Contents lists available at ScienceDirect

Blockchain: Research and Applications

journal homepage: www.journals.elsevier.com/blockchain-research-and-applications

A blockchain architecture for industrial applications

Lodovica Marchesi [*], Michele Marchesi, Roberto Tonelli, Maria Ilaria Lunesu

Department of Mathematics and Computer Science, University of Cagliari, Cagliari, 09124, Italy

A R T I C L E I N F O

Keywords:
Permissioned blockchain
Industrial application
Information system
Software application
Ethereum

A B S T R A C T

Blockchain and the programs running on it, called smart contracts, are increasingly applied in all fields where trust and strong certifications are required. Our work focuses on industrial applications of blockchains and not on cryptocurrencies or tokens. We use frameworks to compare public and permissioned blockchains specifically suited for industrial applications. We also propose a complete solution based on Ethereum to implement a decentralized application, putting together in an original way, components and patterns already used and proven. This solution is characterized by a set of validator nodes running the blockchain using Proof-of-Authority or similar efficient consensus algorithms, by the use of an explorer enabling users to check the blockchain state, and the source code of the smart contracts running on it. From time to time, the hash digest of the last mined block is written into a public blockchain to guarantee immutability. The right to send transactions is granted by validator nodes to users by endowing them with the Ethers mined locally. Overall, the proposed approach has the same transparency and immutability as a public blockchain, largely reducing its drawbacks.

1. Introduction

Blockchain technology is a smart mix of known technologies, first introduced in 2008 by Satoshi Nakamoto, as the enabler of the digital currency known as Bitcoin [1].

The main requirement of Bitcoin, as with every currency, is trust. In fact, to put your savings into a currency, you need to trust that the money will still be expendable for a reasonable number of years, secure against counterfeiting, theft, confiscation, double-spending and inflation, and easily transferable. Another requirement that diversifies Bitcoin from any other currency is the absence of a central authority guaranteeing and managing it. In Bitcoin, these features are obtained by using a distributed peer-to-peer network based on open-source software running in every node, each holding a copy of the transaction database, known as the blockchain.

The high number of nodes and the economic incentives given to miners (who validate transactions and pack them into blocks in return for a reward) guarantee the survival of the network and its robustness; the equality and openness of all nodes make the network decentralized; the transparency and immutability of the blockchain enable trust; the consensus mechanism used to add new transactions to the blockchain (Proof-of-Work) guarantees against massive Sybil attacks; the limited number of Bitcoins mined guarantees against inflation.

The success of Bitcoin, whose market value changed from zero to several hundred billion USD in less than a dozen years, is testimony to the success of the Bitcoin vision and of its underlying technology, the blockchain.

The main focus of this paper, however, is not on cryptocurrencies but on other applications of blockchain. A few years after the introduction of Bitcoin in 2009, developers and businessmen realized that a blockchain can also be used to run a decentralized computer. The first successful blockchain able to run Turing-complete programs, called “smart contracts” (SCs) following Nick Szabo's idea [2], was the Ethereum blockchain [3], started in 2015 and whose cryptocurrency, the Ether (ETH), soon became the second largest for market capitalization after Bitcoin.

The main objective of this work is to clarify, discuss, and add new ideas and tools to the structure and management of permissioned or consortium blockchains, that is, blockchains whose nodes are run by selected organizations. These systems are also called “distributed ledgers” (DL or DLT, where ‘T’ is for “technology”). In principle, a blockchain is a DLT, but a DLT does not necessarily use a chain of blocks to store its information, guaranteeing its immutability. In this paper, we mainly use the term “blockchain”, but most of the concepts can also be applied to DLT.

In particular, we address those based on blockchains similar to Ethereum, where you need to consume their cryptocurrency to send

[*] Corresponding author.
E-mail address: lodovica.marchesi@unica.it (L. Marchesi).

https://doi.org/10.1016/j.bcra.2022.100088
Received 21 September 2021; Received in revised form 15 March 2022; Accepted 30 April 2022

transactions able to modify the blockchain's state. The amount to consume is called "gas".

Contributions. The main contributions of our work are the following:

(i) We recap the features and qualities of blockchain-based systems, based on both public and so-called "permissioned" or "consortium" blockchains. We select criteria from existing frameworks for choosing the blockchain architecture most suited to a specific application.

(ii) We use this framework to justify and propose an architecture for managing consortium blockchains, which retains all the positive characteristics of public blockchains but largely reduces their drawbacks regarding scalability, privacy, cost, and efficiency.

(iii) We collect together, for the first time in a structured way, many ideas already present in the blockchain realm. The result is an architecture that is easily applicable to most consortium blockchains, being efficient, highly configurable and scalable. This architecture makes use of the Ethereum technology but can be easily changed to support other blockchains, such as Hyperledger.

(iv) We better formalize the typical permissioned architecture, explaining the characteristics it must have to have transparency and strength almost equal to that of a public blockchain. Precisely, the permissioned blockchain must be periodically anchored to a public blockchain (a tool which is already used, especially in distributed data storage solutions), and at the same time, an explorer able to explore the blockchain independently from the provided apps must be provided.

(v) For blockchains based on the gas mechanism, a further contribution is to use gas (Ether or other cryptocurrencies of that specific blockchain) to enable writing only for authorized actors. The idea is that instead of giving permissions depending on your login authorization, you are enabled because you have available gas. This is useful because since gas is limited, it also allows for dynamically managing the write permissions.

(vi) We present a real case study to show how the evaluation framework was used and how the system was implemented.

Structure of the paper. The remainder of this paper is organized as follows. Section 2 presents the related work; Section 3 discusses the requirements of permissioned blockchains for industrial applications and selects the evaluation criteria; in Section 4, we apply the framework to choose the blockchain platform best suited to our purposes; Section 5 describes the proposed dApp architecture; Section 6 presents a real case study featuring a dApp implementing agri-food provenance and quality certification; Section 7 draws the paper's conclusions.

2. Related work

The aim of this paper is to discuss the differences between public and permissioned blockchains, to select criteria belonging to existing frameworks to choose among different kinds of blockchains and DLT, and to propose a specific solution to implement publicly accessible permissioned blockchains. Consequently, we will consider only works on these specific subjects and not generic works on blockchain and SC technologies and applications.

Regarding evaluation frameworks, some papers addressed whether to use a blockchain or not when implementing a specific information system. Among them, we may quote the seminal work by Peck [4], and the more recent works by Wüst and Gervais [5] and Hassija et al. [6], although in our paper we already assume that the choice to use a blockchain has already been made.

Once the choice to use a blockchain is made, there are many papers in the literature that provide guidelines on how to choose the best-suited blockchain technology for a specific application. In 2017, Koteska et al. [7] investigated the quality requirements and solutions for blockchain implementations, starting from a literature review. They analyzed the various quality issues of blockchain systems, focusing in particular on public blockchains. They gave a catalog of blockchain-specific quality criteria to provide high data integrity, security, reliability, and node privacy.

In 2018, Scriber [8] proposed a framework for determining blockchain applicability, which includes a list of 10 blockchain characteristics whose presence makes it desirable for a dApp system. Though Scriber's work is mainly oriented to deciding whether a blockchain is suitable for a given application or not, we used many concepts and ideas of his framework to build ours, which is instead oriented to comparatively evaluating different blockchain solutions.

Maranhão et al., working in a focus group promoted by the U.N. agency International Telecommunication Union (ITU) to assess criteria for distributed ledger technology platforms, proposed a DLT Assessment Framework [9]. They defined three layers: (i) Core Technology Layer; (ii) Application Layer; and (iii) Operation Layer; and assigned specific criteria to each layer. They proposed one of the first DLT assessment frameworks to be standardized by an international standardization body. They then showed how the framework can be applied, evaluating the public Ethereum blockchain.

In their very detailed work, Gourisetti et al. proposed the blockchain applicability framework (BAF), specifically designed with the purpose of helping to decide not only whether a blockchain is suitable for a specific application but also what kinds of blockchain, consensus model, and features are most appropriate [10]. The BAF is divided into five domains, 18 subdomains, and about 100 controls, making it comprehensive but conversely not easy to master and apply.

Colomo-Palacios et al. [11] discussed blockchain assessment initiatives from a technology evolution viewpoint, from Blockchain 1.0 (Bitcoin and the like) to Blockchain 2.0 (Ethereum and SCs), Blockchain 3.0 (IOTA, Cardano, Tezos, etc.), and Blockchain 4.0 (use of A.I., Blockchain as a Service, etc., still ongoing). They examined nine papers on blockchain assessment models (including [8,9]), extracting technical and business-oriented aspects. A total of 19 factors were found, 14 technical and five business-oriented.

Garriga et al. [12] proposed Chainmaster, a conceptual framework to aid software architects, developers, and decision makers in adopting the right blockchain technology. They identified seven key architecture features of blockchain systems: (1) cost, (2) consistency, (3) functionality and functional extensibility, (4) performance and scalability, (5) security, (6) decentralization, and (7) privacy. They then analyzed the technological decisions in the most popular blockchains and DLT and mapped them against the key features. The Chainmaster framework was then evaluated on four real blockchain projects.

