Combining Blockchain and IoT: food-chain traceability and beyond

energies

MDPI

Article

Combining Blockchain and IoT: Food-Chain Traceability and Beyond

Jacopo Grecuccio [1], Edoardo Giusto [1,*], Fabio Fiori [2] and Maurizio Rebaudengo [1]

[1] DAUIN, Politecnico di Torino, Corso Duca degli Abruzzi, 24, 10129 Torino, Italy; jacopogrecuccio1994@gmail.com (J.G.); maurizio.rebaudengo@polito.it (M.R.)
[2] Foodchain S.p.A.,Via Cavour, 2, 22074 Lomazzo, Italy; fabio.fiori@food-chain.it
* Correspondence: edoardo.giusto@polito.it; Tel.: +39-011-0907068

Received: 20 June 2020; Accepted: 21 July 2020; Published: 25 July 2020

Abstract: Recently, the interest around the Blockchain concept has grown faster and, as a consequence, several studies about the possibility of exploiting such technology in different application domains have been conducted. Most of these studies highlighted the benefits that the use of the blockchain could bring in those contexts where integrity and authenticity of the data are important, e.g., for reasons linked to regulations about consumers' healthcare. In such cases, it would be important to collect data, coming in real-time through sensors, and then store them in the blockchain, so that they can become immutable and tamper-proof. In this paper, the design and development of a software framework that allows Internet-of-Things (IoT) devices to interact directly with an Ethereum-based blockchain are reported. The proposed solution represents an alternative way for integrating a wide category of IoT devices without relying on a centralized intermediary and third-party services. The main application scenario for which the project has been conceived regards food-chain traceability in the Industry 4.0 domain. Indeed, the designed system has been integrated into the depiction of a use case for monitoring the temperature of fish products within a warehouse and during the delivery process.

Keywords: internet of things; blockchain; distributed ledger technologies; smart societies; industry 4.0; supply chain; food traceability

1. Introduction

During the last ten years, the industrial processes commonly applied in any business field, from manufacturing to agriculture and many others, have been subject to a radical evolution led by the arising of new technologies interconnecting the digital and the physical world and making up the Cyber-Physical-Systems (CPSs). CPS consists of a system of collaborating computational elements controlling physical entities, in which mechanical and electronic systems are embedded and networked using software components. They use a shared knowledge and information of processes to cooperate for accomplishing a specific task [1,2]. The trend of embedding interconnected electronic devices, capable of interacting with the environment in which they live in and enabled to communicate over the Internet, has spread not only in the industrial and manufacturing fields but also in other business sectors such as agriculture, breeding, and food transformation processes [3]. The resulting scenario is known under the context of Internet-of-Things (IoT), which can be defined as a digital overlay of information over the physical world. Each object (referred to as a *"thing"*) connected in such a network is uniquely identifiable and able to sense and react with the environment, as well as with other users or objects [4,5]. The IoT devices market nowadays represents a huge portion of the whole Information-Technology (IT) and electronic devices market with more than $235 billion spent in 2017, and it is expected to double in 2021, growing up to $520 billion [6].

Besides IoT, another technology arose and has grown even faster: the Blockchain. A Blockchain (BC) is a distributed ledger based on a peer-to-peer (P2P) network, in which participants, called *nodes*, agree on a unique version of the distributed data storage through a shared consensus mechanism. All information stored inside the ledger is digitally signed employing cryptographic primitives and data-authenticity is guaranteed by the use of asymmetric key-pairs. The first definition of BC was given in November 2008, in a white-paper titled "Bitcoin: a Peer-to-Peer Electronic Cash System" sent to the subscribers of a mailing list about cryptography by a mysterious author known under the pseudonym of Satoshi Nakamoto [7]. After the advent of Bitcoin, several different implementations of Blockchain have been proposed, targeting not only crypto-currency and finance application but also business processes automation and certification [8]. Several studies nowadays are exploring the possibility of applying the BC technology for product certification processes, especially in the food industry, and most of them believe that this technology would represent a powerful partner for solving problems related to consumer's trust in food products and to implement a traceability system able to find possible sources of contamination in-time, thus reducing the risks for citizens' health [9].

In this paper, a possible integration strategy that combines both IoT and BC for improving business processes is presented. The proposed architecture is able to fully exploit the concept of a public Blockchain. In fact, IoT devices are able to directly sign transactions and store them in the BC in such a way that there is no third party involved in the process, which could lead to a series of undesirable situations. The final result of this study is a *blockchain-enabled gateway* for IoT devices to be integrated in an Industry 4.0 domain and in many other scenarios in which the combination of both technologies brings advantages to both business processes and customer satisfaction. In particular, the selected use-case in which the architecture has been applied and tested is related to the agri-food supply chain, also known as *food-chain*.

The remainder of this paper is organized as follows. Section 2 gives a high-level overview of the Blockchain's basic elements with a specific focus on Ethereum in Section 2.3. Section 3 describes the current landscape of IoT-BC integration, highlighting its main weaknesses. In Section 4, the proposed integration strategy is presented, along with implementation details in Section 5. Section 6 depicts a simulation of a use-case application related to the cold-chain monitoring, also describing the main issues related to the agri-food supply chain. Section 7 displays and comments the results obtained with the proposed architecture. In Section 8, conclusions are drawn, also proposing possible extensions.

2. Background

The decentralized network proposed by Nakamoto [7] is based on a Peer-to-Peer architecture, where all the nodes have the same functionalities and provide the same services without the distinction of roles as it happens in a Client–Server architecture. Each peer in the network, known as a **node**, stores locally a copy of all the history of the **transactions** published on the network. A Blockchain can be formally defined as a structured database of **blocks**, each one containing several transactions, linked together by including in each block the cryptographic hash of the prior block in the chain. This allows us to check the integrity of the whole chain going backward until the genesis block, which is the very first block of the chain. Sometimes separate blocks can be produced concurrently leading to the creation of a temporary fork of the chain. To solve this problem, any BC defines an algorithm for scoring different versions of the history so that only the one with the highest score is considered valid and selected over the others. The mechanism used for verifying new transactions to be added to the chain is called **mining**, and not only makes the users agree on a unique version of the chain but at the same time allows for create new coins that can be spent in the network, exploiting a **reward mechanism**.

2.1. Blockchain Principles

As it appears from the previous section, the fundamental elements upon which a Blockchain is built are blocks and transactions. The need for each node of the network of storing locally the entire chain implies the usage of an efficient data structure that has to occupy a small quantity of memory and at the same time to guarantee that the data stored in the chain are immutable and tamper-proof. In order to be scalable, in Bitcoin BC, each block's hash is the hash of its **block-header** (a data structure containing a timestamp, a nonce and the hash of the previous block) and the root hash of a multi-level data structure known as Merkle-Tree. A **Merkle-Tree** is a binary tree composed of a set of nodes with a large number of leaf nodes containing the underlying data at the bottom. Starting from the leaves each intermediate node stores the hash of its two children up to the root node that is at the top of the tree. The main purpose of this data structure is to allow for retrieve the required data efficiently and eventually piecemeal from different nodes in the network. Moreover, the hash mechanism behind this structure, combined with the shared consensus, ensures that the entire chain can not be changed or altered. However, nowadays the effectiveness of this protocol is argued not to be sustainable for the long-term. Indeed, as of June 2020, the total space occupation of the Bitcoin Blockchain is roughly 270 GB and it keeps growing faster. This factor makes unaffordable the implementation of a node on devices with limited storing and computing capabilities like mobile and embedded devices [10].

