BlockELM - A Public Blockchain Freight Exchange Protocol

Marek Wester
TU Dortmund University
marekwester@gmail.com

Boris Otto
TU Dortmund University
boris.otto@tu-dortmund.de

Abstract

Freight exchanges are central to the logistics industry, as they reduce empty runs and meet spot demands. To improve their efficiency in terms of automation and enhance trust between the participants, we propose a decentralized freight exchange implemented using public blockchains. With our solution, we also address shortcomings of public blockchains, such as scalability and privacy. We present two artifacts: a general architecture for an electronic logistics marketplace (ELM) and a concrete implementation as the proof of concept for a freight exchange. The solution is implemented using two off-the-shelf public blockchains and a public distributed file system. Additionally, we investigate the implications for the general ELM model and show that an ELM based on a blockchain can be viewed as infrastructure rather than a market participant.

1. Introduction

Truck transport is central to our industry, since, at least for the last mile, most freight is moved via trucks. In the European Union, around 20% of all truck kilometers are for empty runs [11]. Through this, up to €100 billion costs of unused capacity were generated in 2016 [33]. This also harms the environment, as hazardous greenhouse gases are emitted unnecessarily while transporting nothing.

Freight exchanges (FX), which are ELMs [13], decrease these costs. These are used to quickly meet spot demands for truckloads [7], allowing empty trucks to be loaded by providing digital blackboards on which shippers can post their transportation demands and carriers can submit bids. FX are mainly involved in the arbitrage of services and not in their processing [12].

Land-transport margins are minimal (1.8%–5.7% in 2018 [10]), and every intermediary, such as a FX, in the chain of transport further decreases these [47]. Key issues with transportation reported in the literature [19] were reliance on the phone as the most important means of communication, difficulties in establishing trust with carriers, and reservations with central platforms due to privacy and neutrality concerns [12]. Trust between parties is a significant issue, since valuable freight is entrusted to a potentially unknown carrier, and freight theft is a serious problem [5] in FX. More than 60 FX exist in Europe. Connecting to all of them is expensive. Payments are usually made within 30–60 days, potentially causing liquidity problems for carriers as well as effort involved in chasing payments.

Electronic logistics marketplaces should have provided the solution; however, many problems remain unresolved. Electronic logistics marketplaces are operated by technology providers (TPs) [46]. Addressing the above problems might be a task for the TP or a consortium of market participants. Both solutions would add additional transaction costs. However, with public blockchains, a technology exists that can run transactions autonomously through smart contracts [42] with a minimal ram-up cost increase for the participants, so TPs can be replaced with technology. However, public blockchains involve scalability and privacy issues [8, 25, 34], which may explain why minimal research is conducted on FX running on blockchains. We aimed to close this research gap by answering the following research questions: first, we aimed to develop an architecture for a decentralized, robust, extensible, efficient, and scalable FX. Second, how can a FX be made public while maintaining the privacy of participants? Third, how can trust be created between participants?

Our contribution comprises two artifacts created following the design science research (DSR) approach. The first is an architecture for establishing an ELM on blockchains that addresses the issues of scalability and privacy. As the second artifact, we evaluated the theoretical architecture via the creation and evaluation of a prototype, generating valuable knowledge for a technical as well as a managerial audience. Creating and evaluating prototypes with DSR is common practice when investigating blockchain solutions [3, 27, 32] but is new to ELMs, since no similar research has yet been published. Using blockchains enables us to extend the traditional model of ELMs with one that can be operated by unknown and untrusted participants in open and closed configurations without a TP, thus making the ELM an infrastructure rather than a participant.

The paper is structured following the related DSR guidelines [14]. First, we introduce prior work, discuss the research method used, describe the first artifact, and evaluate it with the second artifact and further examinations. The paper ends with a discussion of the results and the conclusion.

2. Background

2.1. Electronic Marketplaces

Much research has been conducted on electronic marketplaces (EMs) [15]. One of the first definitions of an EM was provided by Bakos [2]: an EM is "an interorganizational information system that allows the participating buyers and sellers to exchange information about prices and product offerings." Four phases can be distinguished in the interactions within an EM [36, 37]. A buyer obtains information about an offering that a seller provides in the information phase. The trading phase starts with the submission of an offer, may involve negotiation between the buyer and seller, and ends with a binding contract. The transaction ends in the settlement phase with the exchange of goods/services and payments. The after-sales phase "involves after-sales product support, customer service, and evaluation of the transaction's outcome" [36].
A specific EM concerned with logistics services is the ELM. An ELM consists of "shippers, carriers and technology providers" [46]. An ELM may be open to different participants or closed (i.e., only open for a selected group) [15]. Most traditional FX are open ELMs [40], where an unlimited number of known and unknown participants can interact [15] while mainly using single modes of transport without added services [45]. Closed ELMs frequently established for a certain industry involve smaller groups where participants are known and connected, more information is exchanged, and collaboration occurs [15].

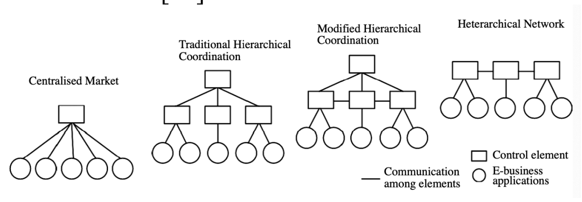

Centralised Market
Traditional Hierarchical Coordination
Modified Hierarchical Coordination
Heterarchical Network
Control element
Communication among elements
E-business applications

Figure 1 EM models [45]

Wang explored [44-46] the characteristics of closed ELMs and categorized them into four types according to the connectivity structure between the participants. The centralized market represents an open ELM where participants are coordinated via bidding and pricing systems. In the traditional hierarchical coordination, a shipper creates a private marketplace to serve their requirements. In the modified hierarchical coordination, one platform is used by several shippers to interact with carriers, and some collaboration between shippers already occurs. In the heterarchical network, much collaboration occurs between shippers with a significantly customized network design to meet the shippers' requirements. Wang states that the heterarchical network might provide the highest flexibility and efficiency [46], as it combines the strengths of open and closed ELMs.
Schwind [38] highlights the limitations of current ELMs, stating that they mainly provide support in the information phase while omitting data concerning the participants and their financial solidity and offering poor data security.

