelock for NTFS file systems: No Author. “FileLock: The Simple Solution for Audit-Proof Archiving According to GoBD.” GRAU Data (accessed 14 March 2020) https://www.graudata.com/filelock.

[35] See SER Doxis4: No Author. “Security from the Outset.” *SER Group* (accessed 14 March 2020) https://www.ser-solutions.com/products-solutions/archiving/ document-archiving.html

[36] See: “High Availability Overview” in No Author. “Oracle Database Online Documentation 11 *g* Release 1 (11.1)” *Oracle* (accessed 14 March 2020) https://docs.oracle.com/cd/ B28359_01/nav/portal_14.htm.

[37] No Author. “Master Data Management.” *IBM* (accessed 14 March 2020) https://www.ibm.com/analytics/master-data-management.

[38] Marcelli, M., Pellegrini, P. “Literature Review Toward Decentralized Railway Traffic Management.” *Institut Français des Sciences et Technologies des Transports, l’Aménagement et des Réseaux (IFSTTAR)* (2018) https://hal.archives-ouvertes.fr/hal-01759779/.

[39] No Author. “One Project from a Single Source—Turnkey Rail Solutions from Siemens Mobility.” *Siemens* (accessed 14 March 2020) https://www.siemens.com/global/ en/home/products/mobility/rail-solutions/turnkey-rail-solutions.html.

[40] See Siemens’ Trainguard MT: No Author. “Communications Based Train Control (CBTC).” *Siemens* (accessed 14 March 2020) https://www.mobility.siemens.com /global/en/portfolio/rail/automation/automatic-train-control/ communications-based-train-control-system.html.

[41] Kapsammer, E., Pröll, B., Retschitzegger, W., Schwinger, W., Weißenbek, M., Schönböck, J. “The Blockchain Muddle: A Bird's-Eye View on Blockchain Surveys.” In Indrawan-Santiago, M., Pardede, E., Salvadori, I. L., Steinbauer, M., Khalil, I., Anderst-Kotsis, G. (Eds.) *Proceedings of the 20th International Conference on Information Integration and Web-based Applications & Services (iiWAS2018)* 370-374 (2018) https://doi.org/10.1145 /3282373.3282396.

ISSN 2379-5980 (online)
DOI 10.5915/LEDGER.2020.158

[42] Valenta, M., Sandner, P. “Comparison of Ethereum, Hyperledger Fabric and Corda.” *Frankfurt School Blockchain Center* (working paper) (2017). Available at: `https://medium.com/@philippsandner/comparison-of-ethereum-hyperledger-fabric-and-corda-21c1bb9442f6`.

[43] No Author. “Parity Ethereum Client.” *Parity* (accessed 14 March 2020) `https://www.parity.io`.

[44] No Author. “Hyperledger Burrow.” *Hyperledger* (accessed 14 March 2020) `https://www.hyperledger.org/projects/hyperledger-burrow`.

[45] No Author. “Quorum.” *J. P. Morgan* (accessed 25 June 2020) `https://www.jpmorgan.com/country/UK/EN/Quorum`.

[46] No Author. “Contracts.” *OpenZeppelin* (accessed 14 March 2020) `https://openzeppelin.org`.

[47] No Author. “Where Java Meets the Blockchain: Connect JVM Applications to Ethereum Blockchains with web3j—A Lightweight, Reactive, Type Safe Library for Java, Android, Kotlin and Scala.” *Web3 Labs* (accessed 14 March 2020) `https://web3j.io`.

[48] Blanco, J. *et al.* “Nethereum.” *GitHub* (accessed 14 March 2020) `https://github.com/Nethereum/Nethereum`.

[49] For more on light nodes and clients see: Sardan, T. “What Is a Light Client and Why Should You Care?” *Parity* (2018) `https://www.parity.io/what-is-a-light-client/`.

[50] Park, D., Zhang, Y., Saxena, M., Daian, P., Rosu, G. “A Formal Verification Tool for Ethereum VM Bytecode.” *In ESEC/FSE 2018: the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* 912-915 (2018) `https://doi.org/10.1145/3236024.3264591`.

[51] Yang, Z., Lei, H., Qian, W. Z. “A Hybrid Formal Verification System in Coq for Ensuring the Reliability and Security of Ethereum-based Service Smart Contracts.” *ArXiv* (2019) `https://arxiv.org/abs/1902.08726`.

[52] Hirai, Y. “Defining the Ethereum Virtual Machine for Interactive Theorem Provers.” In Brenner, M. *et al.* (Eds.) *Financial Cryptography and Data Security: FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, April 7, 2017, Revised Selected Papers* 520-535 (2017) `https://doi.org/10.1007/978-3-319-70278-0_33`.

[53] Yang, Z., Lei, H. “Optimization of Executable Formal Interpreters Developed in Higher-Order Logic Theorem Proving Systems.” *IEEE Access* **6** 70331-70348 (2018) `https://doi.org/10.1109/ACCESS.2018.2880692`.

[54] Torres, C. F., Schütte, J., State, R. “Osiris: Hunting for Integer Bugs in Ethereum Smart Contracts.” In *ACSAC '18: Proceedings of the 34th Annual Computer Security Applications Conference* 664-676 (2018) `https://doi.org/10.1145/3274694.3274737`.

[55] jacqueswww *et al.* “Vyper.” *GitHub* (accessed 25 June 2020) `https://github.com/ethereum/vyper`.

[56] Buterin, V. *et al.* “Serpent.” *GitHub* (accessed 25 June 2020) `https://github.com/ethereum/serpent`.

[57] Projects carried out in the Eisenbahnbetriebsfeld Darmstadt (EBD). See: No Author. “Forschung und Entwicklung im EBD.” *EBD* (accessed 14 March 2020) `http://www.eisenbahnbetriebsfeld.de/projekte/forschung/`.

[58] Naveh, B. *et al.* “JGraphT.” *GitHub* (accessed 25 June 2020) `https://github.com/jgrapht/jgrapht`.

[59] Kiffer, L., Levin, D., Mislove, A. “Stick a Fork in It: Analyzing the Ethereum Network Partition.” In *HotNets-XVI: Proceedings of the 16th ACM Workshop on Hot Topics in Networks* 94-100 (2017) `https://doi.org/10.1145/3152434.3152449`.

[60] No Author. “Sharding-FAQs.” *Ethereum Wiki* (accessed 25 June 2020) `https://github.com/ethereum/wiki/wiki/sharding-faq`.

[61] Singh, A., Click, K., Parizi, R. M., Zhang, Q., Dehghantanha, A., Choo, K.-K. R. “Sidechain Technologies in Blockchain Networks: An Examination and State-of-the-Art Review.” *Journal of Network and Computer Applications* **149.1** 102471 (2020) `https://doi.org/10.1016/j.jnca.2019.102471`.

[62] Dziembowski, S., Faust, S., Hostáková, K. “General State Channel Networks.” In *CCS '18: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* 949-966 (2018) `https://doi.org/10.1145/3243734.3243856`.

Ledger is published by the University Library System of the University of Pittsburgh as part of its D-Scribe Digital Publishing Program and is cosponsored by the University of Pittsburgh Press.