Regarding work supporting architectural design decisions on the blockchain most suited to an application, the literature is still very limited. Wessling et al. [13] supported the process of integrating decentralized elements, but they focused on lower-level design patterns and did not provide true architectural guidance. A very recent work by Wöehrer and Zdun about architectural design decisions covers the implementation and integration of blockchain-based solutions [14]. They described architectural design decisions and related options in terms of patterns and practices. Since most design decisions are driven by the need to offset current blockchain drawbacks—typically scalability, privacy, and usability—by using centralized elements, the authors conclude that a hybrid architecture is beneficial in many design situations.

Regarding the consensus in permissioned blockchains, their controlled environment, the need to obtain high performance, and the absence of the necessity to directly compensate for the validators, rules out the Proof-of-Work and the Proof-of-Stake approaches. In these blockchains, nodes are divided between validators and simple nodes. Simple nodes can send transactions and query the blockchain, whereas only validators can create new blocks and add them to the blockchain.

Most algorithms used in permissioned blockchains belong to the

Byzantine fault-tolerant (BFT) consensus family [15]. In BFT, the consensus can tolerate a ratio of malicious validators below 1/3 of the total number of validators. In an industrial permissioned blockchain, validators belong to trusted organizations. Therefore, the probability that one validator might become malicious and cheat is low and that malicious ones become one-third of all validators, or more, is negligible. Of course, the total number of independent validators should be at least seven, and preferably 10 or more.

Depending on the number of validators and simple nodes, the actual consensus algorithm used may vary. In the case of permissioned blockchains for industrial applications, it is difficult to forecast networks with more than a few tens of validators. In this case, the preferred consensus mechanisms are variations of BFT, namely “Practical BFT” (PBFT) [16], “Istanbul BFT” (IBFT) [17], QBFT [17], “Delegated BFT” (DBFT) [18], or other specific algorithms such as “Clique” [17], “Proof of Elapsed Time” (PoET) [19], “Authority Round” (AuRa) [20], and Tendermint protocol [18].

Comparative evaluations of these consensus mechanisms in practical blockchain testbeds were presented by Shapiro et al. [18], Ahmad et al. [19], and Gerrits et al. [17]. All these studies evaluate the throughput of the system, measured in the maximum number of transactions per second (Tx/s), as a function of the number of nodes. Note that, due to communication overload, performance tends to decrease with the number of validators.

Shapiro et al. evaluated IBFT, DBFT, and Tendermint, showing that DBFT outperforms the other two by about one order of magnitude.

Gerrits et al. compared PBFT, IBFT, QBFT, and Clique, in the context of a use case taken from the automotive industry. This study shows that the performance of the original PBFT is not adequate; IBFT and QBFT can handle up to about 450 Tx/s and Clique three times more.

Ahmad et al. compared PBFT, PoET, and Clique, as well as Proof-of-Work and Proof-of-Stake. In their test, Clique outperformed PoET in terms of Tx/s, up to 50 validators. For more validators, Clique and PoET had approximately the same throughput. PoS was substantially slower, except for blockchains of more than 150 nodes. PoW and PBFT were always heavily outperformed.

The cited works cannot be directly compared because they used very different benchmarks and contexts. However, DBFT and Clique look like the best choices for permissioned blockchain consensus. In particular, according to Gerrits et al., Clique can handle a maximum of 1500 Tx/s, decreasing to 1100 Tx/s with 25 validator nodes. Ahmad et al. claimed 8000 Tx/s with 5 nodes and 10 nodes, and almost 5000 Tx/s up to 50 nodes. Afterwards, the throughput decreases to around 1000 Tx/s for 200 nodes and 250 nodes.

All things considered, Clique emerged as the best consensus mechanism for permissioned blockchains up to about 50 validator nodes, also due to its popularity and availability. In our experience, the number of independent organizations participating in the blockchain and willing to run a validator node seldom exceeds 20–30 units. This further confirms Clique as the consensus protocol of choice.

3. Uses of dApps and kinds of blockchains

The software programs using a blockchain are called “decentralized applications” or “dApps”, and are one of the main new trends in software development. A search of scientific and technical documents made with Google Scholar in July 2021 found 36,700 results for “smart contracts” development, a number higher or much higher than the results for microservices development (20,500), global software engineering (7670), devops development (23,500), and even IoT “software development” (30,400). dApps include not only the SCs actually running on a blockchain but also the software managing data outside the blockchain and the user interface to interact with it.

Initially, the primary use of SCs was to manage second-level digital currencies, called “tokens”, mainly used to finance the Initial Coin Offers—crowdfunding operations gathering cryptocurrencies to finance startups [21]. Besides tokens, dApps are now being used for many applications in the fields of data notarization, finance and insurance contracts, supply chain management [22–24], smart and microgrid management [25], health sector (personal records, pharmaceutical product delivery, clinical trials, etc.) [26], identity management and access control systems [24], decentralized notary [24], gambling, gaming, voting [24], and many others [25,27].

dApps and SCs can be used for automated enforcement of contractual obligations, without having to trust a central authority, and without space and time constraints.

In short, the properties that make a blockchain-based system unique are:

- Distribution: the information is stored on multiple computers, giving resilience and security to the system;
- Traceability: all transactions are traceable in every part, and their original address can be known with certainty;
- Trust: the ownership of a given address, where assets can be stored, is guaranteed by asymmetric cryptography;
- Anonymity: the addresses' owners do not need to publish their names to send a transaction, but only to prove the ownership of the related private key;
- Decentralization: transactions are managed without a central authority;
- Transparency: the contents of the blockchain are easily accessible and verifiable;
- Immutability: the accepted data can no longer be modified in any way;
- Programmability: complex actions (smart contracts) can be programmed whose code and execution are also fully verifiable.
- Low-cost: the system is managed by open-source software, with low maintenance costs and possibly low running costs (depending, however, on transaction fees).

3.1. *Kinds of blockchains*

The first blockchains were public, that is, truly decentralized, censorship-resistant, anonymous, and without participation and access limits. However, public blockchains are not without problems and limitations. They are typically unique systems, so they have performance and scalability issues because the maximum number of transactions per second is low, and the size of the blockchain is ever growing. Moreover, their energy consumption is high due to the “Proof-of-Work” consensus still used by the most popular ones, and there is no privacy or control of the information to be written and read.

To address these issues and still be able to use the technology in real-world applications, permissioned blockchains were introduced. These are closed networks in which previously designated parties interact and participate in data validation and management. For this reason, they are also called “consortium blockchains”.

The main feature a blockchain must exhibit is that it is an intrinsically distributed system without a central authority. In permissioned blockchains, the number of nodes is much lower than those in public ones, but they must still be decentralized across known participants. The reasons for decentralization may be various. For instance, no single organization might be willing to run the system for reasons related to cost or legal liability; or the organizations involved might not wish to let just one of them run the system; or having many independent nodes could be a guarantee of persistence and immutability of the system.

It is also possible that a single organization runs individually a blockchain or DLT to take advantage of its features for internal applications. Then, however, most of the reasons to use a blockchain do not hold, so we will not consider this case in the rest of this work.

Nodes in a permissioned blockchain can be validators, which are able to participate in the consensus mechanism to validate and add new blocks to the blockchain, or simple nodes hold a copy of the blockchain.

To run a consortium blockchain, there are many software systems available. Most public blockchain software is open source and suitable for use in managing a consortium blockchain. A prominent consortium blockchain, aimed mainly at banking applications, is Ripple, developed by a private company and presently run by about 150 invited validators. There are also projects aimed at building consortium blockchain software. The most popular among these is Hyperledger, an open-source collaborative effort hosted by the Linux Foundation, aiming to build cross-industry blockchain technologies.

In general, blockchains can be classified according to how they grant the right to add new validators or simple nodes, and the right to read/write information. Regarding permissioned blockchains, we made a further distinction between systems intended only to be used by specific, authorized partners (closed systems) and systems intended to be accessed also by the general public (open systems). An example of the former might be a system to perform and clear money transfers among banks, which should be accessed only by the banks and possibly by a control authority. An example of the latter might be a system to guarantee the provenance and quality of foods, which should be accessible by whoever buys the certified food. Table 1 shows the proposed classification.

In public blockchains, everyone can add a node and be able to validate and add new blocks through mining, Proof-of-Stake or other consensus algorithms compatible with uncensored participation. Access to the functions of specific SCs, however, might be granted only to authorized addresses, implementing control at the SC level. For instance, you can change the state of an SC holding ERC20 tokens on the Ethereum blockchain only if your address already owns some tokens—the change can only consist of the transfer of one's own tokens to another address.

