An Application of Ethereum smart contracts and IoT to logistics

Leonor Augusto
Department of Electrical Engineering
Faculdade de Ciências e Tecnologia, UNL
Caparica, Portugal
l.augusto@campus.fct.unl.pt

Ruben Costa
Department of Electrical Engineering
Faculdade de Ciências e Tecnologia, UNL
Caparica, Portugal
rddc@uninova.pt

José Ferreira
Department of Electrical Engineering
Faculdade de Ciências e Tecnologia, UNL
Caparica, Portugal
japf@uninova.pt

Ricardo Jardim-Gonçalves
Department of Electrical Engineering
Faculdade de Ciências e Tecnologia, UNL
Caparica, Portugal
rg@uninova.pt

***Abstract*—Over the last few years, interest has emerged in blockchain, a decentralized ledger technology (DLT) created for use in cryptocurrencies, but with a great potential to be used in other application domains. One of them is supply chain management, tracking and tracing, which are key processes to the logistics industry, made difficult due to lack of standards or trust between actors, miscommunication, fraud, bureaucratic delays, among other issues. In order to overcome some of these challenges, the solution presented in this article proposes a blockchain system application created with Ethereum smart contracts technology, to be used in supply chain and logistics for tracking and tracing products through the storage of valuable data in a trustworthy and decentralized system. The technical solution presented here implements methods for automated tracking, certification and authentication, and integrates the communication of blockchain with IoT devices, which play a role in monitoring products and automating the tracking and clearance processes. We validate our approach by developing an application that highlights the benefits of these technologies applied to logistics, ultimately giving insight into the capabilities, qualities, but also limitations a system like this can have.**

I. Introduction

A. Blockchain and logistics

Over the last decade, concerns over the privacy and security of the most common computer-based systems have led to a surge of interest in their decentralization. One of the innovations that emerged from this interest was blockchain, a DLT (Decentralized Ledger Technology) that was originally created in 2008 by Satoshi Nakamoto [1], and serves as the basis of Bitcoin and all other cryptocurrencies.

Blockchain, like any other technology, has benefits and drawbacks, but it can be particularly well applied to transaction systems that must be secure, trustworthy and decentralized, and where transparency and traceability are advantageous. This is due to its reliance on distributed database-keeping, where every transaction/data alteration's integrity is kept secure through varied consensus system solutions, which can be adapted to the kind of network at hand but often rely on cryptographic security technology as their basis.

As such, it has been increasingly noted that blockchain technology can be used outside of the cryptocurrency spectrum and potentially benefit other kinds of applications [2] - public notaries, authentication systems, anti-counterfeit mechanisms and decentralized storage are just few, but in this article we will focus on an instantiation of a blockchain application for product tracking in logistics, and aided by Internet-of-Things (IoT) technology.

Logistics refers to the business of maintaining and managing supply chains of goods, from their point of origin to the point of consumption. This must be done in a way that meets the requirements of government authorities and customers. With every exchange of carriers/bearers, the transaction of data must also occur in the form of documents, certifications, contracts and other important data. Having in mind common problems such as faulty tracking and tracing, data tampering and costly bureaucracy delays, the use of a decentralized, immutable ledger such as blockchain as a common platform for companies in the logistics industry, coupled with the use of IoT sensor devices for tracking, could have significant impact in the improved security, speed, trust and transparency for data exchange occurring in supply chains [3].

However, due to varying industry standards and the large diversity of items being maneuvered, logistics systems contain a fairly high degree of complexity. Their application in blockchain can only be done due to some recent innovations on this technology that have given developers more freedom and versatility for developing applications. Some of the most notable of these inventions are smart contracts.

Smart contracts were first successfully implemented in blockchain by Vitalin Buterik in 2014, via the Ethereum platform [4], being essentially protocols that verify, secure and enact transactions or agreements between consenting parties in a decentralized network. They enable developers

978-1-5386-9282-0/19/$31.00 ©2019 IEEE

to create blockchain systems that adhere to the most varied sets of rules for ownership, transactions and state transitions, permitting also the decentralized storage and management of data. In Ethereum technology specifically, smart contract code is stored in the blockchain. To interact with it, users execute function calls in the form of transactions, which can then change the smart contract's state. Smart contracts are therefore comparable to state machines that exist within a blockchain.