2.2. Digital-Signature and Elliptic-Curve Cryptography

Asymmetric key cryptography, also known as **public-key cryptography**, is a cryptographic scheme that makes use of a pair of large numbers (*keys*) that are somehow related together but they are not identical. One key, referred to as *private* because it has to be known only by the owner and not shared with any other parties, is used for decrypting a message over an insecure channel. Instead, the other key of the pair referred to as *public* because the owner shares it with other parties, is used by the message's sender for encrypting the message to be sent to the other party. It appears clearly that such mechanism works under three fundamental assumptions:

- Private- and public-key should be related, but at the same time it should be unfeasible to obtain the private key knowing only the public one,
- The decryption function gives the correct result if and only if the correct private key (i.e., the one related with the public key used for the encryption) is given and for no other different value,
- Private keys should be kept secret and not shared with any other parties.

Therefore, asymmetric cryptography protocol implementations differ mostly in the algorithm used for the key pairs generation and the encryption/decryption functions. For example, the RSA algorithm is based on the *large integer factorization problem* and, simply speaking, uses the product of two large prime numbers as part of the public key, and derives the private key from the same number as well. The encryption strength relies mostly on the key-size and its robustness increases exponentially as such parameter grows. Actually, RSA [11] keys can be typically 1024 or 2048 bits long, but experts believe that RSA 1024 is near to be cracked and it should not be considered secure anymore, especially with the advent of quantum computing [12]. Another implementation of the public key cryptography's protocol relies on the algebraic structure of elliptic curves over finite fields and is known as Elliptic Curve Cryptography (ECC). It is based on the *elliptic curve discrete logarithm problem* that consists in finding the discrete logarithm of a random elliptic curve element concerning a publicly known base point. The approach of using elliptic curves in cryptography was proposed for the first time in 1985 by Neal Kolbiz [13] and Victor Miller [14], but it became widely used starting from 2004. It is used in Blockchain implementations such as Bitcoin and Ethereum as a Digital Signature scheme [15,16]. The security of ECC stands in the fact that, roughly speaking, given a point $k * G$ on the curve it is difficult to go back to the original value of k, this is known as the **Discrete Logarithm Problem**, which is said to be one of the NP-hard problems, and, compared to RSA, ECDSA offers the same level of security but with smaller keys.

2.3. Ethereum

Ethereum is an Open-Source project aiming at the creation of a public BC for distributed computing. It was born in 2013 thanks to Vitalik Buterin, a Canadian developer and researcher in the field of cryptocurrencies. Through a crowdfunding campaign, he was able, in 2014, to implement his idea, published and available online the following year [17].

The project intended to create an alternative protocol for building decentralized applications (DApp), allowing a different set of trade-offs to be used in those applications where rapid development time, security and the interaction between different distributed applications or systems is crucial. Following this idea, Ethereum provides a fundamental layer composed of a Blockchain and a Turing-complete programming language so that anyone can write its own DApp, implementing its arbitrary rules for ownership, transaction formats, and state-transition functions. The most interesting and explored functionality of this new architecture is one of the so-called *Smart-Contracts*. *Smart-Contracts* can be seen as cryptographic entities containing a value that can be represented by a certain amount of currency or information or both, unlocked only if an application-defined set of conditions are met. To achieve the goal of creating a simple framework for building powerful decentralized-application, the basic philosophy pursued in Ethereum development is based on a few principles:

- **Simplicity**: the protocol has to be as simple as possible, even at the cost of some inefficiency in terms of execution time or data storage. The idea is that an average programmer should be able to follow and implement the entire specification.
- **Universality**: Ethereum should not provide features. Instead, it has to provide a complete fundamental layer upon which any kind of application can be easily built.
- **Modularity**: the framework has to be structured in modules, as separate as possible.
- **Agility**: the protocol's architecture should not be extremely fixed, and any meaningful further proposal bringing significant benefits or improving scalability and sustainability is accepted.
- **Non-discrimination** and **non-censorship**: the protocol should not attempt to actively restrict or prevent specific categories of usage.

Ethereum's fundamental currency is *Ether* (*ETH*) and is used to pay for *gas*, which represents the unit of computation used in a transaction and other state transitions, such as Smart-Contracts execution.

2.4. Smart-Contracts

The term Smart-Contract (SC) was introduced for the first time in the 1990s by Nick Szabo, a computer scientist and cryptographer, who realized that the Distributed Ledger Technology can be exploited for securing relationships over a network [18]. Bitcoin was the first Blockchain implementation to support a basic form of Smart Contracts through its scripting language, which was however somewhat limited. The Ethereum project, on the other hand, was born with Smart-Contracts in mind and aiming at realizing a framework for DApps running over a BC ledger in a peer-to-peer network. Just as the term suggests, Smart-Contracts are somehow related to "normal" contracts, with the difference that the latter ones are signed by partaking entities and enforced by the law, whereas the former ones are digitally signed and define relationships among parties exploiting cryptographic mechanisms. Basically, a Smart-Contract is a self-executing application that runs on top of the BC and is therefore immutable. It can be seen as a complex *if-then* statement that is executed if and only if a set of conditions is met. Smart-Contract can either produce an output or even trigger the execution of another Smart-Contract for performing a sophisticated task. Because of their decentralized nature and the possibility to exploit BC's features for handling trust-less contexts, SCs are becoming an attractive topic for various kinds of business fields spreading from finance, insurance, manufacturing, and many others.

To better understand how SCs work and how big their potential can be if exploited in some applications domain, it is worth describing an example use case (Figure 1). Let's assume that farmer

A sells tomatoes to company B which then uses them for producing tomato sauce. The transfer of tomatoes from A's farm to B's warehouse is in charge of a third delivery company C. The actual most common scenario would be the one in which A, B, and C have their own digital business layer, generally made of a centralized data storage (in a local Database or using some cloud services) and their own software products that deal with such data structure for registering shipping, invoices, inventory and so on. Moreover, this software may differ between one company from another, making it difficult to automatize their cooperation. Let us assume now that the same business scenario was managed through the use of Smart-Contracts and a Blockchain in a Peer-to-Peer network at which all of the three companies participate. A smart DApp developer would implement three Smart-Contracts able to interact together:

- A first Smart-Contract that can handle the transfer of ownership of the batch of tomatoes from A to C, ensuring that the latter would be responsible for any loss or damage during shipping.
- Another Smart-Contract that can handle the transfer of ownership from the delivery company to recipient B.
- A third Smart-Contract, triggered by the previous one that handles payments between the parties in something similar to the following statement: *If tomatoes are delivered without any damage a certain amount of funds will be transferred to both the selling and shipping companies. If tomatoes are damaged then company C has to refund the selling company, and so the proper amount of crypto-money will be transferred from C's account to A's one.*

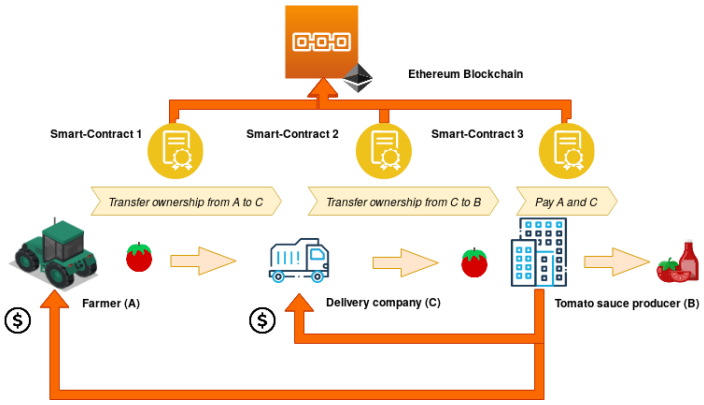

Figure 1. Application of Smart Contracts to traceability in the food-chain domain (image created using https://draw.io).

Thanks to the BC, all operations occurred between the parties will be registered in the ledger forever, and so it is straightforward to think to more complex mechanisms to be built upon the simple scenario described above (insurance for the delivery process, supply-chain traceability, etc.).