2.2. Blockchain

Bitcoin [26], introduced in 2009, was the first commercial blockchain with a protocol to transfer value via a distributed immutable ledger. Transactions are organized in blocks, created by miners, who must solve a computational problem to tamper-proof the chain. This mechanism, known as proof of work, is used as part of a consensus algorithm that results in only seven transactions per second.
Anyone can join a public permissionless blockchain [53]. However, private blockchains require participants to obtain authorization to enter them. Consensus can be achieved via proofs that are much more rapidly executed (e.g., proof of stake or practical Byzantine fault tolerance), but trust is required between the participants or between the participants and a central entity. Public chains [8] cannot be altered by a single entity, while this is mainly untrue for private chains. No trust between the participants is required, since the consensus algorithm is built for an environment with untrusted parties.
Second-generation blockchains [51] enable the use of smart contracts [42], which are executed during block creation by the participants in the blockchain. Ethereum [6, 48] is the most important Turing-complete blockchain of this type. Smart contracts must always produce the same results when run, since many miners execute them simultaneously to create blocks. Thus, no external state must be accessed, because this could lead to different results for each execution. To add the external state to a smart contract, oracles [50] can be used.
Newer generations of blockchains attempt to overcome the limits of scalability, such as the public IOTA [31] protocol, where transactions are linked via a directed acyclic graph data structure, called tangle, to achieve high throughput. Since no miners are involved, no fees are incurred. By continuously removing unwanted transactions, IOTA sacrifices some of its immutability for the sake of performance.

Blockchains are not well suited to store larger files such as freight documents. The InterPlanetary File System (IPFS) [4] is a distributed file system that allows decentralized file storage. A property of this is the immutability of files. Files are referenced by their hash value, so a file behind a certain address will always be the same.
Ideas for using the IPFS as a store to offload data from the main chain were discussed [52]. Here, a custom blockchain is required, which would be counter to our goal to use standard components. Another approach to storing Internet of Things data in the IPFS as a side chain was conducted [1]. Ethereum and Monax were used as a consortium chain that enabled access to the data packages in the IPFS. However, the architecture is limited to a private blockchain, since the mechanism of updating the data might cause scalability problems.
Privacy is an issue that still requires resolution in public blockchains. Since all nodes contain a copy of all data, no private information should be stored in them. We needed to investigate the metadata, such as the transactions, the content that is part of the transactions, and the state stored on the chain. Much research was conducted concerning the metadata. Protocol extensions exist that require off-chain computation, third parties, much computation, and coordination using, e.g., Hawk [22] or Enigma [54]. A set of algorithms called zero-knowledge proofs [30] allows one party to prove that they possess a secret without revealing it. This can be used to hide parts of transactions and enable complete anonymity. None of these can currently be executed efficiently on a smart-contract-enabled blockchain [8]. They also require a trusted setup, which the participants must accept.
No research has been conducted on placing a FX on a blockchain. Since ELMs are a subset of EMs, we investigated their concepts. For example, Notheisen [27] created a marketplace for used cars on a public blockchain to replace a central register with a decentralized solution. This included a component for trade as well as a register and a process for registration. Others seeking similar approaches in different fields of application followed a schema using Ethereum as a main chain, with some adding a side chain [18, 20, 24, 28]; these lack scalability, because the requirement of being able to place many offers for one service request would not scale with the proposed architectures. Moreover, privacy is not well implemented, as the transactions can be tracked. More privacy-aware concepts [23, 41] exist that allow part of a sale or reviews to be anonymous; however, these systems require specialized blockchains or must be executed on private blockchains. Moreover, auctions [43] were implemented using a blockchain; however, scalability and privacy were not sufficiently addressed.

3. Method

The research was conducted following the DSR [17] and related guidelines [14]. This method was chosen because we aimed to develop a novel technical solution for decentralized organization of freight order allocation. We followed the guidelines for design science [17]: we introduced an architecture (model) for a ELM that could also be used in a more general sense for other service exchanges. To evaluate the validity of the architecture, we introduced a model instantiation as an additional artifact.
The problem relevance is high. Currently, only centralized FX are available. Considering the negative effect of empty runs on the climate and the economy, a decentralized FX might prevent some of these. Since no such FX currently exists, our approach might solve a crucial problem.
Through the implementation, we conducted a design evaluation. In simulations, we calculated the cost of transactions and showed via a deductive process that the required privacy was achieved. We defined use cases that were tested using test scripts. Additionally, we validated the scalability promises of the involved components.
Our research contribution is an architecture that can be applied to a FX as well as to other areas where multistep negotiations can be conducted independently of the negotiation result, which must remain private and immutable in a distributed ledger. We provided a blueprint on how to implement decentralized, scalable, open, and privacy-aware FX. We also investigated the consequences of using a blockchain on the ELM model.
We ensured research rigor by following the DSR process and principles and by using a structure suggested in the literature [14]. To validate the model, an instantiation was created, which was additionally validated through automated tests.
In DSR, design is a search process. Both artifacts were created through iterations. The insights from the model instantiation influenced the model adjustment to better fit the state of the problem environment, as described in the evaluation section.
The results of this research are presented through this paper as well as through presentations to the beneficiaries of this study in a managerial setting.

4. Artifact Description and Architecture

To verify whether a blockchain should be used, several similar decision models are available [29, 49]. A flow chart determines which blockchain to use, if any [49]. We store state, have multiple writers (our participants), and do not wish to trust a third party. Our writers are not known centrally and no trust exists between them. Since public verifiability is an important property of our architecture, we were guided to use a public blockchain.