In permissioned blockchains, the right to add a node is managed by the consortium of organizations running the blockchain, according to the original legal contract among them. Both open and closed permissioned blockchains manage validators and simple nodes by granting them specific permissions at the Internet connection level. This requires the intervention of systems engineers to administrate the network.

In open permissioned blockchains, the addition of a node and the downloading of the blockchain are granted to everyone. Validator nodes, however, must be approved, either automatically through a poll among existing validators, using a suitable SC, or by the consortium members. Deploying a new SC on the blockchain can be made only by participants with the permission to do so. Everyone can send transactions to the blockchain, but, as in public ones, it is the task of the SC receiving the transaction to decide whether to accept or not the request.

Closed permissioned blockchains typically grant access permission only to clients who access the system using appropriate credentials. The specific authorizations granted also depend on these credentials, as in classic information systems.

Of course, it is also possible to have a permissioned blockchain granting open access only to specific SCs but not to the whole blockchain. In this case, the system is classified as closed but holding services that are in fact open.

3.2. *An evaluation framework*

One of the main goals of our work is to facilitate the decision of which specific blockchain architecture to choose once the choice is made to use a dApp for implementing a given application. For this purpose, there are already various frameworks, among which the most relevant were reported in Section 2. We decided to use the work of Scriber as a starting point, noting, however, that this framework is focused on evaluating the suitability of blockchain technology for a given application, more than choosing among different blockchain architectures.

We kept seven features of Scriber, removing “Workflow”, “Transactions” and “Inefficiency”, which are more focused on the decision of whether to use a dApp to implement a specific system, or not. We added three blockchain-specific quality criteria taken from three other evaluation frameworks (described below), which complement Scriber's ones: “Privacy”, “Cost”, and “Scalability”. We added two more specifications of cost—the cost of adding a new node (Deployment cost) and of dApp development (Development cost), which we deemed very important in the choice of the architecture and technology to adopt.

The features a dApp system must exhibit according to our evaluation criteria are reported in the first three columns of Table 2. The remaining two columns show an evaluation of generic public and permissioned blockchains, as described in Section 3.1.

Features 1–6, taken from Scriber, basically aim to gain user trust without having to trust all blockchain nodes. Here, a node might cease to be trustworthy also because it is withdrawn from the network, and not necessarily because it tries to attack the system.

Features 7–12 are desirable for all software systems but are especially difficult to obtain in public blockchains.

When developing a dApp system, the first issue to address is whether to use a public or a consortium blockchain. Public blockchains are open to everyone; the most used for implementing dApps is Ethereum, but others are available, such as EOS, Binance Smart Chain, Steem, TRON, and many others. As of November 2021, 2886 dApps were running on the Ethereum public blockchain, out of a total of 3799 surveyed dApps [28].

Table 2 also shows a comparison of public and permissioned blockchains with regard to the proposed framework, with qualitative scores. Note that in the table, we consider the features of the most used and

Table 1
Classification of blockchain types in relation to validation and access.

Action	Blockchain type		
	Public	Permissioned, open	Permissioned, closed
Managing the right to add a node	Not contemplated	Depending on the original legal contract	Depending on the original legal contract
Adding a node able to mine/validate	Everyone; the cost of mining might be high	Only if granted permission, possibly through voting using a smart contract	Only if granted permission
Adding a node holding the blockchain	Everyone	Everyone	Only if granted permission
Deploying a smart contract	Everyone, paying a fee or “gas”	Only if granted permission	Only if granted permission
Sending trans-actions able to change the state (writing rights)	Everyone, paying a fee or “gas”. Most smart contracts will change their state only if transactions come from authorized addresses	Everyone. Smart contracts will change their states only if transactions come from authorized addresses	Only if correctly logged in, with proper authorizations. A check on the address might also be performed
Sending read only transactions to one or more smart contracts	Everyone	Everyone, but the request may be accepted only if transactions come from authorized addresses	Only if correctly logged in, with proper authorizations. A check on the address might also be performed
Reading the content of the blockchain	Everyone	Everyone	Only if correctly logged in, with proper authorizations

Table 2
The features needed by a dApp system and how public and permissioned blockchains support them.

#	Feature	Description	Public blockchain	Permissioned blockchain
1	Immutability	A blockchain is an append-only system—once written, the information cannot be changed or deleted. The data and programs running on the blockchain must be verifiable, immutable and counterfeit-proof	Very high	High, can be very high if periodically "anchored" to a public blockchain
2	Transparency	The data and the activities performed on the blockchain must be entirely traceable. Anyone, possibly with suitable access rights, should be able to explore the blockchain, to verify this	Very high	Depending on the system; can be very high
3	Trust	What is the level of trust among participants? A blockchain can ensure trust even with no trustworthy participants.	It works well even if participants do not trust each other	Trust is needed to field the initiative. The blockchain can withstand attacks from a few participants
4	Identity	All writing activities performed on the blockchain must come from certain origins	Very strong, based on private key ownership; the owners can publicly associate their identities to their addresses	Strong if based on username and password, very strong if based on private key ownership
5	Historical records	The system repositories and apps, blockchain included, must be kept running for a suitable amount of time—typically in the range of years or decades—with negligible risk of being interrupted or terminated before the time	Very high, based on miners' reward	High, depending on the willingness and convenience of the validators
6	Ecosystem	Does the architecture support interoperability among partners, as opposed to a single company system?	Totally achieved	Easily achieved
7	Efficiency	The system should be able to provide the required throughput, with proper response times, even in the case of many users and many transactions per unit of time	The number of transactions per second is quite low	The number of transactions per second can be high
8	Privacy	The permission to access the blockchain, in particular to change its state, must be granted only to known users, possibly at various access levels	Very low; smart contracts can allow actions depending on the specific address sending the transaction	High; can be enforced at various levels
9	Scalability	The system should be able to scale, if needed	Poor scalability if the number of dApps and users increases	High, by deploying further blockchains on the same node, and/or by splitting the nodes
10	Cost	The blockchain system should be open source, easily deployed, and requiring limited hardware and network bandwidth resources, compatibly with the size of the dApp, and the number of transactions per second. The cost should not be volatile	There are only software development and execution costs; the latter costs can be very volatile and unpredictable	Infrastructural costs are typically low; execution costs are low and predictable
11	Deployment costs	System deployment costs are low	Costs to add a node are usually very low	Costs to add a node can be quite low
12	Development costs	System development costs are low	Costs depend on the availability of developers and maturity of development tools	Costs depend on the availability of developers and maturity of development tools

proven blockchains, such as those cited above. We are aware that there are new projects aiming to overcome the limitations of public blockchains in terms of throughput, cost, and scalability. However, these projects are still in progress. From an industrial applicability perspective, the cited technologies are by far still the best in terms of reliability and ease of finding development resources—both tools and skilled people.

Public blockchains look the most stable and easiest to start with but lack performance and scalability. Their cost is unpredictable due to the high volatility in cryptocurrency values and transaction validation fees. Moreover, they do not support data privacy, and thus can be non-compliant with respect to the strict guidelines of modern privacy laws, such as the European GDPR.

For these reasons, public blockchains are mainly used for applications managing digital money, such as the above-cited tokens, and for the notarization of information. In our proposal, we focus on non-monetary, industrial applications, and thus on consortium blockchains. The proposed solution also includes the use of a public blockchain to make the permissioned blockchain immutable.

In Table 3, we show our criteria against those of four other evaluation frameworks: (i) the 10 criteria of Maranhão et al. [9], which will be part of the forthcoming ITU standard; (ii) those of Chainmaster by Garriga et al. [12]; (iii) the assessment factors of Colomo-Palacios et al. [11]; and (iv) the characteristics of Scriber [8].

Note that we considered some criteria—those tagged with "This feature is assumed"—to be fulfilled by default by blockchain technology or to be not relevant to permissioned blockchains, so they are not considered by our framework. The last criterion shown, "Consistency" of the Chainmaster framework, was deemed not relevant because it is defined as the time to confirm that a transaction is securely appended to the blockchain. This criterion is important in public PoW blockchains, but it is not relevant in permissioned blockchains, whose consensus algorithms are not subjected to forks that can cancel valid transactions.

4. Choosing the blockchain platform

We define as dApp a software system that uses DLT, typically a blockchain, as a central hub to store and exchange information through SCs. A dApp is composed of SCs running on a blockchain and of applications able to create and send transactions to them. These applications typically provide a human interaction interface, running on a PC or on a mobile device. Additional information could be stored on one or more servers, and business logic could also be executed on these.

Our primary goal was to design a suitable blockchain architecture for "industrial" applications, that is, information systems whose goal is to manage contractual relationships between industrial customers and suppliers, including supply chain management. Clearly, the first step is to choose the underlying blockchain "engine". We evaluated the public Ethereum blockchain against what we believe are the most mature and widely used technologies to implement permissioned blockchains. They are: (i) Ethereum using Clique, a Proof-of-Authority (PoA) consensus mechanism discussed in Section 2—we call this platform "Ethereum PoA"; and (ii) Hyperledger Fabric.