2.5. Quadrans: The Industrial Blockchain

The Quadrans [19] platform is an open-source and public Blockchain network that runs Smart-Contracts and decentralized applications. The first implementation of this platform, born in 2012 as a fork of Ethereum, has been conducted under the Foodchain S.p.A. brand, and its initial goal was to enable traceability, transparency, and authenticity of the information within agri-food supply chains. In August 2018 Quadrans Foundation was officially formed and acknowledged by Swiss local authorities. The goal of the foundation is to guarantee continuous advancement in technology in an ethical and publicly open way. Quadrans' infrastructure is public, and its open-source blockchain has been designed to overcome scalability issues and to reduce the instability of operational costs affecting other existing blockchains. The current protocol, used by Quadrans, ensures a high throughput (average block-time is about 5 s) that can maintain at the same time both high security and decentralization grades and supports all the core Ethereum's features, such as EVM (Ethereum Virtual

Machine) and Smart-Contracts. Having reached these goals, Quadrans represents a valuable solution for effectively applying blockchain technology in many business scenarios.

3. Literature Review

Having given enough details on the basic principles of BC's technology, it is possible to introduce the real goal of this article. As stated in the introduction, the project aims to find a new method of interaction between the BC infrastructure and the world of internet-connected devices.

The most common solution applied for this scenario consists of a Client–Server communication (as in Figure 2) between a central server and the IoT devices. The server is in charge of (*a*) gathering the data streams collected from the field and (*b*) storing these data, or part of them, in the Blockchain. Even if this solution is starting to be applied in different proof-of-concept and business applications, this has some point of weakness:

- The central server may become a single-point-of-failure that can inhibit some data to be stored in the BC when it is in a fault condition,
- Assuming that redundancy is applied, , in order to minimize the probability of failure, there is still the presence of some kind of centralization that makes difficult the interaction between cooperative business parties,
- The data which is then sent to the BC is not signed-in-place, a way in which its authenticity and integrity can be guaranteed to start from the source, but it is signed only when it is received by the central server before posting it into the Blockchain,
- Nevertheless, such kinds of systems are now mostly realized using cloud solutions for the IoT field. Experts believe that the cost of such services is going to increase rapidly and to become a significant component of business costs for most companies.

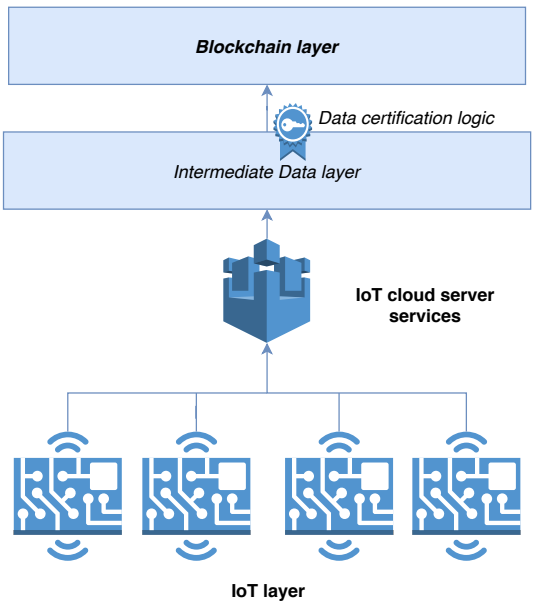

Figure 2. The common architecture used for storing data collected from IoT devices into the blockchain (image created using https://draw.io).

The surveys in [10,20] present extensive study of these and other issues. The authors of [20] describe the issue of storage size in relation to an application in the IoT field. The solution we present solves this issue by implementing light-nodes that act as a gateway. These light-nodes only perform the tasks of signing and sending their own transactions. They do not participate in the consensus mechanism (there is no mining), and thus they do not have to locally store the whole blockchain.

A relevant paper on the quest for decentralization, secure data storage, and IoT compatibility can be found in [21]. This paper describes a model to handle data-streams coming from IoT devices with a decentralized access mechanism control. This solution has some undesired aspects though. It uses an intermediate level between the cloud and the BC, meaning that the data are not set directly to the BC by the device. Moreover, it uses the Bitcoin BC, even if in its VirtualChain version, supposedly lighter and faster.

The solution described in [22] seems focused mainly on performing some kind of action based on the conditions of Smart-Contracts on the Ethereum Blockchain. It has interesting propositions regarding the use of smartphones to set/update conditions for Smart-Contracts, and the possibility of IoT devices to act according to these conditions. In addition, some pieces of Smart-Contracts are shown in the paper. Despite these characteristics, the whole architecture is not clearly described, especially for what concerns the communication with Ethereum blockchain, where the blockchain itself resides, and the signing functionality of the IoT device.

In [23], BPIIoT is presented. The authors refer to it as a key-enabler for Cloud-Based Manufacturing, enabling peers of a peer-to-peer decentralized network to interact among them without a trusted intermediary. They also implement Smart-Contract handling on a BeagleBone single-board computer (SBC) which seems to be connected to an Arduino board for sensing capabilities. However, this paper also does not give many details about the implementation of such system, especially regarding the connection with the Ethereum blockchain and the signing capabilities.

Authors in [24] present AgriBlockIoT, a decentralized traceability system for the Agri-Food supply chain management. AgriBlockIoT is nicely organized in layers and can support either Ethereum or Hyperledger Sawtooth. The paper also presents an interesting *from-farm-to-fork* use case, depicting all actors, materials, and tasks involved in such a process. Our approach can be considered an evolution of this, since we remove the need of running a full node of the blockchain in the stakeholder's local network. Moreover, the use of IoT devices is only simulated and not practically implemented.

A blockchain-IoT-based food traceability system (BIFTS) is proposed in [25]. Not only it does integrate BC and IoT, but it also proposes an appealing quality decay evaluation of perishable goods based on fuzzy logic. A critical point in this architecture is the fact that, to get to the BC, the data have to pass through a cloud server. This represents in any case some sort of centralization, which we would like to avoid. In fact, it is not possible, inside the cloud, to authenticate the data with respect to a certain device.

In [26], authors present a blockchain-based fair nonrepudiation service provisioning scheme for industrial IoT scenarios. It is a complex and interesting framework in which the BC acts as an evidence recorder and a service publication proxy. A limitation of this proposal is that their devices communicate through a BC node, like the first intermediate solution proposed later in Section 4.

The perspective presented in [27] is really compelling. Their solution features the direct signing capabilities of devices, which is one of our aims, and the offloading of communication with the BC to a proxy service. The framework is built in such a way that the proxy server itself cannot forge the devices' signatures, thus maintaining the authenticity of the data. In addition, the applicability to the cold-chain use case is the same as our approach. However, their solution relies on the use of Hyperledger Fabric as a permissioned BC, which is not public by definition. A permissioned BC implies the presence of a certain authority issuing credentials and certificates, which is indeed a form of centralization. Moreover, the SDK proposed does not seem ready yet to target very constrained IoT devices built upon a microcontroller.

The advantages of the proposed solution are the following:

- Implemented devices do not perform mining and do not need to store the whole BC, making this solution suitable for low-end hardware (low-power and low-cost),
- The use of a BC (Quadrans) directly derived from Ethereum and natively supporting smart-contracts, with average block times of around 5 s,
- IoT and BC levels communicate directly, without a cloud intermediary,

- Every device has an associated *wallet*, meaning that it is possible to guarantee principles of data-authenticity and non-repudiation since the device itself signs the data before sending it to the BC.