The architecture was designed via an iterative process, regularly testing it with the prototype. We were guided by the design process for blockchains by Xu [51] to evaluate different options in the implementation, which then influenced the architecture. Layering the data store was also inspired by a literature study [16].

4.1. Architecture

The ELM model [46] involves a shipper (S) who requests a service and wishes to obtain offers from carriers (Cs). The S selects an offer and the C executes the service. To authorize payments for the service, the service quality must be verified and reported back to the exchange. To achieve this, the ELM model must be extended to include a validator (V) who performs that role. In contrast to the ELM model, we replaced the TP with the blockchain.

The data repositories are divided into three layers. The **management layer** maps the information phase, settlement phase, and after-sales phase of the EM model [36, 37]. The **negotiation layer** is used to store the process of establishing the contracts, mapping the negotiation phase of the EM model [37]. The **binary large object (BLOB) store** is used to securely store bulk data.

The durability and cost decrease with each layer of data stores. In the first layer, the highest durability is required, because this is where the payments are made. No money should be lost. This will also be the most expensive layer, as durability increases the storage price. For the other layers, some of the durability can be sacrificed, since the negotiation data is only important while the contract is unsigned, and data in the BLOB storage only needs to be downloaded once. Afterwards, it can be confirmed which data was available when the contract was signed.

The **management layer** holds the signed contracts, the money, information about the Cs, and a repository with references to the negotiation layer. After the contracted services are conducted, payments should be released. Thus, the implementing blockchain technology requires the ability to execute smart contracts so that the payments can be released automatically without involvement from the S, which requires the necessary information to be available in this layer. Since the cost of storing data in this layer is high, the minimum amount of data is stored. The data in the management layer is organized in the following repositories:

Signed contracts repository This holds the contracts. Its properties are payment amount, due date for the service completion, C, S, and V.

C/S/V repository This contains information about the Cs, Ss, and Vs. Its properties are unique identifier, public key, name, and link to the BLOB store with further information.

Service request repository list Here, Ss store references to their service request lists in the negotiation layer. Cs use this as a root for their search for service requests. Its properties are unique identifier, owner, name, link to negotiation layer, and minimum deposit.

Deposit repository Here, a C can deposit money for an S so that the S can collect the money in case of an exception in the service. Its properties are S, C, amount, and valid until.

The **negotiation layer** stores the service requests and offers. The implementing technology must support the speed of data changes and have low operating costs because of the larger amount of data. Additionally, the negotiation layer is used to determine who can access the service request list. The repositories of the negotiation layer are as follows:

Service request list This contains a list of service requests from one or many Ss as well as the quotes from the Cs. A service request must contain all the information necessary to enable a C to make a binding offer. Additional documents are references in the BLOB store. The list must be encrypted. The S can provide access to interested parties by providing the necessary key. The C sends encrypted quotes that only the S or, depending on the operation mode, the other Cs of the service request list can read.

Service request list approval queue Here, the C can request access to a service request list by providing a public key. The S uses this to provide the encrypted access keys to the service request list. The public key provided here is not required to be the same as that in the C repository.

The last layer is the **BLOB store** for storing large data objects. These objects are referenced from the other layers. It must be possible to prove that the file stored in the BLOB store is the one referenced from the other stores. This implies that the files in the BLOB store must be immutable, as the references are immutable as a property of a blockchain. The data here should be encrypted.

One goal of this research was to develop a process without media disruption to maintain low transaction costs. Therefore, the payment must also be conducted on the blockchain. Since native blockchain cryptocurrencies are often highly volatile [35], a more stable currency is required. Therefore, in this case, an optional token should be used that can be exchanged at a stable price for fiat money.

The final component of the exchange is the client software, which provides a user interface.

Protocols

The protocol is divided into the following categories: setup management, service request handling, contract management, and exception handling.

Setup Management

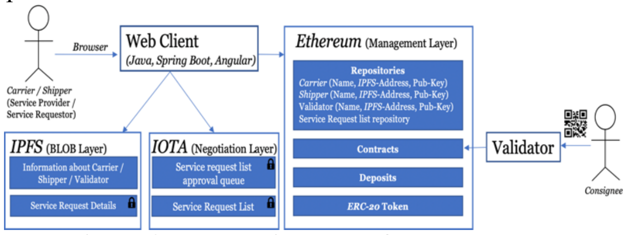
Creation of a carrier, shipper, or validator If a new C, S, or V wishes to join the exchange, they must register with the public registry. This is necessary to allow the participants to build trust between each other. This allows a C to prove successful contract executions, because only they can produce the matching public key offered in their profile.

Creation of a service request registry Service request registries can be only created by any S registered in the S repository, so that it is clear who created it.

Service Request Handling

Creating service requests The S adds a signed service request to one of its service request lists so that the origin of the request is provable. The request data defines the exact information required to offer a price for the service. The request might also contain data to enhance collaboration between the participants, such as forecasts for further shipments.

Each service request is valid for a certain time. It also has a deadline by which the service must be started and another by which it must be completed. The S also specifies the validator that must be used to validate the correct service execution. For a service exception, the S also specifies an amount the C must pay if they do not provide the service.



Carrier / Shipper (Service Provider / Service Requestor)
Browser
Web Client (Java, Spring Boot, Angular)
Ethereum (Management Layer)
Repositories
Carrier (Name, IPFS-Address, Pub-Key)
Shipper (Name, IPFS-Address, Pub-Key)
Validator (Name, IPFS-Address, Pub-Key)
Service Request list repository
Contracts
Deposits
ERC-20 Token
Validator
Consignee
IPFS (BLOB Layer)
Information about Carrier / Shipper / Validator
Service Request Details
IOTA (Negotiation Layer)
Service request list approval queue
Service Request List

Figure 2 The architecture of the ELM

Finding service requests The C can find service requests by browsing the service request list repository. From there, they can check the service request lists for new service requests.

Accessing a service request list and handshake To access the service request list, a C must send a request to the service request approval queue. The request is encrypted with the public key of the S. This contains a newly generated public key that the C must use in this service request list as well as the unique identifier that the C used in the C repository. The S can now validate how much money the C deposited, who they are, and how many contracts they completed successfully. The S can then decide to send the C the address as well as the access key to the service request list in a message encrypted with the public key the C sent in this message exchange.