Table 3
A comparison of the features needed by a dApp system according to different evaluation frameworks.

#	This paper	Maranhao ITU	CHAINMASTER	Colomo Palacios	Scriber
1	Immutability	–	–	Immutability	Immutability
2	Transparency	3.3.3 Auditability	–	Transparency	Transparency
3	Trust	–	–	Trust	Trust
4	Identity	–	–	Identity	Identity
5	Historical records	3.1.3 Sustainability	–	Historical record	Historical record
6	Ecosystem	3.1.5 Interoperability	Extensibility	Ecosystem	Ecosystem
7	Efficiency	3.1.2 Performance	Performance	Efficiency	–
8	Privacy	–	Privacy	–	–
9	Scalability	3.3.1 Scalability	Scalability	Scalability	–
10	Cost	–	Costs	Costs	–
11	Deployment costs	–	Costs	Costs	–
12	Development costs	–	Costs	Costs	–
	This feature is assumed	3.1.1 Security	Security	–	–
	This feature is assumed	3.1.4 Governance	–	Governance	–
	This feature is assumed	3.2.1 Smart Contract Programmability	Functionality	–	–
	This feature is assumed	3.2.2 Smart Contract Data Access Control	–	Smart contracts and data access control	–
	This feature is assumed	3.3.2 Stability	–	–	–
	This feature is assumed	–	Decentralization	Distribution	Distribution
	See costs	–	–	Maintainability	–
	Non-relevant to permissioned blockchain	–	Consistency (time to confirmation)	–	–

This choice is confirmed by the recent work of Polge et al. [29], who listed and compared five major private blockchain frameworks. Besides Ethereum and Hyperledger Fabric, they also considered Quorum, which is a fork of Ethereum; MultiChain, which is a fork of the Bitcoin blockchain, but in its stable version 1.0 does not allow SCs; and R3 Corda, which is especially devoted to financial applications. Another platform whose popularity is increasing is Hyperledger Besu, which is compatible with Ethereum. Both Quorum and Besu can easily be used in place of Ethereum PoA, so we just evaluated the latter.

To justify the choice of the blockchain platform, it is possible to use the features of Table 2 to define a framework to determine the best architecture with respect to a specific application. Each feature is evaluated using an integer scale from 1 (least suited) to 5 (most suited).

The criteria are subjectively weighted by importance, with the weight values related to the specific system to implement. We chose the weights targeting a system whose goal is to manage contractual relationships between industrial customers and suppliers, including supply chains. Such a system would certify orders, provisions and shipments of raw materials, semi-finished and final products, and their processing steps. The system actors are the various supplier and customer firms, the wholesalers, and the certification authorities.

Table 4 shows the 12 criteria, the weight given to each of them, and the evaluation scores of the three platforms. The total scores are the weighted sums of all 12 criteria.

To obtain these scores, we interviewed seven blockchain experts, five from academia and two from a private company that produces dApps. The experts agreed on the weights to assign to each feature and voted independently. The median of the seven votes was adopted to rate every criterion for the three platforms, and the total score was computed for comparison.

The rationale behind these weights is the following:

- The most important criteria for a permissioned blockchain were deemed to be:
  - Immutability, because a data structure managed by a limited set of organizations might be the target of a successful attack, or even some of the participants might collude to alter the data.
  - Identity, because being certain about the identity of participants who send transactions is a key requirement in contractual relationships.
  - Efficiency, in both throughput and data storage is an obviously important requirement.
  - Cost, again for obvious reasons; note that various types of cost are taken into account in three criteria, so overall, it is the most important requirement.
- The second most important criteria are transparency (this could have been even higher), stability over time, ease of management of access permissions, compliance with privacy laws, and scalability. All these

Table 4
The features needed by a dApp system and how public and permissioned blockchains support them.

#	Feature	Description	Weight	Ethereum main network	Ethereum Proof-of-Authority	Hyperledger Fabric
1	Immutability	Risk of forgery of data and/or smart contracts. 1: high risk – 5: low risk	5	5	4	4
2	Transparency	Ease to inspect the blockchain, having suitable access rights	4	5	5	4
3	Trust	The system can increase trust between participants	3	5	4	4
4	Identity	All writing activities must come from certain origins	5	3	4	4
5	Historical records	Risk that the system will not run for a suitable amount of years. 1: high risk – 5: low risk	4	4	3	3
6	Ecosystem	The architecture facilitates integration of various companies	3	4	5	5
7	Efficiency	Ability to provide high throughput and low response time	5	1	5	5
8	Privacy	Ease to manage access permissions to the blockchain	4	3	4	5
9	Scalability	The system should be able to scale, if needed	4	2	4	4
10	Cost	Writing costs are reasonable and stable	5	1	5	5
11	Deployment costs	System deployment costs are low. 1: high cost – 5: low cost	3	5	3	3
12	Development costs	System development costs are low. 1: high cost – 5: low cost	4	4	4	2
	TOTAL SCORE		–	164	206	198

criteria are very important when managing contractual obligations or guarantees of quality and provenance.
- Slightly less important, but still important, are trust, which in a permissioned blockchain is often taken for granted, and ease of integration among parties, which is partially already included in the cost criteria.

Note that there is no criterion whose weight is below 3. In fact, all the framework criteria are important for judging the suitability of a blockchain architecture for industrial applications.

The scores regarding immutability, transparency, and trust of the permissioned blockchains are very close to those of the public ones. This is due to the fact that the permissioned blockchains considered are periodically “anchored” to a public blockchain, and that they are provided with an explorer enabling independent browsing of their state. These features will be described in detail in the next section.

The “winner” is Ethereum PoA, with 206 points, whereas Hyperledger largely prevails over the Ethereum main network.

The Ethereum main network was penalized mainly by the unpredictability and amount of its transaction costs, as well as by its low efficiency and scalability, which were quite highly weighted criteria. Note that the new Ethereum 2.0, or Eth2, which was recently released in December 2020 with the shipping of the Beacon Chain, will provide much higher efficiency and scalability [30]. However, the new version is still in its early experimental phases.

Regarding Hyperledger Fabric, we know that it is one of the most used DLT systems for industrial applications [31,32]. In our comparison, Fabric got a score almost equal to that of Ethereum PoA. It was considered slightly better for privacy but slightly less transparent, and with higher development costs due to the higher complexity of the Hyperledger Fabric platform with respect to the Ethereum one, and to the smaller number of skilled developers available.

5. The proposed dApp architecture

Once we chose the platform, we built the overall dApp architecture, starting from general considerations but also considering the specificities of Ethereum, among which the need to consume “gas” for sending transactions is perhaps the most relevant. The proposed dApp architecture is shown in Fig.1. This is a general-purpose architecture, showing all the possible components. In specific applications, some components might not be needed and should be removed.

The proposed architecture has four kinds of actors:

- **Validators** (shown with a bold “V”), the nodes running the system are managed by the key consortium participants. These nodes hold a copy