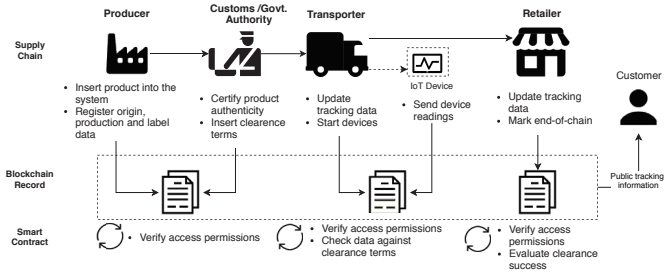

Fig. 1: Conceptual view of supply chain tracking of goods supported by blockchain and smart contracts.

Within this paper we propose a blockchain based, Ethereum smart contract application for use in logistics, focusing on how IoT devices can communicate with our blockchain system for the purpose of aiding the tracking of a product's journey throughout a supply chain. The steps for this process are laid out in Figure 1, showing how a product can be traced from its origin by having its tracking data continuously updated by users and/or IoT sensor devices, which can measure and afterwards register on the blockchain information on the items condition, location or status. This information, whether input by devices or authorized users, is continuously verified against any terms for transport or quality requirements that authorities have seen fit to create and enforce, so that the product's status is automatically evaluated and approved via smart contracts.

The smart contract system presented here implements an RBAC (Role Based Access-Control) system, product tracking and a clearance/quality control system that integrates IoT device use. With these mechanisms in place, and taking into account the untamperable nature of blockchain systems, trust, transparency and data integrity can be achieved both between collaborating logistics entities and with the customers at the end-of-chain, in a blockchain system that provides trustworthy information on the products being supplied.

The objective of the work presented here is to validate a proof-of-concept into a functional blockchain smart contract-based application that interacts with and responds to IoT device data. We tackle some of blockchain's issues as a technology - namely scalability and cost of data storage - in our implementation study, providing some insight into this technology's potential for application in logistics, but also some of its limitations and problems.

B. Objectives

The main objectives for our blockchain system that we will address within this paper can be described as follows:

1) Our blockchain application will be designed in order to permit the tracking and tracing of items. Three mains status changes are contemplated and registered - location, current bearer and state - as well as other types of readings made by devices. The system will be permissioned (meaning only accessible by authorized parties), and utilize Role Based Access Control.
2) IoT devices can connect to the blockchain, and through it receive orders for beginning/ending data acquisition during the shipment of items. They can afterwards register this data in the blockchain.
3) Every time new data is input into the system, smart contracts check it against clearance terms/rules for transport, created by the producer or by customs or government authorities. These entities can also register certifications attributed to items in transport.

With these objectives in mind, the solution being developed will promote transparency to supply chains, provide support for multiple industry standards and showcase the possibilities for automatizing tracking processes, including access control and certification.

This document is structured in the following manner: Section II will provide a brief overview of the State of the Art; Section III contains a detailed description of our conceptual model; an implementation using Ethereum-based smart contract technologies is described in Section IV; Section V presents and validates the results achieved, and finally section VI concludes the paper and points out future work.

II. State of The Art

In this section we will give an overview of the current state of studies, investigations and developments on the topic of blockchain for logistics and its coupling with IoT.

A. Related Work

There are quite a few published articles that give a good overview of the current state of the art on blockchain and logistics. Petersen, Hackis and See [5] published a study on the industry-wide interest and perspective on blockchain solutions for logistics, concluding that there is indeed significant investment being made by corporations, especially for blockchain applications that enable tracking, tracing, finance/payment processing or a combination of these. This is further evidenced by announcements made by prominent companies in this field. IBM and Maersk, for one, have have a partnership project of an open-source logistics blockchain [6].