Responding to a service request with a quote The C attaches one or more quotes to a service request encrypted with the public key of the S. The offers may have a date until which they are valid.

Contract Management

Accepting an offer The S selects a winning offer by sending a message to the service request list. The part showing that the service request is closed is visible to the participants of the service request list, while the part pointing to the documents with the service details is encrypted with the public key used in the service request list of the winning C. The S then creates a contract on the management layer containing a hash value that identifies the service request and the winning quote. This can be used later to prove which terms lead to the contract without exposing the data to the public. The S must also send tokens to the contract so that the contract is paid on completion regardless of whether the S remains solvent.

Service completion and payment On service completion, the validator specified in the contract is called by the C or by other means (e.g., the service beneficiary or a GPS tracker on reaching the destination) to validate service completion. The validator confirms whether the service was conducted as specified. If the service was completed correctly, the validator triggers the service contract to release the funds to the SP.

Exception Handling Trust between the participants can be achieved through both the immutability properties of blockchain and the risk management undertaken by both parties. For cases where the service was not executed as agreed, clear rules must be established to resolve the situation on chain to minimize transaction costs. The main idea behind our resolution strategy is for the C to deposit funds into the contract that can be withdrawn by the S in case of noncompliance with the contract. The amount of compensation is defined in the service request. Every party can precisely assess the financial risks they are willing to take. This also addresses the goal of creating trust between the participants. When the contract is not validated by the validator within the defined time frame, the S can claim the agreed amount from the funds deposited by the C through the smart contract.

Similarly, the C can claim a payment if, for example, the S did not allow the C to provide the service. However, no automated process can be offered here, as it is impossible to efficiently prove the misbehavior of the S on chain. Involving a third party would result in additional overheads and might not be an economically valid option. Therefore, the S must approve this claim. Since the whole interaction is stored on chain, it might benefit the S to approve a claim to earn the trust of other Cs.

5. Evaluation

The general design decision for the prototype was to build on standard components to support the reliability and stability goals and increase acceptance by the users. Ethereum is used for the management layer, as it is the most successful blockchain in terms of market capitalization (coinmarketcap.com), supporting smart contracts and offering both high durability and reliable security [32]. The smart contracts were created using the Solidity language. As the internal currency, an ERC20-compliant token was created to be compatible with public token exchanges.

The remainder of the contracts on Ethereum is divided between the repositories, service request repository list, service contract handling, and exception handling, as specified in the general architecture. The repositories contain the handling of shippers, carriers, and oracles (validator) allowing self-registration. The data required for the registration is the name, IPFS address containing further description, and a public key (elliptic curve cryptosystem [21] to save space compared to RSA). The service request repository list contains the list of service request repositories, which each contain an owner, name, IOTA address of the service request repository, and a minimal amount that must be deposited to access the list. Any shipper can create such a list. The part residing on Ethereum is governed by smart contracts, since financial transfers must be reliable and depend on the agreement between the parties and not their willingness to cooperate.

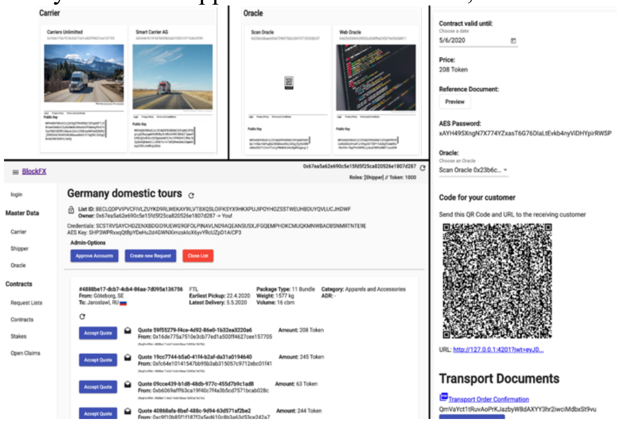
Ethereum alone cannot scale cost, speed, or volume. Because of its speed and scalability, resulting from its leaner approach to the consensus algorithm, we chose IOTA for the negotiation layer, as adding transactions to the tangle involves no direct cost. Since IOTA does not offer smart contracts, compliance with the rules is achieved through a protocol and encryption; IOTA offers the second-layer masked authenticated messaging (MAM) protocol on top of the tangle, which is essentially a linked list where messages are encrypted and signed. Anyone who has a seed key can add elements to these lists.

The IOTA tangle contains the service request list approval queue and a service request list, which are used as described in the general architecture. To encrypt the service requests, the symmetric block cipher, advanced encryption standard (AES [9]), is used, so one key can be shared with all participants, since quote requests should be accessible to them, regardless of when they joined. The key is shared during the handshake, when a carrier is approved to join a service request list.

A service request contains the minimal data required for a carrier to create a quote (e.g., a postal code but no concrete delivery address) and an ID that is unique to the list. A carrier might reply with a signed quote message encrypted with the public key of the shipper.

To accept a quote, the shipper posts a message divided into two parts, one of which is unencrypted and shows the quote ID as well as the type of the message, notifying the participants that no further quotes are required. The second part is encrypted and contains an IPFS address pointing to an AES-encrypted BLOB (e.g., PDF document) with the detailed instructions for the carrier and the accompanying AES key.