4. Proposed Architecture

The proposed architecture is intended for solving the weaknesses presented above, and possibly to introduce some improvements to the described context. It consists of enabling commercial or ad hoc Internet-of-Things devices to communicate directly with the BC infrastructure through a *gateway* device to which they are connected using wired or wireless protocols. Such a gateway device should be able to sign-in-place the data sent from the IoT data-logging device and then to directly push on the BC. The most straightforward method of doing so would be to let the gateway be a full-node of the chain (i.e., a node in the peer-to-peer network that can store and verify the whole chain and keep it updated too). However, this is unfeasible as the target application context lies in the embedded-systems domain, which commonly consists of limited computing capability devices, for sure equipped with not enough storage to maintain an entire copy of the chain.

A first intermediate solution (Figure 3) that has been explored is based on an architecture where the IoT layer communicates directly with an Ethereum full-node placed within the same Local-Area-Network in which the devices are connected. Such a local node is then in charge of signing and broadcasting the transactions requested by the devices, as well as keeping their private key storage and mappings (between IoT device ID and private key). The implementation of this solution resulted in being fast and straightforward because the only part on which one has to deal with requires only the communication mechanism between the local BC node and the IoT devices, defining the data-model of the message to be exchanged and eventually some encryption mechanism to secure the communication channel. However, this solution suffers from the same problems related to centralization and security as the common case previously described. Nevertheless, it would only be worse if no mechanism for device authentication was implemented for the communication with the local full node.

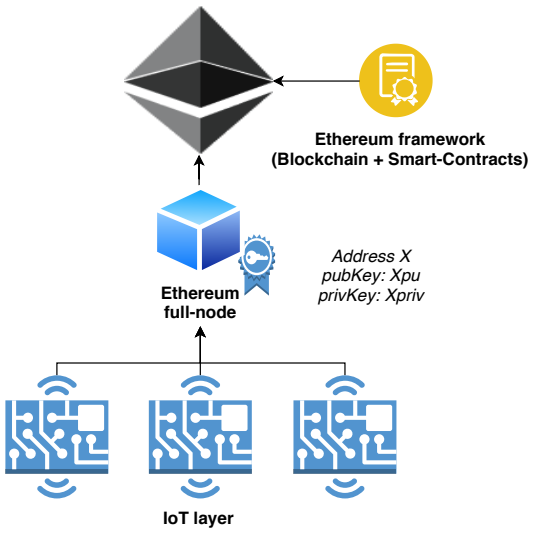

Figure 3. A representation of the first explored solution. IoT devices communicate with the local full node which is in charge of signing and broadcasting transactions for them. However, the IoT layer remains agnostic of the existence of the Blockchain. It simply sends data to a local centralized entity (image created using https://draw.io).

Therefore, the path followed in the project exploits the Remote-Procedure-Call RPC protocol [28] for enabling an embedded device (acting as a gateway) to trigger the execution of some procedure on a remote server exposing such service. Such a device has to be physically placed in the very near proximity of the IoT device (e.g., as expansion hardware for the device) or even be the IoT

device itself. The RPC server exposes to calling clients (i.e., IoT gateways) the Web3 Ethereum API (Application Programming Interface that allows for interacting with Ethereum's BC, so that any embedded device can access the information stored in the chain, exchange coins (e.g., for implementing machine-to-machine payments) and call the execution of Smart-Contracts for implementing complex logic on top of the blockchain layer. Moreover, exploiting the events mechanism of Smart-Contract [29], and the possibility to search for those events in the chain by selectively download block headers through RPC calls, it could even be possible to trigger IoT devices directly within distributed applications and keep the history of such requests. To apply this solution, however, there should be one strict requirement above the others: *the gateway must be identifiable on the networks through its address, and should also be able to sign the transaction locally and offline, using its private-key, before sending it to the RPC server*. This functionality is essential to avoid the possibility that the transaction is maliciously manipulated when it is sent to the server.

The final overall proposed architecture is shown in Figure 4.

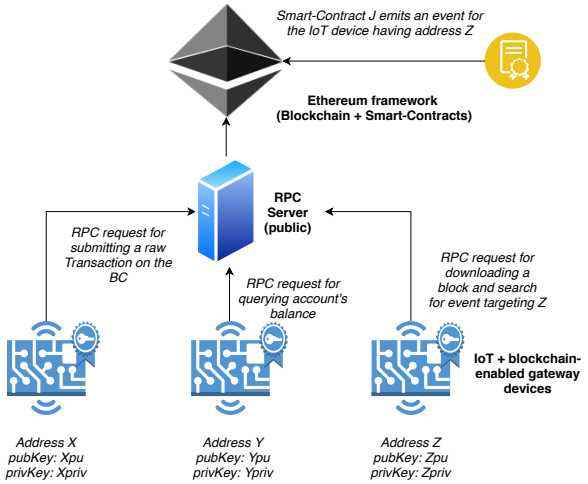

Figure 4. A representation of the final proposed architecture. Each IoT device has its own gateway and can sign transactions locally and offline. Each of them is also identified within the blockchain through its address and can be thus a target for possible Smart-Contract events (image created using https://draw.io).

5. Software Architecture

Most of the project complexity has been demanded from the software layer, meaning that the vast majority of the operations are performed by the CPU without any additional special-purpose peripheral or hardware accelerator.

The developed software framework is intended to be as easy and user-friendly as possible. In this way, it can be used by the majority of IoT developers, with minimal knowledge of the details behind the Blockchain architecture. Following this philosophy, the resulting framework can be represented as a layered architecture (Figure 5) where the top-most software components are those providing the interaction services to the common Internet-of-Things application.

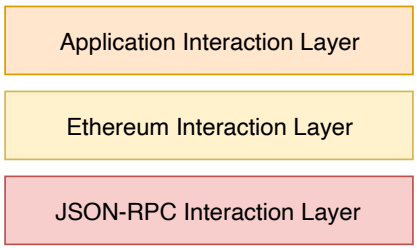

Figure 5. Representation of the Software Architecture (image created using https://draw.io).

5.1. The JSON-RPC Layer

The Remote-Procedure-Call is based on the Client–Server model and is used in those contexts where one program wants to request a service from another one, executed in a remote host. An RPC call is a synchronous event requiring the caller program to be suspended until the remote procedure returns its result. An example of this behavior can be found in Figure 6.

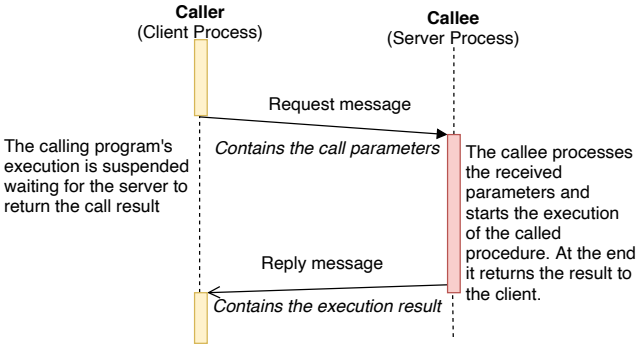

Figure 6. A sequence diagram showing the flow of execution of a remote-procedure-call (image created using https://draw.io).

JSON-RPC represents a state-less and light-weight implementation of an RPC. It is an extension of the original RPC protocol, which uses the JSON data-format for remote-procedures, parameters, and results encoding [30].

In the developed framework, JSON-RPC Client is the software component in charge of performing JSON-RPC calls over the TCP socket on which the RPC server is listening. Its implementation consists of a queue, where other tasks push a data-structure for each RPC-Call that they request. This data-structure is used to model the standard fields of a Remote-Procedure-Call and, in addition to those fields, another one is provided to store the current state of the call. Each call, in this implementation, can assume the following logical states:

- `QUEUED`: it is the first state assumed by a call-object. It describes a call that has been pushed into the queue by a caller task and it is waiting to be executed,
- `SENT`: the call has been sent to the RPC server successfully,
- `WAIT_RESPONSE`: the client is waiting for the server response for this call,
- `SUCCESS`: the call has been executed successfully and its result is now available,
- `ERROR`: an error occurred during the call processing,
- `DELETED`: the call has been removed from the queue.