It is worthy of note that most of the early advances on blockchain for logistics scenarios have been done by independent developers and startups. Provenance [7], one of the earliest, proposed blockchain for tracing the origins of products via QR code tags. Similar projects, such as OriginTrail, Sweetbridge, Blockfreight or Ledgit, have explored the concept of blockchain applied to logistics,with the caveat of having associated cryptocurrency tokens to their business [8].

Some particularly noteworthy academic studies on system proposals for blockchain and logistics have been done. Hasan

and Salah [9] created an Ethereum application for a Proof of Delivery system. It applied some tracking verification methods, such as testing for time limits, access control and a simple form of contract execution between two parties. Feng Tian [10] proposed in his 2017 paper a blockchain RFID-based traceability system for agri-food in a supply chain. Yuan and Wang [11] proposed a blockchain driven intelligent transportation system, to their knowledge the first of its kind, and which they argue could also be used in logistics.

Most of the projects mentioned previously suggest some form of IoT technology to help with tracing, although implementation is not explored. Christidis and Devtsikiotis [12] make a case for pairing blockchain with IoT, arguing decentralization could benefit IoT systems, including in logistics scenarios. We also find Pustišek and Kos [13] study particularly noteworthy, as they set out to analyse three different architectural approaches for the design of IoT device applications based on Ethereum blockchain, concluding that currently it is more feasible to have IoT devices connected to the blockchain through a gateway unit (a computer with more processing power). This is the method used in the implementation described here as well.

Overall, many proposals for blockchain systems geared towards logistics exist. The companies involved in these projects generally don't give a lot of technical insight into their solutions, while many of the studies done by the scientific community focus more on theoretical formulation and less on implementation. The work presented here contributes to the study of blockchain applications for logistics with the implementation of several complex mechanisms in our tracking system, as well as its connection and use with IoT device input. This is done in part with the use of Ethereum technologies, not very explored previously because while they are some of the most versatile and advanced technologies being developed, they are also costly and geared towards public network use. This paradigm has changed recently, and solutions for consortium-oriented, private network applications such as the one presented here are now more viable [14].

III. System Design

In accordance with the descriptions and objectives we have laid out in Sections I-A and I-B, we set out to design a blockchain system meant to facilitate the tracking and tracing of products as they travel through supply chains. The approach presented here will serve not only to register important product tracking data in an decentralized ledger, but will also implement an authentication and access control system, procedures that agilize certification and clearance /quality control as well as integration of IoT sensor device data in this process.

A. Product Tracking

We will start by explaining the central object of our smart contract system, which in our case is the product. Since our main objective is to track and trace products, our developments focus around the product object itself - which in the blockchain can represent a single item or a container of items. On the left side of Figure 2 the product class and the five main data points we associated with it are represented.

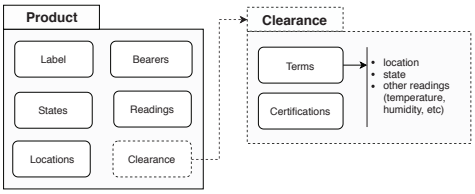

Fig. 2: The product and clearance classes, and their respective data fields.

Label - Products in a logistics system can use different types of labelling, so to enable support for varying industry standards, users register a set of key to access this data in an external smart contract.

Bearers - A list of entities that have physically held the product as it travels through the supply chain. Any change of hands is registered and timestamped.

Locations - As with bearers, changes in a product's location are registered and timestamped.

States - Coded changes in transport/processing of the product's state are registered and timestamped. We implemented this with the UNECE's Status Codes for Transport and Trade in mind [15].

Readings - Any sensor data read from IoT devices is registered here, such as temperature, humidity, or any other measurable conditions that might be important to ascertain while transporting the item in question.

Clearance - This data field concerns a products certifications given by customs, government authorities, cooperatives and the like, as well as its quality control conditions. It is further explained in Section III-C.