The negotiation layer only enforces compliance in posting service requests and quotes and enables the shipper to control who can participate in their section of the FX. At this point, incentives for compliance in executing the service must be set with the promise for payment. This is performed using the smart-contract-enabled Ethereum. In executing smart contracts, data storage is the most expensive component, since the data must be stored on every node participating in the blockchain. The minimal information that must be stored is how many tokens a shipper transfers to a carrier, which vali-



0x67ea5a62e690c5e15fd5f25ca820526e1807d287
Roles: [Shipper] // Token: 1000

Germany domestic tours

List ID: BECLQDPVIPVCFIVLZUYKD9RLWEKAY9ILVTBXQSLOIFKSYX9HKXPUJIPOYHOZSSTWEUHBOUYQVLUCJHDWF
Owner: 0x67ea5a62e690c5e15fd5f25ca820526e1807d287 -> You!
Credentials: SCSTRVSAYCHDZENXBDGID9UEWG9GFOLPINAVLND9AQEANSUSXJFGQEMPHOKCMUQKMNWBAOBSNMRTNTE9E
AES Key: SHP3WPRxxqQtBpYDeHu2d4GWNXimzskloX6yvYRcUZpD1AiCP3

#4888be17-dcb7-4cb4-86aa-7d095a136756 FTL Package Type: 11 Bundle Category: Apparels and Accessories
From: Göteborg, SE Earliest Pickup: 22.4.2020 Weight: 1577 kg ADR: -
To: Jaroslawl, RU Latest Delivery: 5.5.2020 Volume: 16 cbm

Quote 59f55279-f4ce-4d92-86e0-1b32ea3220a6 Amount: 208 Token
From: 0x16de775a7510e3cb77ed1a500ff4627cee157705

Quote 19cc7744-b5a0-41f4-b2af-da31a0194640 Amount: 245 Token
From: 0xfc64e10141547bb95b3ab315057c9712ebc01f41

Quote 09cce439-b1d8-48db-977c-455d7b9c1ad8 Amount: 63 Token
From: 0xb6069afff63ca19f40c7f4a3b5cd7571bcab028c

Quote 40868afa-8baf-488c-9d94-63d571af2be2 Amount: 244 Token
From: 0xc9f10b85f1f187f2a5ed610c8b3a63d53ce242a7

Contract valid until:
Choose a date
5/6/2020

Price:
208 Token

Reference Document:

AES Password:
xAYH495XngN7X774YZxasT6G76DlaLtEvkb4nyViDHYpirRWSP

Oracle:
Choose an Oracle
Scan Oracle 0x23b6c...

Code for your customer
Send this QR Code and URL to the receiving customer

URL: http://127.0.0.1:4201?jwt=eyJ0...

Transport Documents
Transport Order Confirmation
QmVaYct1tRuvAoPrKJazbyW8dAXYY3hr2iwciMdbxSt9vu

Figure 3 The web client with the carrier, validator (oracle) registries, service request view, and, on the right, the summary of a shipment before it is signed.

dator should be able to release it, and for how long this promise is valid as well as a reference to the quote request. Since the tokens are deducted from the shipper's account, the carrier can be sure that they will be paid if they provide their service and persuade the validator to release the payment. The carrier cannot stop this process after filing this contract on the blockchain.

In our implementation, the validator is a service that creates a QR code (ISO/IEC 18004:2015) that the shipper can give to the receiver of the shipment. Once the shipment arrives, the receiver scans the QR code, which encodes a URL secured by a JSON web token (RFC 7519), which then triggers the validator to release the payment on the blockchain.

The IPFS is a distributed file system that offers immutability and verifiability for a specific named resource, and it is free of charge, as it is peer-to-peer-based and therefore a good fit for the BLOB layer.

Additionally, we created a web-based client (based on Java, Spring Boot, and Angular), which can be used by the parties to allow easier access to the blockchains (see Figure 3). In designing the web client, we also created 57 automated tests to validate the functionality of the platform. To better understand how the platform can be used, we also created a complex demonstration setup with demonstration data to allow presentation of the results to a wider business audience as part of the DSR process. The web client is a stateless (the state is on the blockchain) multi-user application that can be run on the user's computer. The remainder of the platform is executed on the blockchain. The design goal of a decentralized application is thus achieved.

The platform was designed in several phases. First, we gathered business knowledge from our business partners through interviews. We then reviewed the literature on the state of the current research as well as searching the internet for solutions to the problem. This process led to an initial architecture artifact. Through the implementation, several changes in the initial architecture were required, as assumptions required revision. Particularly, the speed of Ethereum required us to minimize both the initial data model of the contract representation and the extension with IOTA. For the most used data structures in the management layer (contracts), we only stored IDs, and even the validity date was reduced to a number representing the days since the FX creation.

5.1. Scalability

The next step to validating the architecture was to calculate the cost of transactions on Ethereum, since only these incur direct costs. The contract creation and release of payments are most frequently used. Registrations to the exchange as a C, S, or V would be mainly one-time operations.

Operation	**Gas (10^3)**	**Slow <30min**	**Normal <5 min**
Create contract	60.8	0.043€	0.086€
Release payment by V	36.5	0.026€	0.052€
batch of two	52.0	0.037€	0.074€
batch of four	79.5	0.056€	0.113€
Registration carrier / V	268.8	0.191€	0.381€
Registration shipper	304.3	0.216€	0.431€
Exception handling			
request money (C-> S)	47.9	0.034€	0.068€
obtain money (C-> S)	42.3	0.030€	0.060€
stake (S-> C)	51.6	0.037€	0.073€
claim stake (S-> C)	38.5	0.027€	0.055€
Add service request list	191.7	0.136€	0.272€
Create FX	4,460	3.163€	6.325€

Table 1 Cost on Ethereum. Assumptions: slow execution 5 gwei, normal execution 10 gwei; based on ethgasstation.info, gitcoin.co/gas/history

As shown in Table 1, the cost of handling a shipment would be around 0.07–0.14 €, depending on how fast we require the transactions to be processed. Moving one Euro-pallet from Berlin to Madrid costs around 150 € (price calculator at a logistics company); this would be around 0.05% of the cost that would be added through the FX. As a percentage, this amount appears minimal. Comparing the cost to that of self-hosting, where infrastructure, servers, and personal maintenance must be paid for, this is minimal. Even creating the whole FX does not cost more than 7 EUR. From an economic viewpoint, the FX appears scalable.