A graphical representation of the state-transition diagram for a call can be seen in Figure 7. The calls queue is accessed in a circular-buffer manner and its dimension, in terms of the number of calls, is static and does not rely on dynamic memory allocation. The module's user can then set the dimension, according to the application requirements, through a C-language `define` pre-compiler instruction and therefore select the optimal trade-off between memory occupation and performance. Summarizing, the `JSONRPC_Client` interface provides functions for the upper layers , allowing the caller to:

1. Push a new RPC call in the queue,
2. Retrieve the status of a call previously submitted,
3. Retrieve the result of an executed call,
4. Delete a call,
5. Check the queue status (empty/full).

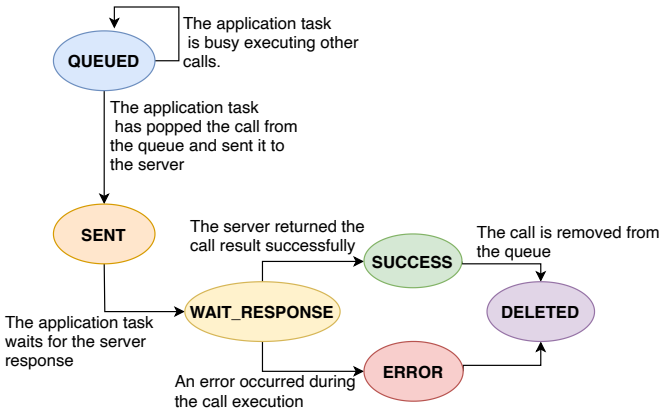

Figure 7. The state-transition diagram for a call-object (image created using https://draw.io).

In addition, the module implements the application-task function (`JSONRPC_Task`) in charge of maintaining the queue, converting calls-object into JSON, sending calls over the TCP socket, retrieving the response, and parsing it from JSON. The sequence diagram in Figure 8 illustrates an example of an RPC call execution.

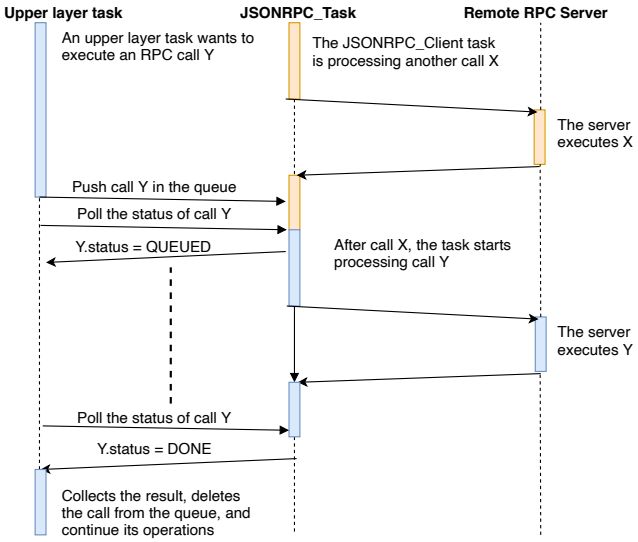

Figure 8. A sequence diagram illustrating how the developed JSONRPC client interacts with upper layers requesting a remote call to be executed (image created using https://draw.io).

5.2. The Ethereum Interaction Layer

The Ethereum interaction layer is one of the frameworks that implements a subset of the functions provided by the Web3 API. Each of the provided functions is in charge of encoding the necessary parameters for the correspondent RPC call to be executed, as well as adapting the returned result into user-friendly C-language types.

The provided functions allow the caller to check the status of an account (i.e., nonce and balance), perform transaction and message calls, and request a specific set of blocks from the chain. The fundamental data-structure of this layer is the `WEB3_ETH_OBJ` that holds the information about the RPC call object as well as the state of the Web3 call. The implementation of these functions follows the state-machine model,with a given web3-call evolving through the following states:

- `CALL`: Parameters are encoded into JSON format and the proper RPC call is pushed into the queue of the JSONRPC client module,

- `WAIT`: Polling on the call-object status to check when the result is ready or an error is returned,
- `DONE`: The result is ready to be processed,
- `ERROR`: An error occurred.

A graphical representation of the state-transition diagram for a Web3 call can be seen in Figure 9.

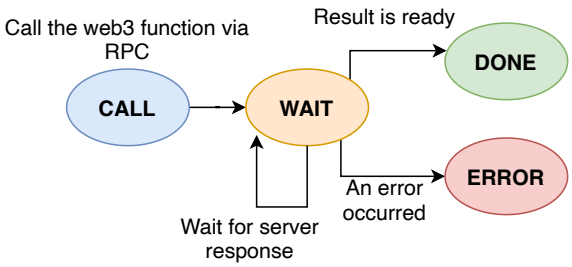

Figure 9. The state-transition diagram for a web3 call in the developed framework (image created using https://draw.io).

5.3. The Application Interaction Layer

The Application interaction layer represents a wrapper of the Ethereum interaction layer in the sense that it hides the details for performing specific operations on the blockchain (e.g., signing a transaction before sending it as raw). It wraps all the functions of the layer below and performs the pre-processing and post-processing operations needed for some of them.

Again, they are internally implemented as state-machines, cycling through the following possible states:

- `WEB3TASK_INIT`: In this state, all the pre-processing is performed before the web3 function is called,
- `WEB3TASK_WAIT_RESPONSE`: The state machine waits for the web3 call to be executed and return its result to perform the post-processing operations,
- `WEB3TASK_IDLE`: Nothing to be done,
- `WEB3TASK_ERROR`: An error occurred.

6. Use Case Application

This chapter presents a use case covering the supply chain of fish and seafood products. The use case has been simulated in collaboration with Foodchain S.p.A., a company whose mission is to enhance the traceability process of food's supply chain through the Blockchain technology.

To simulate the use case scenario, an IoT device has been built and programmed in such a way that it can autonomously communicate with the Blockchain by directly signing its own transactions. Moreover, to corroborate the usefulness of the system, an actual use case scenario has been depicted later in this section.

For what concerns the Blockchain side, Foodchain S.p.A. has developed a platform, based on Quadrans' infrastructure that provides services for both companies and final buyers. Companies can use the platform as a common Enterprise Resource Planning (ERP) software, registering products, batches, and processesand any information related to the product's lifecycle (documents, certificates, media, etc.). Such information is immutably registered into the BC, on top of which the platform lies, using Smart-Contracts. At the end of the product's transformation process, the platform generates some kind of Smart-Label (like a QR code or an RFID tag) that uniquely identifies the product and can therefore be scanned by the product's buyer for viewing the entire traceability process. Another important feature provided by the platform is that it enables many companies and producers to cooperate together within the network, allowing them to follow the transformation cycle of a product even when it passes through the production and distribution chains of different actors.

6.1. Food-Chain Issues

Recent studies have demonstrated that customers' loyalty in food companies and in all the products they sell has drastically decreased year after year. In 2018, the Center for Food Integrity published its annual report about consumers' habits and feelings when buying food. The investigation showed that an increasing number of customers are concerned about what they eat, thus they are asking for more information about the food they buy, to choose the healthy one [31]. However, most of them do not trust the information written behind the product's package, especially when it does not directly come from farmers, but it went through some transformation processes instead. In addition, the awareness of consumers about the damages caused to the environment by some farming and breeding processes is encouraging them to not buy entirely certain categories of products [32].