So for the purpose of tracking, three kinds of changes are registered

TABLE I: Different roles for actors in the blockchain and their respective permissions

Roles \ Permissions	Permission to become bearer	Change location[a]	Change status[a]	Create products[a]	Manage users[a]	Mark arrival at end-of-chain[a]	Create terms for clearance/ transport[a]	Certify a product[a]
Producer	x	x	x	x			x	
Transporter/ Warehouse	x	x	x					
Customs/ Govt. Authorities	x	x	x				x	x
Retailer	x	x	x			x		
Registrar					x			

[a]When bearer of product

There is also a role for customs, government authorities and other types of certifier entities, which are involved in processing the clearance of a product. They can register any certifications they give on the blockchain, as well as create terms for the handling of products in the supply chain.

It must be noted that almost all of the permissions to act in the system shown in Table I are also dependent of the user being the bearer of the product in question. Being the bearer of a product essentially means one is currently in physical possession of the item, and as a consequence can update its tracking status. Therefore, the roles we define in Table I come into action while the user is acting as a product's bearer.

C. Clearance

Besides the product class, Figure 2 also gives a more detailed view of state record we called clearance, which registers the certifications and terms of quality control a product has attained and successfully completed.

Government authorities or other certifiers can grant certifications on a particular product or shipment, and those events are registered under the clearance record of that product. Furthermore, authorities can create special terms or conditions for a specific transport, which are validated when tracking data is made available in the system. These terms can be relative to locations or transport states that logistic operators must fulfil in order to meet quality requirements or standards, including on measurable, sensor-read condidtions (i.e. temperature, humidity). Checking for time constraints was also implemented.

To monitor the products in the supply chain, we integrate the use of IoT devices in our system. To each device, we attribute the equivalent of a user account so that they can interact with the blockchain. These types of account have very limited permissions - they can only insert data in the form of Readings. Each device is associated to their owner entity, and can only generate readings for products whose bearer corresponds to their owner entity as well.

When the product reaches the end of the supply chain (a.k.a. end-of-chain or EOC), the accomplishment of all the terms and conditions for its quality control/clearance are validated, with the item's overall journey being judged successful, or otherwise faulty throughout some part of the process.

IV. Implementation

The implementation of the system described throughout Section III was made using Ethereum smart contract technology, and is described throughout this section. We discuss the general structure of our smart contract system's code, data storage methods we used and how user interaction happens.

A. Smart Contracts Inheritance Structure

The general view and code structure of the smart contract implementation can be seen in Figure 3, and which was developed using Truffle [16], a testing tool for smart contracts, and Ganache-cli [16], an Ethereum blockchain simulator. We used a feature Ethereum smart contracts possess called inheritance, which permits descendent smart contracts to access methods and data structures of their parent contracts (similarly to inheritance properties in object oriented programming). The final contract launched in the blockchain in our case is the 'Product Manager', but it inherits all the methods and structures of its preceding ancestor contracts as well.

The drawback of this method is that the final contract became too big, exceeding the storage size currently allowed on any one Ethereum block. A different implementation could solve this issue by having these smart contracts be launched separately and communicate via interfaces. This is what was done for the smart contract manages labelling, 'Product Labels', most easily detachable from the rest of the system.

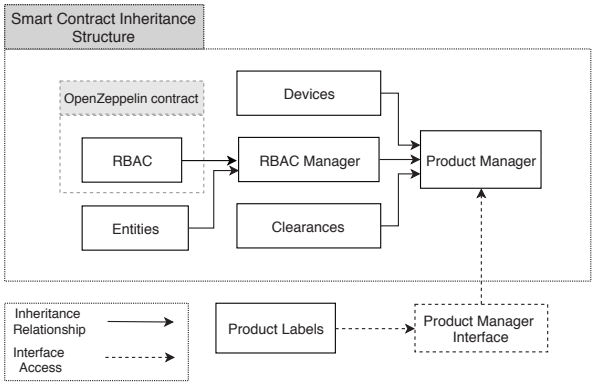

Fig. 3: Smart contracts implemented through inheritance system, or interfaces.