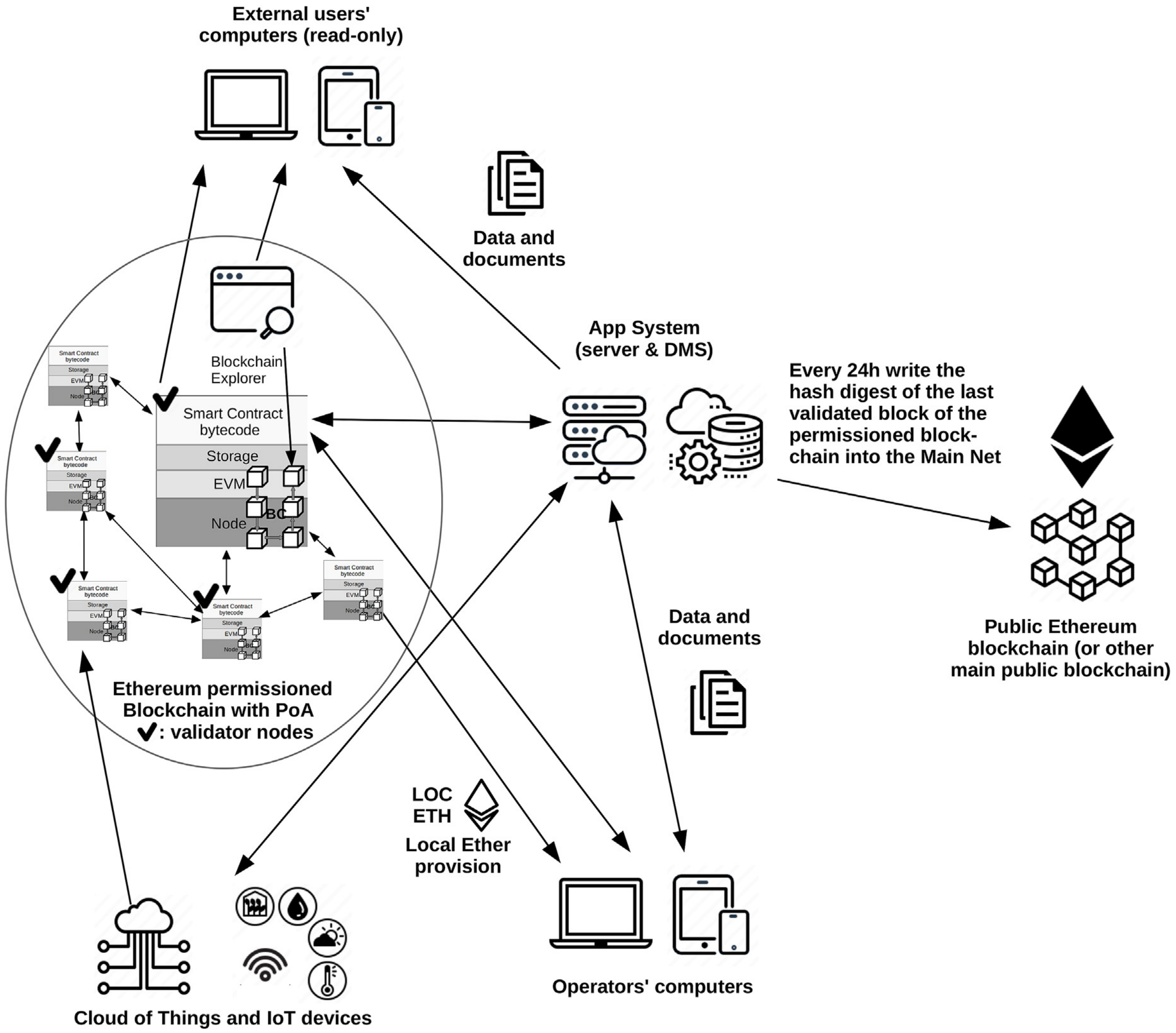

External users' computers (read-only)
Data and documents
Blockchain Explorer
Smart Contract bytecode
Storage
EVM
Node
BC
App System (server & DMS)
Every 24h write the hash digest of the last validated block of the permissioned blockchain into the Main Net
Public Ethereum blockchain (or other main public blockchain)
Data and documents
Ethereum permissioned Blockchain with PoA
✓: validator nodes
LOC ETH
Local Ether provision
Cloud of Things and IoT devices
Operators' computers

Fig. 1. The proposed architecture of a dApp application. PoA: Proof-of-Authority.

of the blockchain, validate transactions, group them into blocks, and decide to add blocks to the blockchain using the Clique consensus mechanism, or a similar one.
- **Participants**, the nodes holding a copy of the blockchain are able to receive, validate, and broadcast transactions but not able to participate in the consensus mechanism. These nodes are managed by organizations that have obtained the permission to do so but are not (yet) full members of the consortium.
- **Operators**, who are enabled to send transactions changing the blockchain state. Operators use terminals and GUI software, which are part of the overall system and belong to organizations participating in the system.
- **External users**, who can access the system nodes in read-only mode, using standard terminals and software provided by the system.

The validators are run by the organizations of the consortium, which should be independent of each other, to avoid a single organization trying to falsify the blockchain data, or simply deciding to stop supporting the system.

It is important to assess the reasons why validators take the burden of managing the blockchain. The main reasons are either proposing the blockchain as a service to customers for a profit or being involved in the management of the dApp(s), which in turn can provide a benefit to the validators. This benefit might be direct, coming from the sale of products or services, or indirect, think for instance to a public body promoting some service linked to its mission. An assessment of validators before they are added to the system is necessary and allows for performing risk analysis, estimating the probability that validators might turn off their node within one or more years, and thus computing the minimum number of them needed to guarantee the persistence of the dApp. Clearly, in the case that some validators leave, this should trigger a search for new validators to keep the dApp stable.

Regarding external users, there are two possibilities:

- everyone can access the blockchain—in this case, the validators allow public access to the SC interfaces and to the explorer (see later);
- the access is reserved for authorized users—in this case, authentication and access control must be provided by validators before users can access the dApp.

The components of the architecture are:

- The Ethereum PoA blockchain is shown as a network of validators and regular nodes on the left.
- An external system, called the App System, holds the data and applications not residing in the blockchain; it is shown in the center.
- The terminals of operators and external users (top and bottom of the figure), running dApp software providing the user interface, and able to manage the private keys of operators.
- A system to perform identity management and access control, integrated into the App System and possibly also using an SC.
- A link to a public blockchain (in this case Ethereum) to periodically write the hash digest of the last block locally mined.
- An Explorer running on one or more nodes holding a copy of the blockchain to browse the actual state of the blockchain without the mediation of the user interface.
- Links to IoT devices, which send data to the blockchain or receive commands from the system.

Let us describe these components in greater detail.

5.1. Ethereum PoA blockchain

Ethereum foundation and also Ethereum implementation under the Hyperledger project (called Besu), offer Clique as one of the preferred consensus protocols. Clique is a form of PoA, a customized form of Proof-of-Stake where the identity and reputation of the validator perform the role of stake instead of stake with some monetary value. As already stated in Section 2, Clique is one of the best and most popular consensus algorithms for permissioned blockchain, so we decided to use it in our architecture.

In Clique, each validator is not allowed to validate two consecutive blocks, in order to preserve equilibrium among validators and to minimize damage if a validator becomes malicious and validates a wrong block—in which case it is quickly spotted by other validators, ousted from the validators' set using a vote, and the wrong block is eliminated from the valid blockchain through a fork. Also, regular nodes may be present, holding a copy of the blockchain.

The validators also create Ethers expendable in the consortium blockchain, which we call “Local Ethers” (LOCETHs). LOCETHs do not have monetary value and cannot be exchanged against other currencies. However, they must be used to send transactions able to change the status of the system, as further explained in Subsections 5.3 and 5.4.

All nodes hold the blockchain enabling software, which includes the Ethereum Virtual Machine, running SCs. The SC bytecode, endowed with its permanent data (storage), is stored in the blockchain and is loaded into the node memory for its execution. All the nodes execute every SC, and execution results must be the same for all nodes. Hence the impossibility for SCs to access the external world. They can access only their data and other SCs stored in the blockchain, which are the same in all nodes.

5.2. App system

Another key dApp component is a software system running on mobile devices and/or on servers, possibly on the Cloud, which we call the “App System”, following the nomenclature of the ABCDE method to develop dApps [33]. It holds the information that cannot stay in the blockchain because it is too large, or for privacy reasons. The App System exchanges information with users and external systems and devices and performs business computations.

Of course, it is also able to send transactions to the blockchain, having a direct connection with a node and being the owner of an address and of the corresponding private key.

If the dApp must hold large amounts of information, such as documents and images, these documents are stored off-chain on one or more Document Management Systems (DMSs) by the App System. The hash digest of the document and a link to retrieve it can be stored in the blockchain, guaranteeing the date of the document and its integrity. This approach is also called the “Off-Chain Data Storage” pattern [27,34]. In the case of sensitive data stored off-chain, the App System also takes care of managing access rights to them, providing the information only to qualified users.

Saving data in this way is compatible with privacy regulations because no actual data are stored in a transparent medium such as the blockchain. Moreover, huge amounts of data can be managed, stored, and certified, despite the relatively limited room available in blockchains, most of which were never intended to substitute a DMS or a database. In fact, storing large amounts of information on a permissioned blockchain based on Ethereum is not viable for the following reasons: (i) big data means big transactions to write them into the blockchain, which in turn means overladen communications and less performance; (ii) the computation needed to assemble and communicate the block with these transactions again means reduced performance of the system; and (iii) the size of the blockchain, which is an append-only repository, would quickly become huge and impair efficient data retrieval.

To conclude this section, we stress that the App System is not necessarily a single, centralized system, nor does it have to manage a single, centralized database or DMS. The App System is a service that, if needed, can run on several physical or cloud servers. The operators who need to store a document can directly specify the URL of the DMS where to store it, and there can be many of them. For instance, each organization storing data might manage its own DMS, including granting access permission to it. What is important is that the data can be accessed by whoever is entitled to access it and that the access permission is given by the owner of the data, possibly also through the blockchain itself. Also, the database holding the system data might be a decentralized one, like IPFS. An example of medical records management using IPFS is reported in Ref. [35].

5.3. Terminals and apps

This component includes the applications, running on PCs and/or mobile terminals, which enable interaction with human users. For external users, it can be a simple app, that is able to connect to a blockchain node or to an authentication server and to show the user the requested information gathered from the blockchain and/or from the App Server. In Ethereum, all users can send "view" queries, which return information from the SCs without changing the blockchain and which cost no gas.

For operators, the app includes a wallet, which is software able to generate and store the private key associated with the operator's blockchain address, to create transactions, sign them with the private key, and send them to a node. The operator's private key is unblocked by a password and possibly by the very ownership of a mobile phone. In this way, the identity of the sender of the transaction is guaranteed, in a way compatible with European Union eIDAS Regulation [36].