Scalability also depends on the blockchain capacity. Although the IOTA protocol should contain no limiting factors, we validated the promise of scalability. To add a new transaction to the network, a small proof of work must be computed by the client. We tested the scalability with the largest piece of data from our protocol, the request for an offer, which is around
500 bytes, and we sent it via the MAM protocol to the tangle. We utilized the Amazon Web Services cloud, sending 1,000 requests for offers to the tangle to determine whether the number of transactions per second increased with the computing power.

We correctly assumed that with more computing power, the number of transactions processed within a given time frame would increase almost linearly (Table 2).

The IPFS is a distributed peer-to-peer file system. In contrast with the blockchains, the data is not stored by every node or client. The clients only help to find the data via a distributed hash table (DHT) in the network and decide independently whether they wish to store the data as a copy. Since it is trivial to run an IPFS client that delivers the data, every FX participant enhances the network; therefore, it scales sufficiently. To further validate this claim, we created a server running an IPFS node in the USA and an IPFS node in Germany. On the server in the USA, we created 1,000 random files with a size of 200 kB, which corresponds to around 50 pages of a PDF file. We added the files to the IPFS and then tried to download them in Germany via the IPFS protocol. This was possible with a short delay that often lasted only a few seconds but did not exceed 2 minutes. The time required to find the file depended on how well the peers were connected and how many other peers were already storing the file. The transfer was instant when the two peers exchanging the files were connected directly. Additionally, we reviewed a study on the IPFS performance [39], which shows that for our small file sizes, the scalability of the IPFS is sufficient.

Requests for offers / s	**Number of instances**
0.79	2x c4.8xlarge
1.41	4x c4.8xlarge
2.62	8x c4.8xlarge

Table 2 Results of the experiment on IOTA.

In Ethereum, the capacity is limited by the gas consumption allowed while creating a block, which is constantly adjusted by the Ethereum community. The highest number of transactions achieved in one day was 1.3 million in January 2018. The current rate is around 0.6 million transactions, indicating that the network retains some capacity. The gas used limits the transactions, which is

currently around 45×10^9 per day with peaks reaching 60×10^9. For the naïve case, the contract creation and release of payments for around 100,000 transactions would be possible with a gas usage of 10×10^9, which would fit into the gap and could be handled by the blockchain. Around 20%[1] more transactions would be enabled by releasing the payments in batches, as the cost of 21,000 gas [48] for calling a smart contract would occur only once. No data is available regarding how much freight is processed per day on current FX. Some exchanges publish the number of daily offers with no reference to time (when the number was achieved; how many contracts were closed; maximum vs. average per day). Considering these numbers, the median is around 100,000 shipments per day. Conducting the test runs on the Ethereum main net would provide minimal insight, since it is already known that the protocol is constrained in terms of capacity, and this would also be costly. Furthermore, tests on the test net or a private Ethereum blockchain, which we have conducted, would not be insightful, because these have other characteristics.
The blockchains involved are robust and stable. Altering their reliability might be difficult, as the nodes comprising them are distributed around the world. Ethereum has over 3,500 and IOTA more than 150 active nodes; for the IPFS, it would be trivial to run a node that only serves the local content on users' computers.

5.2. Privacy

Preserving anonymity is important. Both the shipper and carrier can create as many accounts as they wish in the Ethereum blockchain. None of the accounts is required to be bound to an identity. Thus, for a third party considering the transactions on the chain, it is impossible to determine identities. Only the parties involved in a transaction see the identity of their counterpart. Ultimately, not even the shipper is required to be identified, since the carrier only requires to know that they will be paid. However, if a shipper wishes to be more confident that the carrier will conduct the service appropriately, they could ask the carrier to show them successful contracts on the blockchain. This could be simply achieved, since the carrier could sign a proof with the key used for the transactions. A service automating this process could be another opportunity for extending the FX without changing the protocol and thus increasing trust.
This study did not address the exchange of tokens for fiat money, since exchanges exist that provide this service. By observing the tokens, one might deduct the identities of the Cs when they are exchanged into fiat money. However, exchanges often have random addresses to send the tokens to that only they can connect to an owner. Additionally, the token exchange could offer an off-chain service that would directly exchange tokens locked in contracts, leaving no traces on the public chain. The account owner could easily prove that they own the coins using a public key mechanism. Moreover, tokens could be exchanged between Cs and Ss, since the Ss generally require tokens and the Cs have them. This represents another opportunity to extend the architecture.

5.3. Extensibility

It would also be possible to integrate loans to finance transport via tokens. Services for this already exist in the Ethereum ecosystem. Other BLOB stores could be used without changing the protocol. While maintaining a stable management layer, the negotiation layer could be extended using a different blockchain, since in the architecture, the link from the service request list repository can point to any type of target.
The protocol does not prescribe how the validator should be implemented. It only provides an interface that enables arbitrary validator implementations. Other even more automated validators could be implemented with a GPS tracking device attached to the shipment. Once the device enters the destination, it can call a validator to trigger release of the payment. Another extension might be an insurance that would act as a validator. When signing a contract, the shipper would pay the insurance validator while enabling it to access information about the shipment. If the cargo did not arrive, all the required information would be available to the insurance to allow the losses to be covered.

6. Discussion

An ELM on a blockchain replaces the TP as one of three participants [45] with technology. In Wang's model the TP involves cost for the participants. The blockchain model enables an infrastructure with fewer overheads, since the TP is not involved. Our architecture enables the shippers but not the TP to choose between an open or closed ELM or even to use both. Wang [46] argues that this would be desirable, as this would enable efficiency and economies of scale (closed) and flexibility and speed (open). Our architecture enables the participants to work in centralized markets and heterarchical ELMs depending on the requirements.
The roles in the blockchain ELM do not need to be fixed, as a C can be a carrier in one list and a shipper in a different list, utilizing their network to resell orders

[1] $batchPrice(n) = 52006 + (n-2) * 13763 + n * 60754; standardPrice(n) = n * (36528 + 60754);$ with $n = number\ of\ transactions$ and $n \geq 2$; for $n \to \infty$, 23,4% gas can be saved

enabling multimodal transport, as our architecture does not restrict this. Föhring [12] adds the problem of finding a neutral TP for the FX and suggests an agent-based system as a solution. Here, our blockchain architecture might provide a suitable infrastructure on which to run the agent-based system.