To sum up the functions contained in each smart contract module:

- **RBAC and RBAC Manager** - RBAC, taken from an open-source library (OpenZeppelin [17]) contains the basic methods to give users roles and perform authentication checks. RBAC Manager uses this code to implement the roles wanted for our logistics system (see Section III-B).
- **Entities and Devices** - Smart contracts for storage and management of user and device data, respectively.
- **Clearance** - Contains the methods and structures used to evaluate quality control and clearance of products.
- **Product Manager** - The most central smart contract to the system. It is where the product class and its associated tracking and clearance management is implemented, along with RBAC authentications and permissions.
- **Product Labels and Interface** - Contract containing the data structures and methods for storing different standards of label data associated to products. This smart contract is launched separately to our main smart contract system, and interacts with it via an interface.

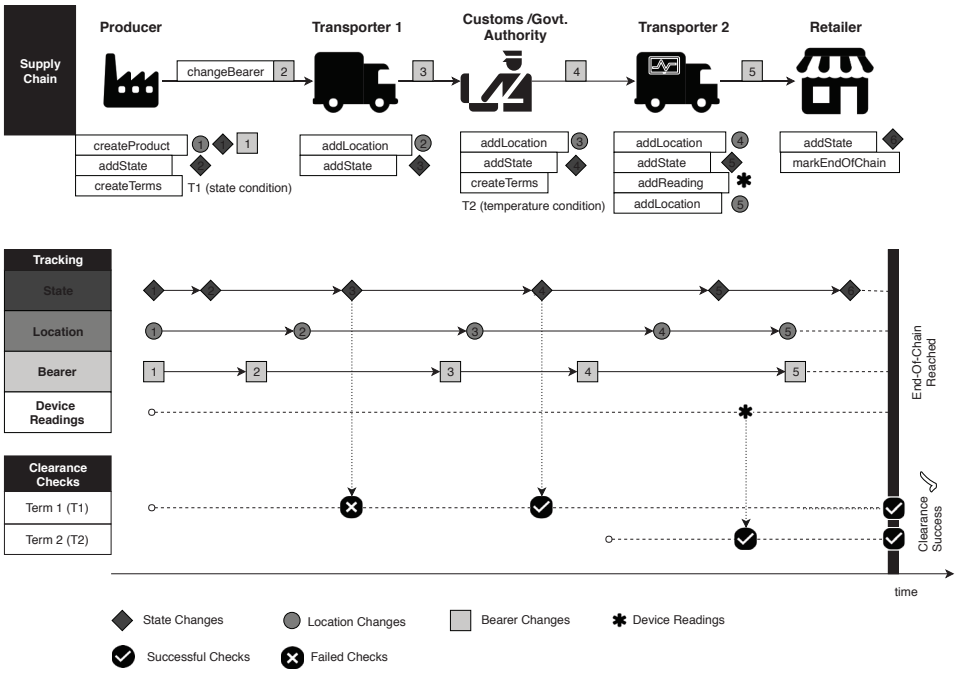

Fig. 4: Representation of the tracking system and the methods implemented, showing an example of a product's journey through the supply chain and a timeline of events.

B. User Interaction

Figure 4 shows how the tracking of a product unfolds as it travels through the supply chain in three different views:

- On the supply chain level, we see what methods each bearer entity uses to introduce data into the blockchain. Changes in state, location or bearer are marked by a diamond, circle and square shape, respectively.
- On the tracking level, we see how the data introduced by the bearers can be used to reconstruct a timeline of events for changes in state, location and bearer, as well as any readings received from IoT devices.
- On the level of clearance checks, we see the terms of transport laid out by the producer or government authorities. We also see at what points the data introduced into the blockchain was checked against these terms, and whether it fit the quality requirements they asked.

In this example, the producer starts out by introducing the product data and its location into the system with `createProduct`. In this way he becomes the first bearer and the point of origin for this item. He updates the product's state accessing the `addState` function, and creates a term for clearance, T1, related to its state - in this case, a requirement for the product to pass through a certain state before it arrives to the EOC. The producer then uses the `changeBearer` method to pass the product on to Transporter 1.