The lack of trust in food producers is the consequence of a minimal customers' knowledge about the whole supply chain and also the fear of a globalized market that may allow fraudulent producers to sell dangerous or counterfeited products. In particular, this last aspect is critical both for consumers, who are subjected to risks related to their health, and "honest" producers, who not only suffer the economical damages but are also affected by the side-effects of the distrust of buyers. In the year 2018, the Italian Central Inspectorate of Quality Protection and Fraud prevention of food products (ICQRF) has registered an increment of 58%, compared to the previous year, of crimes related to food frauds, seizing about 17.6 tons of goods, for an equivalent value of over 37 million USD. The wine sector resulted to be the most affected by this phenomenon [33].

6.2. The Concept of Food Trust

From the previous analysis, it is clear that consumers' confidence in the agri-food industry is the result of their concerns or certainties on various aspects related to the product they intend to buy. One of the most important factors to restore a good level of reputation of the food industry is to guarantee **food safety**, which is every day undermined by the numerous cases of food hazards. The lack of information about transformations and events under which the final product has gone through during the supply chain poses the basis for a serious alarm about the health of citizens and, despite the efforts of government in control and prevention, the current infrastructure turns out to be unsuitable and inefficient in the fight against this phenomenon. Another important aspect of food trust is related to the concepts of **food-integrity** and **food-authenticity**, which have a significant impact both on the healthcare and the economy of countries. Food-Integrity defines the state of being undiminished and unaltered with respect to its original nature, which means that no unapproved and undeclared transformations and alterations have been made on such products. The Food-authenticity definition, instead, is more related to economical aspects, like frauds that generally do not represent a risk for the health of consumers. However, food frauds and counterfeiting are hurting the markets of many countries, causing losses for billions of euros per year. The most subjected products to counterfeiting threats are those that earn a great economical value because of their territory of origin or brand, like oil, wine, and cheese. In July 2016, the European Union Intellectual Property Office (EUIPO) has published research about the economical costs of intellectual property infringement in spirits and wine. The results showed enormous damage to the legitimate industries, with estimated losses for approximately 1.3 billion euros of revenue annually [34]. The last but not least component of the concept of food trust is linked to customers' attention, concerning the sustainability of processes and the agri-food chain. Consumers are becoming ever more aware of the role they play in the game for building a sustainable food industry. Most of them, especially the young ones, base their decision-making when buying a product not only on the "classical" factors (price, nutritional values, brand, etc.) but also on the impact that the product they buy has on the environment. The expansion of this new category of customers, known as *Ethical Consumers*, is leading this new market sector to grow very fast, registering in 2015 a growth of 5.3% and 9.7% in 2017 (UK Market) [35,36].

6.3. Combining IoT and Blockchain to Empower the Traceability Process

There is a strong necessity for an enhancement in the monitoring and certification processes to overcome these issues. In the actual situation, most of the compliance data and information are audited by trusted third parties and stored either on paper or in centralized databases. However, these approaches suffer from many informational problems, such as the high cost and inefficiency of paper-based processes, resulting in fraud, corruption, and error happening both on paper and using IT systems.

Thus, there is a need for a new trustworthy approach for registering information about the food chain. The availability of this information to customers is finding a potential powerful partner in the arising Blockchain technology. Indeed, the application of the BC technology in the agri-food sector could lead not only a better and more trustworthy traceability process, but it also could represent a powerful partner in the assessment of the product's value for the buyer. For these reasons, there are actually several experiments and proof-of-concepts aiming to find a way to apply the BC technology into this business field. This potential can be truly unlocked by enabling the blockchain platform to collect, and register on the ledger, data gathered directly from the environment (such as farm fields, food transformation chains, logistic processes, etc.) employing IoT devices [24].

6.4. System Architecture

The scenario in which the use case is applied is about the cold-chain monitoring during logistics operation. In such a context, the IoT device is assumed to be installed into a refrigerated truck that transports seafood from a harbor's warehouse to the final fish-shop. The device can connect to the Internet through a mobile network connection and is in charge of monitoring the fridge's temperature and truck's position employing a temperature sensor and a GPS module, respectively. As soon as the data are collected, the device then sends them to a proper Smart-Contract, in charge of storing them in the BC, using the IoT-blockchain-gateway presented in this paper.

The gateway's hardware configuration used for this specific simulation consists of:

- A PIC32MX 32-bit microcontroller
- A 256-byte EEPROM (used for storing device keys in the developed Proof-of-Concept, keeping in mind that in a real-life application this should become a secure-storage element to avoid the stealing or alteration of the device's key)
- A GSM Module for communicating with the remote RPC server
- A GPS module
- A Temperature sensor

This simple architecture has been chosen to show the limited requirements of this approach. The implemented code is for sure hardware-dependent, but the general architecture is independent from the hardware chosen, and applicable to any kind of similar device.

Moreover, the simulation has been carried out using the Quadrans Testnet blockchain, accessed through its public RPC node, reachable at address `rpc.testnet.quadrans.io`. The communication with the remote RPC is based on HTTPS protocol, which ensures a higher layer of security within the interaction between the IoT device and the remote RPC server.

6.5. Monitoring the Cold-Chain of Fish Products

The boats of Fresh-Fish's fleet are equipped with our device, able to periodically record GPS coordinates. This makes it possible to have the complete history of the route navigated by every boat. When the boats arrive at the harbor, fishermen store the catch into the company's warehouses. Here, each batch of fish (e.g., one kilogram of mussels) is registered into Food-Chain's platform, through its web interface, and is identified with a unique ID that is written into an RFID tag. Then, according to fish shops daily requests, each batch is charged into refrigerator trucks to be delivered to its final destination. When a batch is charged on the truck, the device installed onto

the truck scans the RFID tag on the batch package so that the device knows for which product it is going to start the monitoring process. Once the truck starts its delivery path, the IoT device starts to periodically monitor the temperature and the GPS position of the truck and sends data to the proper smart-contract for being registered, providing also the IDs of the batches under the current monitoring process. In the end, when the truck reaches its finals destination, batches' tags are scanned again for registering that the delivery has been completed and that the recipient retailer is now the owner of such products. At this point, the retailer shares the information with the customers by tagging its product with a specific QR code that represents an entry point to access the information stored in the BC. The customers can verify the complete story of the mussels and check if any violation of the cold-chain parameters has occurred.

A UML diagram of the use case is depicted in Figure 10.

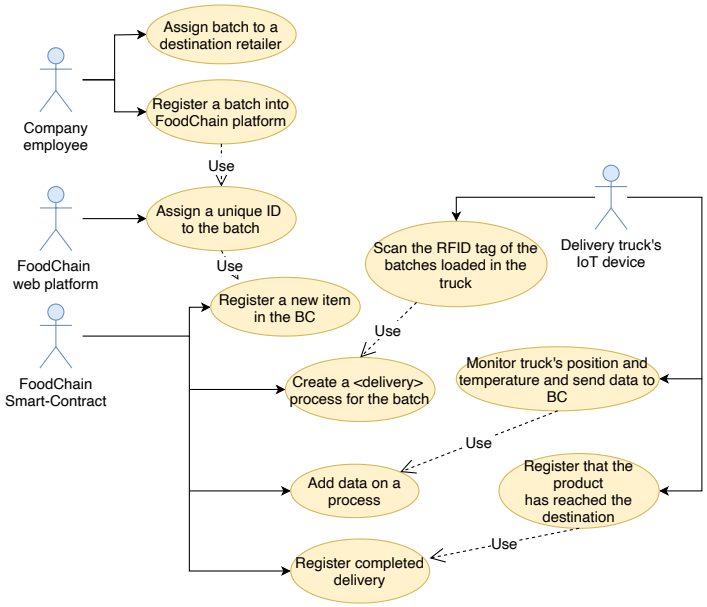

Figure 10. The UML use case diagram of the system (image created using https://draw.io).

A description of a scenario for the use-case involving the IoT device is given in Table 1.