Since operators send write transactions to the blockchain, their wallet also holds LOCETHs, which are used to pay the needed gas, thus acting like a true cryptocurrency wallet. In this way, writing can be controlled by the validators, providing LOCETHs only to approved organizations and in the proper amount. These organizations will in turn send the LOCETHs to their operators' wallets to enable writing to the blockchain.

If an organization opts out of the system, they will have to return the residual LOCETHs and will not be provided with more, thus effectively stopping their use of the system.

The operator's app will also facilitate the data input and control operations the operator is in charge of. Depending on the specific applications, the app is able to exchange data with the blockchain (by sending transactions) and/or with the App System.

5.4. Identity management and access control

In the previous section, we stressed how the apps running on mobile or PC terminals can work as a wallet, guaranteeing the association between the address and the ownership of the corresponding private key.

When operators register to the system, they generate the address, and a register managed by the App System associates the address with their identity (name, SSN, and other data). This association can be made public—for instance, to identify an authorized auditor or the organization that the operator is the legal representative of—or not.

Additionally, the system must register the access permissions of the user. This can be done in a traditional way, using access control lists or role-based access control managed by a server (which is part of the App System), or through a dedicated SC able to associate the users' addresses to their permissions.

Further access control, as cited before, can be granted by endowing operators with LOCETH, the gas enabling the sending of transactions. This functionality of the system works in the following way:

1. LOCETHs generated by validators are sent to a system wallet.
2. From this wallet, LOCETHs are sent to the wallets of the organizations that need to use the system in proper amounts. In this way, it is possible to control system usage and bill for it.
3. The organizations send the LOCETHs to the wallets of their operators, thus enabling them to send transactions. The amounts depend on the actual number and complexity of transactions to be sent.
4. When the LOCETH level in a given operator's wallet falls below a given threshold, a request to top up the wallet is sent to the wallet of their organization.
5. Organizations can receive LOCETHs by the system wallet upon request or according to an agreed schedule.

5.5. Explorer and anchoring on a public blockchain

In our architecture, one or more blockchain nodes provide an Explorer, which is software that allows its users to access blockchain transactions and to inspect the source code of SCs. For Ethereum, there are various open-source Explorers available for this task. Among them, we may quote BlockScout, Expedition Block Explorer, and Alethio.

In this way, the transparency of the consortium blockchain equals that of a public one because all transactions and accounts, including those of SCs, can be independently inspected. The use of the Explorer can be granted to everyone or only to registered users with the proper credentials, depending on the specific system.

If the Explorer guarantees a transparency similar to that of public blockchains, anchoring to a public blockchain guarantees a similar level of immutability. The idea is that, from time to time, the hash digest of the last block validated in the permissioned blockchain is written into a public blockchain. This idea has already been applied, especially in the field of distributed data storage solutions [37]. The time interval might be 12 h or 24 h, or even less.

The public blockchain used to anchor the permissioned one can be Ethereum but also Bitcoin or others, provided they are consolidated enough and stable. The cost of each registration, at the current fee rate, is of the order of a few USD or less, so it should not be an issue for an industrial initiative. The transaction towards the public blockchain is sent by an address managed by the App System, which is published. Clearly, these registrations need to manage a "true" cryptocurrency wallet, which is not related in any way to the wallets holding LOCETHs.

In this way, everyone can access the last registration on the public blockchain, using a public Explorer on it, which can access all the transactions sent by a given address. Then, it is possible to verify that the registered hash digest is equal to that of a block validated in the permissioned blockchain at a date and time immediately prior to the public registration. The hash digest of local blocks can be browsed using the local Explorer.

The combination of the immutability of the public blockchain and of the transparency of both public and permissioned blockchains, made possible by the respective Explorers, make the latter as immutable and transparent as the former.

5.6. IoT devices

The Internet of Things (IoT) is the extension of the Internet to connected physical objects that can be monitored, controlled, or interacted with to enable ubiquitous industrial services. Examples of IoT industrial use are freight transportation, automatically registering temperatures, position, arrival times, and status of shipping containers and trucks as they move; tracking components in aircraft, automotive, or other industries, which is critical for both safety and regulatory compliance; supply chain and digital product passport digitalization and control; logging of operational maintenance data, and many others.

The interaction between blockchain and IoT has been proposed since the introduction of SCs for two main reasons. The first is because the blockchain can provide IoT devices with security and the ability to be tamper-proof. The second is the fact that a blockchain is distributed, and an IoT device can connect to any of its nodes, avoiding the bottleneck of a single access point.

An IoT sensor can be provided with an address, a private key, and a connection to the blockchain, and thus be able to send its data through a transaction, which guarantees the timestamp and immutability of the registration. For this purpose, many initiatives aim to develop and field blockchains specifically suited to IoT management, such as IOTA and IoTex.

Things, however, are not so simple because the number of IoT devices can be huge, and the rate of transactions coming from each of them can be high, stressing both the throughput and the size of the blockchain. To solve this issue, sets of IoT devices are connected to some flexible and robust cloud computing environments that are able to process and manage IoT services. This solution is called the “Cloud of Things” (CoT), and its integration with the blockchain (BCoT) is the subject of a large amount of research, aptly reviewed and summarized by Nguyen et al. [38].

In Fig. 1, we show both single IoT devices directly connected to the blockchain and a set of them connected to a CoT, a service running on the cloud, gathering the IoT data, and registering them to the blockchain—thus becoming a BCoT. The IoT data are typically not entirely registered on the blockchain, but only a digest of them is written. If needed, the raw data can be stored in the cloud, or in a server of the App System, which is drawn as connected with a line to the CoT.

6. Case study—certification of an agri-food chain

Together with our spinoff, FlossLab Ltd., we have already implemented a prototype of the system, aiming to track the provenance and events of a food supply chain. This system is part of research projects performed by FlossLab Ltd. and by our department, funded by the Sardinia Region and by the Italian Ministry for Economic Development. The project developed a configurable dApp to build agri-food provenance and quality certification.

In this system, external users are the buyers of food, so read-only access is granted to everyone. Through a QR code on the label and an app, external users can access the dApp and verify all transformations and relevant events related to the product, registered by identified actors in the supply chain.

An advanced user can also independently access the permissioned blockchain and verify its data using the Explorer. The trust is provided not only by the blockchain immutability and tamper-proof features but also by assertions made by qualified professional auditors and analysis laboratories, whose identity is guaranteed.

The goals of the system are:

- document all relevant events of agri-food production in a transparent and trusted way, using a smartphone or tablet in an easy and intuitive way;
- allow laboratories and agricultural professionals to certify the products, proving their identity;
- allow both manual recording and automatic recording by IoT devices;
- keep track of the quantities produced so that these cannot be increased by introducing products of non-certified origin;
- allow auditors, retailers, and consumers to know the trusted history of the products purchased from the field to consumption.

Since agri-food production processes typically differ greatly, the system can be configured and assembled to match a specific production process. The basic concepts are Operator (Farmer, Producer, Winemaker, Cheesemaker, Auditor, Analysis Lab, Retailer, etc.), Land, Harvest, Product, Transformation, Event.

The key information is stored in a set of SCs, created for each product, and linked together. The events are recorded inside each SC; some transformation events can create or merge products, controlling their quantity.

Fig. 2 shows a piece of the Solidity code of SC “Product”, which represents a generic agri-product. The events are represented by the record “Event”, stored in a mapping inside the SC. In addition to the data common to all events (type, timestamp, registrant, and generic description), the field “_parameters” can hold one or more specific parameters, described by their name, type and value and encoded in bytes.

The system uses the Proxy pattern [34] to store the products, to save space and to allow easier code updating. The factory of Products is the SC “Producer”, whose code is partially shown in Fig. 3. In particular, we show the method “createProduct”, which in fact creates an instance of the contract “ProxyTarget”, which is the actual proxy for the Product.

The agri-food production process is described using a domain-specific language, in the form of pre-defined tables, which are subsequently translated into a json format. These tables, easily understandable and editable using a spreadsheet also by domain experts, define producers, operators, products, events, tokens, as well as their relationships and constraints. Their consistency is then checked by the system. This information is used to automatically generate the code of the SCs used for tracking the food supply chain—in particular the event types and their data to encode in the “_parameters” field—and also the user interface of the apps used by operators and customers.

The agri-food process and its tracking proceed through different levels. The first is the physical level (fields, plants, products, packed products, etc.).

The second level is digital data associated with entities of physical level (documents, certifications, pictures, etc.), which is stored in the cloud but can be digitally signed and registered in the blockchain using the “Off-Chain Data Storage” pattern quoted above.

The third level is that of tokens, tracking physical products. The fourth level is the registrations on the blockchain.

Fig. 4 shows a wine production process managed by the system. The end customer can open the app, point their smartphone to the QR code of the bottle they are drinking, and see the history of the wine. Alternatively, they might also directly explore the blockchain using the explorer publicly provided by the system.