The products tracking data continues being updated, and state changes checked to see if they fit the requirement of T1. A government authority eventually becomes the products bearer and makes a state change that fits the requirement for term T1. He also creates a new term for transport, this time a temperature requisite. As the product changes hands to Transporter 2, he uses the blockchain to send an order to an IoT temperature sensor to start registering temperatures. At the end of the journey, he sends an order to stop reading, and the values registered by the device are inserted into the blockchain as a Reading object. This data is checked against term T2 and turns out to fulfil its requirements.

As the product reaches the end-of-chain, it has passed all its clearance checks. A record of its journey exists in the blockchain in the form of transaction logs for state, location, and bearer changes, as well as device readings, certifications (not represented here) and terms of transport fulfilled. This information can be used to reconstruct the timelines seen on the lower half of Figure 4. It can be read from the blockchain using view type functions, with no cost to the users who call them. These user interactions can happen through a browser application, a topic we discuss further in Section V-B.

C. IoT devices

When considering the implementation of IoT devices into our project, first, we must address the issue of where to run our blockchain client. Most IoT devices, due to energy consumption and cost constraints, are designed to have low processing power. However blockchain clients are heavy applications that require large amounts of storage and processing power to function. Consequently, currently the most straightforward way to connect an IoT device to a blockchain is through a gateway unit that runs the blockchain client and communicates with the IoT device via another protocol.

Secondly, as we have mentioned previously, each device has its own user account or private and public key pair to interact

with the blockchain. We must then consider where we store the private key, which is used to sign transactions and spend the account's currency. Knowing that handling transactions is demanding in terms of processing power, and that storing a private key in the device itself might prove unsafe when there's no access control on its memory storage, we chose to use a blockchain wallet [16] software in the gateway unit itself, which can enforce better security for this key.

Lastly, our application needs to know when it is supposed to send readings to the blockchain, of what type and on what products. To this end we implement the event log feature of Ethereum smart contract technology, so that the owner of a product can remotely send orders to start/stop reading values and to add them to the blockchain. This process unfolds in the following way:

1) A government authority creates a term for storage of an item. When a warehouse receives it, becoming its bearer, it accesses a method in the Product smart contract that fires a `StartReadingOrder` event for one of its devices, (if all bearer and ownership permissions are met).
2) The gateway unit is listening to the blockchain for events sent to devices connected to it. When the order to start reading is received, our application starts storing the device reading information that was requested, and fires its own `OrderReceived` event, so the bearer can know its request is being attended to.
3) Before the product is shipped to another location, the bearer fires a `StopReadingOrder` event. Upon receiving it, the gateway unit finishes its stops its reading process and registers the data collected on the blockchain, under that product's tracking data. It is automatically checked by the smart contract against the terms for clearance created.

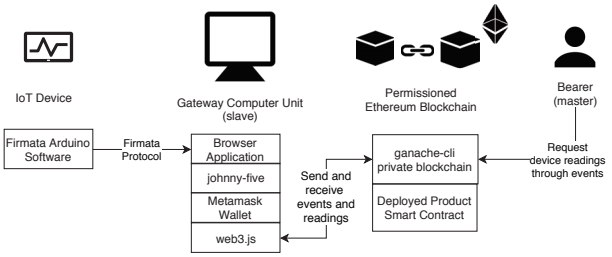

Fig. 5: Diagram illustrating how we collect sensor data from an Arduino and send it to the blockchain.

Figure 5 illustrates the connection made between the device, a gateway unit, blockchain and the device's bearer, including the software we used to bridge the communication gap. We decided to use an Arduino UNO microcontroller and an Adafruit SHT31 temperature and humidity sensor as the data collector. The Firmata protocol [18] is used for microcontroller - computer software communication. The johnny-five library interprets the pin output received via Firmata. The bridge between device and blockchain is made through a browser application in the gateway device. It was made with Node.js and uses the dweet.io [19] and socket.io [20] libraries emits this data in real-time, whilst the web3.js [16] library connects it to a private Ethereum blockchain, where our smart contract is deployed, and uses the JSON-RPC protocol to communicate. Metamask is used to manage the device's user account and transaction signing. We discuss our implementation further in V-B.

V. Results

In this section we will analyse tests we made to our smart contracts and the applications we developed to interact with them.