Table 1. Use case scenario description

Step	Description
Pre-Condition	*The item X exists and a <delivery> process p has been activated for it*
1	Read Temperature from temp. sensor
2	Get the position from the GPS module
3	Encode data in the proper format required by the target Smart-Contract
4	Build the transaction object
5	Sign the transaction using the device's private key stored in the EEPROM
6	Send the signed transaction to the BC through an RPC call
8	Wait for the server to reply with the hash of the transaction submitted on the chain
Post-Condition	*Gathered data are now immutably stored in the BC for the item with id X.*

7. Results and Comments

The proposed system tackles the two fundamental issues in the food-chain domain: food integrity and food safety. These are the crucial parameters that efficiently protect customers from frauds or health-related problems, with the positive side effect of helping them to become loyal to a certain vendor. This permits a shift in paradigm, from a trust-based situation to a trust-less scenario. In the former, the buyer is led to put trust in a certain vendor because of an ad campaign, while in the latter the vendor shares with her all the information about the product that cannot be tampered thanks

to the Blockchain technology. The use-case demonstrated how the fish tracking process, carried out combining both Blockchain and IoT devices, can enrich the amount of information that is shared with final consumers guaranteeing their authenticity and integrity without relying on a centralized trust authority.

Moreover, from the logistic company point-of-view, publishing data about their food shippings publicly and in transparency can become a good marketing tool for acquiring new customers interested in both shipping their products and, at the same time, sharing data about shipping-quality with their final consumers.

Last, but not least, the data-authenticity and data-immutability properties implemented by the blockchain can make the information registered by IoT devices valuable in legal trials, in case of debate about food-safety issues or even issues related to the shipment insurance.

Table 2 presents a structured comparison between the proposed solution and the other papers referred in the literature and described before. As the table shows, despite the fact that also other implementations in literature could be theoretically applied to the cold-chain monitoring task, the one presented here is the only one able to fulfill all the requirements together: actually working on resource-constrained IoT devices; the lack of need of a local full node of the BC; the lack of an intermediate and possibly insecure cloud platform; operating with a public domain blockchain, without need for trust-certifying parties; employing devices able to directly sign their own transactions.

Table 2. Comparison between the proposed paper and the cited literature, with a focus on cold-chain monitoring.

Reference	IoT hw Required	Full Node Required	Cloud Required	Kind of BC	On-Device Signing	Directly Applicable to Cold-Chain Monitoring
[21]	n/a	yes	yes	Custom, based on bitcoin	n/a	no
[22]	high-end/SBC	n/a	n/a	Ethereum	n/a	yes, but requires high-end device
[23]	high-end/SBC	yes	no	Ethereum	yes	yes
[24]	n/a	yes	no	Ethereum and Hyperledger	n/a	no
[25]	low-end	n/a	yes	n/a	n/a	no
[26]	high-end/SBC	yes	no	Ethereum	n/a	no
[27]	high-end	no	no	Hyperledger	yes	yes, but on a permissioned BC
This paper	yes	no	no	Quadrans	yes	yes

In order to estimate the performance and the effectiveness of the presented system, the following metrics have been considered:

- Hash Time (T_{hash}): the time needed by the microcontroller to hash the transaction, previously encoded as byte-stream using Recursive-Length-Prefix (RLP) algorithm. This metric is related to the amount of data that is included in the transaction.
- Signature time (T_{sign}): the time needed by the microcontroller for executing the ECDSA algorithm and generating a valid signature for the transaction's hash. Since the signature is always computed over the hash of the transaction (having a fixed length of 32 bytes), this metric is not related with the amount of data making up the transaction, but it gives a good index for evaluating the performance of the hardware on which the system is based.
- Confirmation time ($T_{confirmation}$): the time needed by the BC network to validate the transaction sent by the IoT device. This metric depends only on the blockchain layer and is strictly related to the average block-time of the network.

These three metrics have been chosen since their sum represents the least possible time interval that must elapse between the moment the data are gathered by the device (i.e., an on board sensor) and the moment these data become immutable in the blockchain.

From Table 3 it can be noticed that, as expected, the time required by the microcontroller for running the ECDSA algorithm is constant, and represents the major component of the total time needed by the gateway for preparing and issuing a transaction ($T_{transaction} = T_{sign} + T_{hash}$, in first approximation). At the moment, both hashing and ECDSA operations are implemented in software, but it would be worth exploring possible implementations of the system that can rely on hardware accelerators (maybe implemented on an FPGA connected to the micro-controller) which can speed up the execution of such tasks. Another important aspect is related to the time needed by the BC network for validating the transaction. It can be observed that such time goes in the order of 10–20 s, which reflects the average block-time of Quadrans Testnet that is about 15 seconds. Even if Quadrans Mainnet, the one used in production contexts, has a block-time of 5 seconds that would result in a 3x speed-up, it can still represent a barrier for some time-critical applications in automation and automotive fields, for example the control of a motor/robot or the automatic emergency braking of a car. However, in these particular mission critical applications, the use of a BC does not bring any benefit to the system architecture. This aspect goes beyond the goal of this paper, but trying to improve and reduce this time can become a valuable research case for the Quadrans Foundation because a further improvement in this direction will open the possibility to adapt the presented gateway also in scenarios where time constraints are more strict.

Table 3. Use case performance metrics.

	Min.	***Avg.***	***Max.***
T_{sig} **[ms]**		*15*	
T_{hash} **[ms]**	*2*	*3*	*5*
$T_{confirmed}$ **[s]**	*12*	*16*	*18*

8. Conclusions and Future Works

Summarizing, the integration between two emerging and disruptive technologies such as Internet-of-Things devices and BC-based infrastructure for decentralized applications can bring benefits in a very wide variety of fields, spanning far beyond the simple use case presented in this document. The proposed and developed system gives a proof-of-concept of a possible path to follow for a successful and efficient interaction between Internet-of-Things devices and decentralized applications based on a Blockchain ledger. The *blockchain-enabled gateway* presented in this document satisfies the essential requirements for performing secure and reliable data operation onto the BC infrastructure and, at the same time, provides both hardware and software interfaces for easy and seamless integration even with already developed applications. From the side of the decentralized applications, the possibility to add functionalities that can use data coming directly from the environment, without doubting over their integrity or authenticity, represents a powerful tool that can be exploited to enhance a lot of processes. Finally, for business actors, the services offered by the resulting kind of systems allow them to run a new form of marketing, focused on product and processes transparency. However, the possible application of this promising technologies combination can go even further from traceability and business processes, and be applied for example to healthcare, infrastructures, services, and so on.

Another important feature provided by the designed architecture, and which should be worth exploring better by developing proper use cases and DApps, is the one allowing Smart-Contracts running on the BC to asynchronously trigger some action on specific target IoT devices, identifiable on the network through their public address. Shortly, the idea would be to implement some logic on the IoT devices in order to periodically download the latest N blocks in the chain and search for some emitted event which is targeted to them. However, further studies should evaluate how these concepts can be applied in soft or hard real-time embedded systems, for example starting from the time-critical applications cited above. An exhaustive analysis should take into account not only both the delay of

the network and the architecture itself, but also the delay related to the block-time parameter of the BC. Probably, explorations in this direction may lead to the development of IoT-oriented Blockchain infrastructures that optimize their parameters to be more adaptable to those contexts where timing constraints are critical and sometimes restrictive.

In conclusion, with respect to the currently applied solutions, the one presented in this document allows for developing decentralized applications of any kind that could interact directly with the Cyber-Physical-System, keeping costs low because they do not need to rely on any other service for accessing to the distributed network and, at the same time, satisfying the constraints about data-integrity and data-authenticity.