The blockchain enables better process digitalization, as all phases of the EM are implemented on one technology layer, giving better cost efficiency in contrast to the information phase alone [38]. Electronic logistics marketplaces threaten the business of traditional freight forwarders [38]. These often buy capacity on the market and therefore function as another intermediary, like the ELM. When the ELM is on a blockchain, no single party directly gains from higher usage of the ELM; therefore, it changes from a competitor to an infrastructure. Carriers fear transparency through ELMs [13, 38], leading to lower prices. This effect may not be influenced by the change in the underlying technology; however, further research is required.

To achieve scalability and privacy, we separated the three areas: smart contracts, negotiation data, and file storage. This implies a design compromise, since having all the data in one system would make it possible to ensure that participants follow the rules, since the smart contracts control every transaction. The compromise is ultimately smaller than it appears. Although a S could create arbitrary contracts and collect the deposited money from the Cs, this might result in loss of trust from the Cs because of the immutable transaction log.

Considering the technologies used, some limitations remain. Although we heavily optimized the protocol, Ethereum remains a limiting factor in terms of scalability. Further developments on Ethereum 2.0 are in progress, which should allow much improved scalability.

7. Conclusions

Freight exchanges are important instruments in the logistics industry, as they avoid empty runs and meet spot demands. However, they have many drawbacks, such as lack of automation, lack of transparency, difficulty in extending, high cost, and lack of trust between participants. This study shows how these issues can be overcome by creating a FX on public blockchains, while maintaining the scalability of the solution and the required data privacy. As blockchains have their own shortcomings, such as scalability and privacy, we extracted three research questions. These were addressed by creating two artifacts in a DSR process and evaluating the results. We thoroughly analyzed the scalability aspect of the solution during the evaluation. Additionally, the exchange should be open not only to participants but also to third parties who wish to extend it. We showed examples of how this can be achieved with this architecture. Since trust is a crucial aspect when moving freight, we also elaborated on how trust is enhanced with the decentralized FX. We provide a blueprint for building decentralized, scalable, privacy-aware, and extensible ELMs that can act as an infrastructure for the logistics community.

8. References

[1] Ali, M.S., et al., "Iot Data Privacy Via Blockchains and Ipfs", Proceedings of the Seventh International Conference on the Internet of Things, 2017, pp. Article 14.

[2] Bakos, J.Y., "A Strategic Analysis of Electronic Marketplaces", MIS Quarterly, 15(3), 1991, pp. 295-310.

[3] Beck, R., et al., "Blockchain – the Gateway to Trust-Free Cryptographic Transactions", European Conference on Information Systems (ECIS), 2016

[4] Benet, J., "Ipfs - Content Addressed, Versioned, P2p File System", arXiv preprint arXiv:1407.3561(1), 2014, pp. 1–11.

[5] Burges, D., Cargo Theft, Loss Prevention, and Supply Chain Security, Butterworth-Heinemann, 2012.

[6] Buterin, V., "Ethereum White Paper: A Next Generation Smart Contract and Decentralized Application Platform.", https://github.com/ethereum/wiki/wiki/White-Paper, accessed 19.01.2020.

[7] Caplice, C., "Electronic Markets for Truckload Transportation", Production and Operations Management, 16(4), 2007, pp. 423-436.

[8] Christidis, K. and Devetsikiotis, M., "Blockchains and Smart Contracts for the Internet of Things", Ieee Access, 4(1), 2016, pp. 2292-2303.

[9] Daemen, J. and Rijmen, V., Aes Proposal: Rijndael1999.

[10] Dvz, "Weltweite Ebit-Margen Ausgewählter Logistikkonzerne Im Bereich Landverkehr in Den Jahren 2016 Bis 2018.", https://statista.com/statistik/daten/studie/829977/, accessed 15.04.2020.

[11] Eurostat, "Summary of Annual Road Freight Transport by Type of Operation and Type of Transport", https://ec.europa.eu/eurostat/product?code=road_go_ta_tott, accessed 02.12.2019.

[12] Föhring, R. and Zelewski, S., "Towards Decentralized Electronic Market Places and Agent-Based Freight Exchanges for Multimodal Transports", 11th IEEE International Conference on Automation Science and Engineering, CASE 2015, 2015, pp. 249-254.

[13] Goldsby, T.J. and Eckert, J.A., "Electronic Transportation Marketplaces: A Transaction Cost Perspective", Industrial Marketing Management, 32(3), 2003, pp. 187-198.

[14] Gregor, S. and Hevner, A., "Positioning and Presenting Design Science Research for Maximum Impact", MIS Quarterly, 37(2), 2013, pp. 337-355.

[15] Grieger, M., "Electronic Marketplaces: A Literature Review and a Call for Supply Chain Management Research", European Journal of Operational Research, 144(2), 2003, pp. 280-294.