A. Javascript Tests

We tested out the most important methods and functionalities of our smart contracts using Truffle and a local blockchain generated using ganache-cli.

For testing the RBAC system what these tests did essentially was have a registrar user attribute different roles to a number of other user accounts. These user accounts were then used to test the several different methods that altered the contract's state. It was verified that indeed if the user calling certain methods did not have the appropriate role or permission to do so, the blockchain network would reject his transaction.

Afterwards, view type functions (which access state variables, but do not alter them or execute transactions) were used to retrieve values from the blockchain system, and verify if the smart contract's state was being changed as expected, overall simulating a scenario such as the one seen in Fig.4.

Our tests as a whole were successful and showed that data was being stored as expected and state changes and clearance were being processed properly. RBAC features also functioned correctly, with the blockchain rejecting transactions whenever accounts that weren't bearers or didn't have a specific role tried to access non-authorized functions.

B. User & IoT Browser Applications

We built two simple browsers applications, one meant for regular bearer users to access information on the blockchain, and another to bridge the communication between IoT devices and our smart contract system.

The first browser application we built uses web3.js and Metamask [16]. For the user account logged into Metamask, the application retrieves the tracking data of the products in their possession, and displays it in the form of a timeline of tracking changes, as seen in Figure 6.

The second application we built manages communication between IoT devices and the blockchain. Its functioning and implementation were already discussed in Section IV-C. In the application shown on Figure 7, we monitor our local blockchain for events that order the start or end of readings for the device logged into Metamask, sending the data required upon receiving the stop request. We found that using Metamask as a wallet, although safe, brings the inconvenience of having to confirm every transaction made to the blockchain,

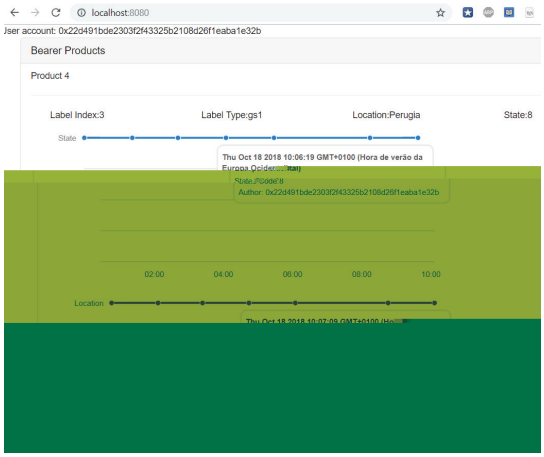

Fig. 6: Product tracking viewer application.

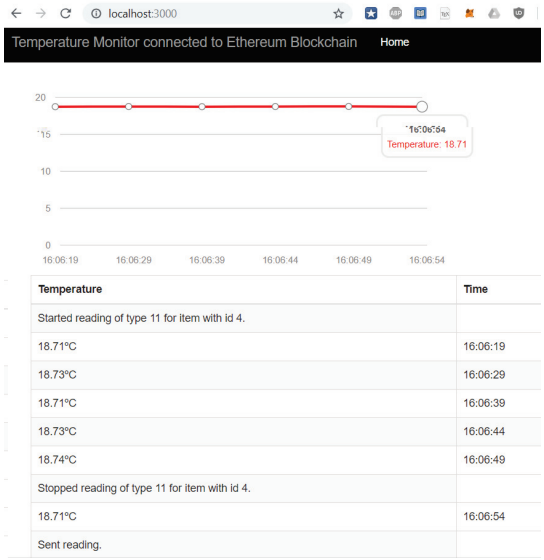

Fig. 7: Blockchain to IoT bridge application.

a function we would like to eventually automatize by using a custom type of wallet.

VI. Conclusions

With this paper we have presented the design and implementation of a blockchain smart contract system geared towards management of products in a logistics system. We have successfully implemented a number of different features to this end - namely Role Based Access Control, product tracking and tracing and semi-automatic clearance procedures with the aid of IoT device input. Our solution showcases the potential of blockchain technology in solving some of the issues facing the logistics industry, whilst taking advantage of the decentralized character of these systems.