Author Contributions: J.G. provided the main idea and conducted the research and investigation process. E.G. and J.G. wrote and edited the paper. F.F. provided support during the development of the experimental use case. M.R. critically revised the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lee, J.; Bagheri, B.; Kao, H.A. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. *Manuf. Lett.* **2015**, *3*, 18–23. [CrossRef]
2. Lee, J.; Azamfar, M.; Singh, J. A blockchain enabled Cyber-Physical System architecture for Industry 4.0 manufacturing systems. *Manuf. Lett.* **2019**, *20*, 34–39. [CrossRef]
3. Ferrero, R.; Gandino, F.; Montrucchio, B.; Rebaudengo, M. A cost-effective proposal for an RFID-based system for agri-food traceability. *Int. J. Ad Hoc Ubiquitous Comput.* **2018**. [CrossRef]
4. Ashton, K. That Internet of Things Thing. *RFID J.* **2009**, *22*, 97–114.
5. Atzori, L.; Iera, A.; Morabito, G. The Internet of Things: A survey. *Comput. Netw.* **2010**, *54*, 2787–2805. [CrossRef]
6. Unlocking Opportunities in the Internet of Things-Bain & Company. Available online: https://www.bain.com/insights/unlocking-opportunities-in-the-internet-of-things (accessed on 29 November 2019).
7. Nakamoto, S. Bitcoin: A Peer-To-Peer Electronic Cash System. Technical Report. 2008. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 29 November 2019)
8. Frizzo-barker, J.; Chow-white, P.A.; Adams, P.R.; Mentanko, J.; Ha, D.; Green, S. Blockchain as a disruptive technology for business : A systematic review. *Int. J. Inf. Manag.* **2019**, *51*, 102029. [CrossRef]
9. Tian, F. An agri-food supply chain traceability system for China based on RFID & blockchain technology. In Proceedings of the 2016 13th International Conference on Service Systems and Service Management, ICSSSM 2016, Kunming, China, 24–26 June 2016; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2016. [CrossRef]
10. Salimitari, M.; Chatterjee, M. A Survey on Consensus Protocols in Blockchain for IoT Networks. *arXiv* **2018**, arXiv:1809.05613.
11. Rivest, R.L.; Shamir, A.; Adleman, L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM* **1978**. [CrossRef]
12. Fedorov, A.K.; Kiktenko, E.O.; Lvovsky, A.I. Quantum computers put blockchain security at risk. *Nature* **2018**, *563*, 465–467. [CrossRef] [PubMed]
13. Koblitz, N. Elliptic Curve Cryptosystems. *Math. Comput.* **1987**, *48*, 201–209. [CrossRef]
14. Miller, V.S. Use of Elliptic Curves in Cryptography. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer: Berlin/Heidelberg, Germany, 1986. [CrossRef]
15. Gallagher, P.D.; Romine, C. *Fips Pub 186-4 Digital Signature Standard (DSS) Category: Computer Security Subcategory: Cryptography*; Information Technology Laboratory, National Institute of Standards and Technology: Gaithersburg, MD, USA, 2013. [CrossRef]
16. Wood, G. Ethereum: a secure decentralised generalised transaction ledger. *Ethereum Proj. Yellow Pap.* **2014**. [CrossRef]

17. Buterin, V. Ethereum White Paper. Etherum. 2014 . Available online: https://ethereum.org/en/whitepaper/ (accessed on 29 November 2019).
18. Szabo, N. *Formalizing and Securing Relationships on Public Networks*; First Monday: Canton, TX, USA, 1997. [CrossRef]
19. Costa, D.; Fiori, F.; Milan, P.; Sala, M.; Vitale, A.; Vitale, M. *Quadrans White Paper*; Quadrans Foundation: Mendrisio, Switzerland, 2019.
20. Wu, M.; Wang, K.; Cai, X.; Guo, S.; Guo, M.; Rong, C. A Comprehensive Survey of Blockchain: From Theory to IoT Applications and beyond. *IEEE Internet Things J.* **2019**. [CrossRef]
21. Shafagh, H.; Burkhalter, L.; Hithnawi, A.; Duquennoy, S. Towards Blockchain-based Auditable Storage and Sharing of IoT Data. In Proceedings of the 2017 on Cloud Computing Security Workshop-CCSW '17, Dallas, TX, USA, 30 October–3 November 2017; ACM Press: New York, NY, USA, 2017; pp. 45–50. [CrossRef]
22. Huh, S.; Cho, S.; Kim, S. Managing IoT devices using blockchain platform. In Proceedings of the 2017 19th International Conference on Advanced Communication Technology (ICACT), Bongpyeong, Korea, 19–22 February 2017; pp. 464–467. [CrossRef]
23. Bahga, A.; Madisetti, V.K. Blockchain Platform for Industrial Internet of Things. *J. Softw. Eng. Appl.* **2016**, *9*, 533–546. [CrossRef]
24. Caro, M.P.; Ali, M.S.; Vecchio, M.; Giaffreda, R. Blockchain-based traceability in Agri-Food supply chain management: A practical implementation. In Proceedings of the 2018 IoT Vertical and Topical Summit on Agriculture-Tuscany, IOT Tuscany 2018, Siena, Italy, 8–9 May 2018; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2018; pp. 1–4. [CrossRef]
25. Tsang, Y.P.; Choy, K.L.; Wu, C.H.; Ho, G.T.S.; Lam, H.Y. Blockchain-Driven IoT for Food Traceability with an Integrated Consensus Mechanism. *IEEE Access* **2019**, *7*, 129000–129017. [CrossRef]
26. Xu, Y.; Ren, J.; Wang, G.; Zhang, C.; Yang, J.; Zhang, Y. A blockchain-based nonrepudiation network computing service scheme for industrial iot. *IEEE Trans. Ind. Inform.* **2019**, *15*, 3632–3641. [CrossRef]
27. Dittmann, G.; Jelitto, J. A blockchain proxy for lightweight iot devices. In Proceedings of the 2019 Crypto Valley Conference on Blockchain Technology, CVCBT 2019, Rotkreuz, Switzerland, 24–26 June 2019; Institute of Electrical and Electronics Engineers Inc.: Piscataway, New Jersey, USA, 2019; pp. 82–85. [CrossRef]
28. Birrell, A.D.; Nelson, B.J. I*mplementing Remote Procedure Calls*; Association for Computing Machinery: New York, NY, USA, 1984.
29. Technical Introduction to Events and Logs in Ethereum. Available online: https://media.consensys.net/technical-introduction-to-events-and-logs-in-ethereum-a074d65dd61e (accessed on 29 November 2019).
30. JSON-RPC 2.0 Specification. Available online: https://www.jsonrpc.org/specificationt (accessed on 29 November 2019).
31. Foodintegrity.org. *A Dangerous Food Disconnect When Consumers Hold You Responsible But Don't Trust You*; Technical Report; The Center for Food Integrity: Gladstone, MO, USA, 2018
32. Etienne, J.; Chirico, S.; McEntaggart, K.; Papoutsis, S.; Millstone, E. EU Insights—Consumer perceptions of emerging risks in the food chain. *EFSA Supporting Publ.* **2018**. [CrossRef]
33. Mipaaf-ICQRF-Report Attività 2018. Available online: https://www.politicheagricole.it/flex/cm/pages/ServeBLOB.php/L/IT/IDPagina/13602 (accessed on 29 November 2019).
34. The Economic Costs of IPR Infringement in Spirits and Wine. Available online: https://euipo.europa.eu/ohimportal/en/web/observatory/ipr_infringement_wines_and_spirits (accessed on 29 November 2019).
35. Hancoak, A. Younger Consumers Drive Shift to Ethical Products. *The Financial Times*, 22 December 2017.
36. Venkatesh, V.G.; Kang, K.; Wang, B.; Zhong, R.Y.; Zhang, A. System architecture for blockchain based transparency of supply chain social sustainability. *Robot. Comput. Integr. Manuf.* **2020**, *63*, 101896. [CrossRef]