7. Conclusions and future work

The key reason to use a blockchain is trust. If a system can be developed and deployed by an organization and its users trust this organization, there is no reason to use a blockchain.

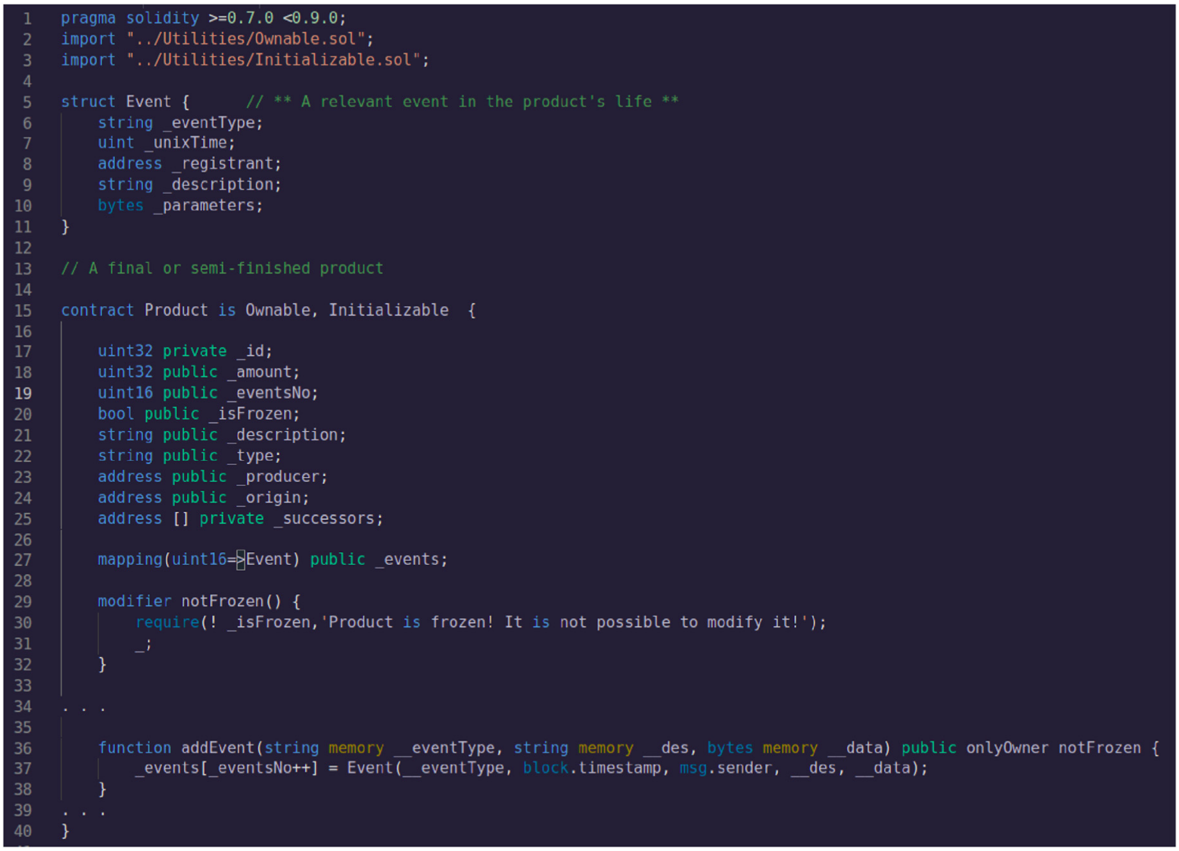
In the case that it is not possible to trust a single organization managing the system, which should be open to all participants—some of whom might try to attack or exploit the system—a public blockchain is the choice. In managing digital currencies and tokens, public blockchains like Bitcoin, Ethereum, and many others have proved to be very effective and reliable.

If the system to develop deals with contractual relationships between participants, does not directly manage digital currencies, and there is no single operator who has everybody's trust, a permissioned blockchain is a typical choice. There are many possible platforms and architectures to develop such a system, so we proposed an evaluation framework to ease the choice, which extends and blends the criteria of existing frameworks.

We also specified in detail an architecture for industrial dApp systems, which again clarifies, extends, and merges existing ideas and patterns in a comprehensive approach. It is based on Ethereum software, using a PoA consensus mechanism, which is fast and energy-saving. The described architecture guarantees the same level of trust and transparency as the public blockchain it is anchored to, allowing much better performance and scalability at a low, predictable cost.

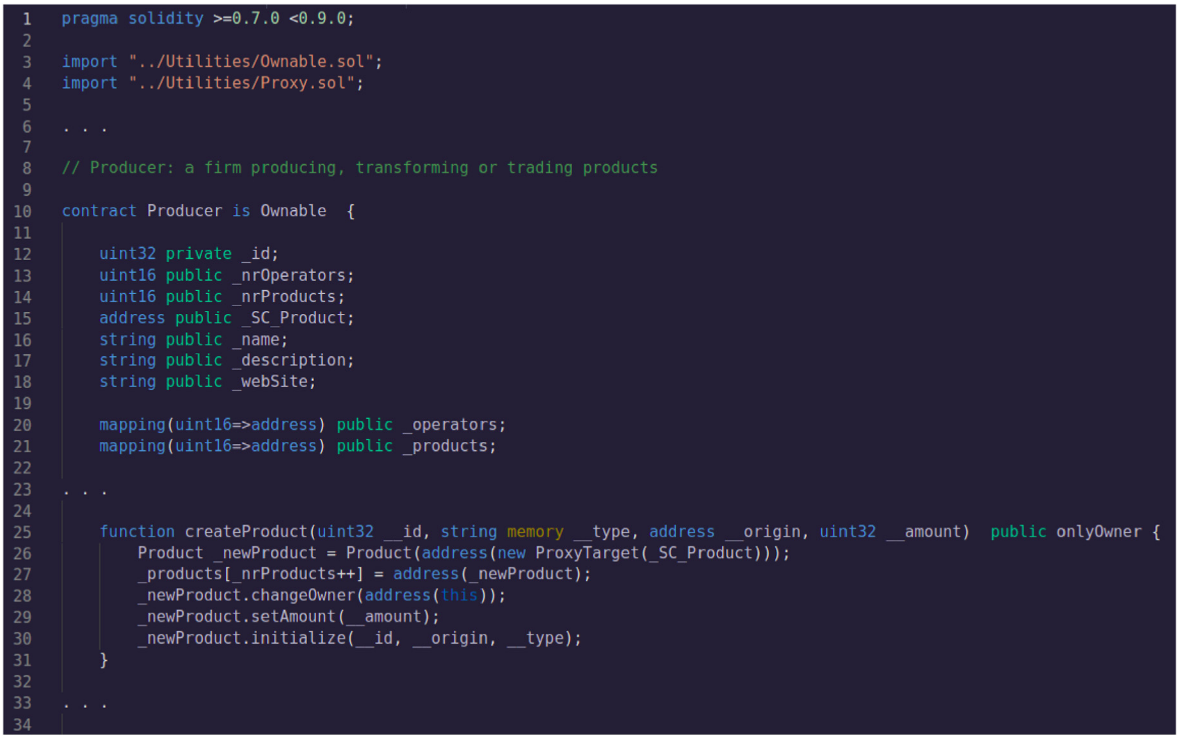
At the same time, it encompasses compliance with privacy regulations, preserving the same level of privacy granted by a private blockchain, and enables the consortium to set different access permissions for different users. The control of writing rights is also performed by means of the local Ethers produced by validator nodes, embracing the advantages of a public blockchain and those of a private one.

We used Ethereum PoA as a reference blockchain, but in principle, it could be substituted by any blockchain provided with the possibility to install an explorer. The proposed architecture is already used in some industrial projects, among which we may quote Etherna, a BaaS (Blockchain as a Service) product, which allows and encourages customers to set up and run their own nodes [39]. Depending on the participants' commitment, these nodes can even be validators.



```
pragma solidity >=0.7.0 <0.9.0;
import "../Utilities/Ownable.sol";
import "../Utilities/Initializable.sol";

struct Event {      // ** A relevant event in the product's life **
    string _eventType;
    uint _unixTime;
    address _registrant;
    string _description;
    bytes _parameters;
}

// A final or semi-finished product

contract Product is Ownable, Initializable  {

    uint32 private _id;
    uint32 public _amount;
    uint16 public _eventsNo;
    bool public _isFrozen;
    string public _description;
    string public _type;
    address public _producer;
    address public _origin;
    address [] private _successors;

    mapping(uint16=>Event) public _events;

    modifier notFrozen() {
        require(! _isFrozen,'Product is frozen! It is not possible to modify it!');
        _;
    }

. . .

    function addEvent(string memory __eventType, string memory __des, bytes memory __data) public onlyOwner notFrozen {
        _events[_eventsNo++] = Event(__eventType, block.timestamp, msg.sender, __des, __data);
    }
. . .
}
```

Fig. 2. Some snippets of the Solidity code of the SC Product, also showing the Event data structure and its usage.



```
pragma solidity >=0.7.0 <0.9.0;

import "../Utilities/Ownable.sol";
import "../Utilities/Proxy.sol";

. . .

// Producer: a firm producing, transforming or trading products

contract Producer is Ownable  {

    uint32 private _id;
    uint16 public _nrOperators;
    uint16 public _nrProducts;
    address public _SC_Product;
    string public _name;
    string public _description;
    string public _webSite;

    mapping(uint16=>address) public _operators;
    mapping(uint16=>address) public _products;

. . .

    function createProduct(uint32 __id, string memory __type, address __origin, uint32 __amount)  public onlyOwner {
        Product _newProduct = Product(address(new ProxyTarget(_SC_Product)));
        _products[_nrProducts++] = address(_newProduct);
        _newProduct.changeOwner(address(this));
        _newProduct.setAmount(__amount);
        _newProduct.initialize(__id, __origin, __type);
    }

. . .
```

Fig. 3. Some snippets of the Solidity code of the SC Producer, showing the usage of the Proxy pattern when a product is created.