We have found, however, a number of limitations to applying these solutions to real-world use. While the technology has developed to allow complex applications to built on top of it, there are still large restrictions to the usage of these systems. Difficulties in managing scalability, and the expensiveness of storage, are some of the core problems we have found. Integration with IoT devices is made difficult due to the processing power requirements of blockchain clients, as well as privacy concerns and an overall shortage of tools and resources to this end. Many developments are being made towards fixing these issues, however, so it is our belief that the viability of system such as the one we have presented here will become a reality in the near future.

In future work, we would like to implement communication of the smart contracts seen in Section IV-A completely via interfaces, so that they could be deployed separately and used in networks that are restrictive in regards to contract size (such as the public Ethereum blockhain). We would also like to implement the communication of more types of IoT sensors with our blockchain, and explore the benefits of their usage in supply-chain processes by testing our system in real use cases.

References

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.
[2] R. Beck, “Beyond bitcoin: The rise of blockchain world,” *Computer*, vol. 51, no. 2, pp. 54–58, 2018.
[3] T. Felin and K. Lakhani, “What problems will you solve with blockchain?” *MIT Sloan Management Review*, vol. 60, no. 1, 2018.
[4] G. Wood, “Ethereum: A secure decentralised generalised transaction ledger,” *Ethereum project yellow paper*, vol. 151, pp. 1–32, 2014.
[5] M. Petersen, N. Hackius, and B. von See, “Mapping the sea of opportunities: Blockchain in supply chain and logistics,” *Information Technology*, vol. 60, p. 263–271, 2018.
[6] “Maersk and ibm introduce tradelens,” http://newsroom.ibm.com/2018-08-09-Maersk-and-IBM-Introduce-TradeLens-Blockchain-Shipping-Solution, (Accessed on 08/9/2018).
[7] “Blockchain: the solution for supply chain transparency | provenance,” https://www.provenance.org/whitepaper, 2015, (Accessed on 02/17/2019).
[8] “The latest blockchain supply chain startups,” https://medium.com/applicature/the-latest-blockchain-supply-chain-startups-7b0e5c07548f, (Accessed on 02/16/2019).
[9] H. R. Hasan and K. Salah, “Blockchain-based solution for proof of delivery of physical assets,” in *International Conference on Blockchain*. Springer, 2018, pp. 139–152.
[10] F. Tian, “An agri-food supply chain traceability system for china based on rfid blockchain technology,” in *2016 13th International Conference on Service Systems and Service Management (ICSSSM)*, June 2016, pp. 1–6.
[11] Y. Yuan and F. Wang, “Towards blockchain-based intelligent transportation systems,” in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 2663–2668.
[12] K. Christidis and M. Devetsikiotis, “Blockchains and smart contracts for the internet of things,” *IEEE Access*, vol. 4, pp. 2292–2303, 2016.
[13] M. Pustišek and A. Kos, “Approaches to front-end iot application development for the ethereum blockchain,” *Procedia Computer Science*, vol. 129, pp. 410–419, 2018.
[14] “Consortium chain development ethereum/wiki,” https://github.com/ethereum/wiki/wiki/Consortium-Chain-Development, (Accessed on 02/03/2019).
[15] “Trade and transport status codes,” https://www.unece.org/fileadmin/DAM/cefact/recommendations/rec24/rec24_ecetrd258e.pdf, (Accessed on 01/27/2019).
[16] A. M. Antonopoulos and G. Wood, *Mastering ethereum: building smart contracts and dapps*. O’Reilly Media, 2018.
[17] “Get started · openzeppelin,” https://openzeppelin.org/api/docs/get-started.html, (Accessed on 12/20/2018).
[18] “Github - firmata/protocol: Documentation of the firmata protocol.” https://github.com/firmata/protocol, (Accessed on 03/04/2019).
[19] “dweet.io - share your thing.” https://dweet.io/, (Accessed on 03/04/2019).
[20] “Socket.io,” https://socket.io/, (Accessed on 03/04/2019).