[16] Guggenmos, F., et al., "How to Develop a Gdpr-Compliant Blockchain Solution for Cross-Organizational Workflow Management: Evidence from the German Asylum Procedure", 53th Hawaii International Conference on System Sciences, 2019

[17] Hevner, A., et al., "Design Science in Information Systems Research", Management Information Systems Quarterly, 28, 2004, pp. 75.
[18] Kabi, O.R. and Franqueira, V.N.L., "Blockchain-Based Distributed Marketplace", 21st International Conference on Business Information Systems, BIS 2018, 2019, pp. 197-210.
[19] Kleedorfer, F. and Huemer, C., "Towards a Web Based Transportation Infrastructure", e Hamburg International Conference of Logistics (HICL) – 23, 2017
[20] Klems, M., et al., "Trustless Intermediation in Blockchain-Based Decentralized Service Marketplaces", 15th International Conference on Service-Oriented Computing, ICSOC 2017, 2017, pp. 731-739.
[21] Koblitz, N., "Elliptic Curve Cryptosystems", Mathematics of computation, 48(177), 1987, pp. 203-209.
[22] Kosba, A., et al., "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts", 2016 IEEE Symposium on Security and Privacy (SP), 2016, pp. 839-858.
[23] Liu, D., et al., "Anonymous Reputation System for Iiot-Enabled Retail Marketing Atop Pos Blockchain", IEEE Transactions on Industrial Informatics, 15, 2019, pp. 3527-3537.
[24] Marathe, A., et al., "Dinemmo: Decentralized Incentivization for Enterprise Marketplace Models", 25th IEEE International Conference on High Performance Computing Workshops, HiPCW 2018, 2019, pp. 95-100.
[25] Min, H., "Blockchain Technology for Enhancing Supply Chain Resilience", Business Horizons, 62, 2018, pp. 35-45.
[26] Nakamoto, S., Bitcoin: A Peer-to-Peer Electronic Cash System, Cryptography Mailing list at https://metzdowd.com, 2009.
[27] Notheisen, B., et al., "Trading Real-World Assets on Blockchain an Application of Trust-Free Transaction Systems in the Market for Lemons", Business & Information Systems Engineering, 59(6), 2017, pp. 425-440.
[28] Ozyilmaz, K.R., et al., "Idmob: Iot Data Marketplace on Blockchain", 2018 Crypto Valley Conference on Blockchain Technology, CVCBT 2018, 2018, pp. 11-19.
[29] Pedersen, A., et al., "A Ten-Step Decision Path to Determine When to Use Blockchain Technologies", MIS Quarterly Executive, 18(2), 2019, pp. 99-115.
[30] Poelstra, A., et al., "Confidential Assets", International Conference on Financial Cryptography and Data Security, 2017
[31] Popov, S., "The Tangle", https://iota.org/IOTA_Whitepaper.pdf, accessed 15.04.2020.
[32] Regner, F., et al., "Nfts in Practice – Non-Fungible Tokens as Core Component of a Blockchain-Based Event Ticketing Application", International Conference on Information Systems, 2019
[33] Riedl, J., et al., "Why Road Freight Needs to Go Digital - Fast", accessed 03.12.2019.
[34] Romano, D. and Schmid, G., "Beyond Bitcoin: A Critical Look at Blockchain-Based Systems", Cryptography, 1, 2017, pp. 15.
[35] Routledge, B. and Zetlin-Jones, A., Currency Stability Using Blockchain Technology, No 1160, Meeting Papers, Society for Economic Dynamics, 2018.
[36] Scharl, A., Evolutionary Web Development, Springer Science & Business Media, 2012.
[37] Schmid, B.F. and Lindemann, M.A., "Elements of a Reference Model for Electronic Markets", Proceedings of the Thirty-First Hawaii International Conference on System Sciences, 1998, pp. 193-201 vol.194.
[38] Schwind, M., et al., "Electronic Transportation Marketplaces: How Can Green-Is Help to Promote Sustainable Logistics?", 2011 44th Hawaii International Conference on System Sciences, 2011, pp. 1-8.
[39] Shen, J., et al., "Understanding I/O Performance of Ipfs Storage: A Client's Perspective", Proceedings of the International Symposium on Quality of Service, 2019, pp. Article 17.
[40] Skjøtt-Larsen, T., et al., "Electronic Marketplaces and Supply Chain Relationships", Industrial Marketing Management, 32(3), 2003, pp. 199-210.
[41] Soska, K., et al., "Beaver: A Decentralized Anonymous Marketplace with Secure Reputation", IACR Cryptology ePrint Archive, 2016, 2016, pp. 464.
[42] Szabo, N., "Smart Contracts", Unpublished manuscript, 1994
[43] Van Moergeste, L., et al., "Using Blockchains for Agent-Based Auctions", 10th International Conference on Agents and Artificial Intelligence, ICAART 2018, 2018, pp. 192-199.
[44] Wang, Y. and Naim, M.M., "B2b E-Business Reference Architecture for Tailored Logistics", International Journal of Services Operations and Informatics, 2(3), 2007, pp. 253-266.
[45] Wang, Y., et al., "Electronic Marketplaces for Tailored Logistics", Industrial Management and Data Systems, 107(8), 2007, pp. 1170-1187.
[46] Wang, Y., et al., "An Exploratory Study of Electronic Logistics Marketplaces and Its Impact on Customised Logistics", POMS 18th Annual Conference Dallas, Texas, USA, May, 2007
[47] Witkowski, J., "Electronic Freight Exchange and Logistics Platforms in Building of Supply Chains", CLC 2018: Carpathian Logistics Congress: Conference Proceedings, 2018, pp. 77.
[48] Wood, G., "Ethereum: A Secure Decentralised Generalised Transaction Ledger", https://ethereum.github.io/yellowpaper/paper.pdf, accessed 19.01.2020.
[49] Wüst, K. and Gervais, A., "Do You Need a Blockchain?", 2018 Crypto Valley Conference on Blockchain Technology (CVCBT), 2018, pp. 45-54.
[50] Xu, X., et al., "The Blockchain as a Software Connector", 2016 13th Working IEEE/IFIP Conference on Software Architecture, 2016, pp. 182-191.
[51] Xu, X., et al., "A Taxonomy of Blockchain-Based Systems for Architecture Design", 2017 IEEE International Conference on Software Architecture (ICSA), 2017, pp. 243-252.
[52] Zheng, Q., et al., "An Innovative Ipfs-Based Storage Model for Blockchain", 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI), 2018, pp. 704-708.
[53] Zheng, Z., et al., "An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends", 2017 IEEE International Congress on Big Data (BigData Congress), 2017, pp. 557-564.
[54] Zyskind, G., et al., "Enigma: Decentralized Computation Platform with Guaranteed Privacy", arXiv, 1506.03471v1(1), 2015, pp. 1-14.