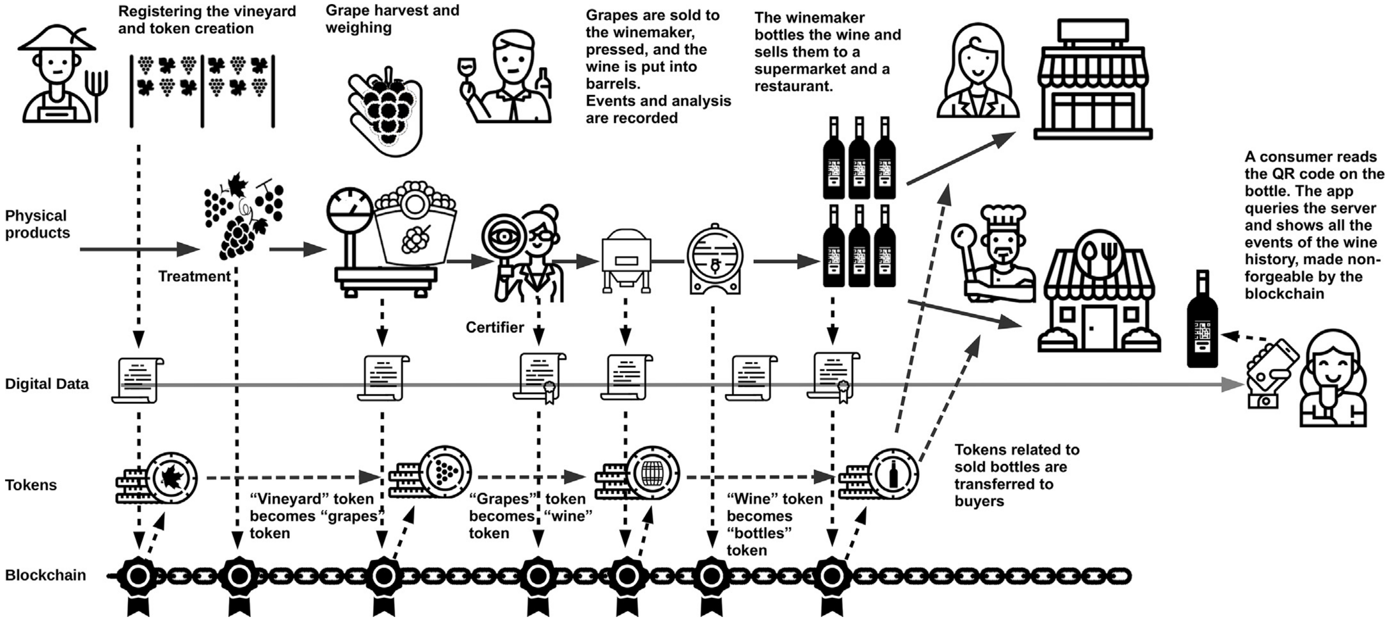

Registering the vineyard and token creation
Grape harvest and weighing
Grapes are sold to the winemaker, pressed, and the wine is put into barrels. Events and analysis are recorded
The winemaker bottles the wine and sells them to a supermarket and a restaurant.
A consumer reads the QR code on the bottle. The app queries the server and shows all the events of the wine history, made non-forgeable by the blockchain
Physical products
Treatment
Certifier
Digital Data
Tokens
"Vineyard" token becomes "grapes" token
"Grapes" token becomes "wine" token
"Wine" token becomes "bottles" token
Tokens related to sold bottles are transferred to buyers
Blockchain

Fig. 4. The agri-food tracking system applied to wine production.

Presently, we are working with a set of Sardinian institutions and firms to start a consortium blockchain as described above, with the aim of certifying the provenance and quality of local products.

We are also working on extending the architecture, enabling multiple blockchains to communicate and exchange transactions. This functionality is obtained by defining "Edge nodes", which are validators in a blockchain but also have the credentials and gas to send transactions to other blockchains through the Edge nodes of the latter. This allows the approach to scale virtually without limits, adding new nodes and new blockchains.

The application we are targeting is the Digital Product Passport (DPP), which is part of the European Union Circular Economy Action Plan [40]. A DPP is a combination of (1) a unique product identifier; (2) data collected by different value chain actors related to this unique identifier; and (3) a physical link (tagging) between the product and the data. Note that a final industrial product will often be an assembly of complex parts, each in turn having its DPP.

An architecture like the one proposed in this paper is very suited to DPP management, with multiple instances aimed at managing the supply chains and certifying the quality of the various sub-products and the final product. The final blockchain system would be devoted to tracking and certifying the useful life of a product, including maintenance and repairs, and the operations on its parts after its disposal, tracking reuse, recycling, and final disposal. Each dApp instance tracking one or more parts would manage their unique identifiers, and the various dApps should be able to easily exchange data, thus providing a complete DPP of the product and its parts in a tamper-proof and transparent way.

Fundings

This work was partially funded by the "MoniFive—MONItoraggio distribuito sicuro, affidabile ed intelligente su tecnologie 5G: applicazione alla mobilità ed al servizio idrico" project, funded by the Ministero dello Sviluppo Economico—Asse II - Progetti di ricerca e sviluppo Programma di supporto Tecnologie emergenti (FSC 2014–2020) del Piano Investimenti per la diffusione della banda larga ai sensi della lettera c) della delibera CIPE n. 61/2018, CUP: F94I20000130006 and by the ABATA project (Application of Blockchain to Authenticity 1370 and Traceability of foodstuffs), funded by the Ministero dello Sviluppo Economico, National Operational Program "Enterprises and Competitiveness", project No. F/200130/01–02/X45, CUP: B31B19000650008.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] S. Nakamoto, Bitcoin: a peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf, 2008. (Accessed 23 December 2021).

[2] N. Szabo, Smart contracts: formalizing and securing relationships on public networks, First Monday 2, https://firstmonday.org/ojs/index.php/fm/article/view/548. (Accessed 14 March 2022).

[3] G. Wood, Ethereum: a secure decentralised generalised transaction ledger. https://ethereum.github.io/yellowpaper/paper.pdf, 2014. (Accessed 14 March 2022).

[4] M.E. Peck, Blockchain world—do you need a blockchain? This chart will tell you if the technology can solve your problem, IEEE Spectrum 54 (10) (2017) 38–60.

[5] K. Wüst, A. Gervais, Do you need a blockchain?, in: 2018 Crypto Valley Conference on Blockchain Technology (CVCBT); 20–22 Jun 2018; Zug, Switzerland IEEE, Piscataway, NJ, USA, 2018, pp. 45–54.

[6] V. Hassija, S. Zeadally, I. Jain, et al., Framework for determining the suitability of blockchain: criteria and issues to consider, T. Emerg. Telecommun. T. 32 (10) (2021) e4334.

[7] B. Koteska, E. Karafiloski, A. Mishev, Blockchain implementation quality challenges: a literature review, in: Proceedings of SQAMIA 2017: 6th Workshop of Software Quality, Analysis, Monitoring, Improvement, and Applications; 11–13 Sep 2017; Belgrade, Serbia, CEUR Workshop Proceedings, 2017.

[8] B.A. Scriber, A framework for determining blockchain applicability, IEEE Software 35 (4) (2018) 70–77.

[9] S. Maranhão, J.-M. Seigneur, R. Hu, Towards a standard to assess blockchain & other dlt platforms, Tech. rep., ITU. https://archive-ouverte.unige.ch/unige:112558, 2019. (Accessed 14 March 2022).

[10] S.N.G. Gourisetti, M. Mylrea, H. Patangia, Evaluation and demonstration of blockchain applicability framework, IEEE T. Eng. Manage. 67 (4) (2020) 1142–1156.

[11] R. Colomo-Palacios, M. Sánchez-Gordón, D. Arias-Aranda, A critical review on blockchain assessment initiatives: a technology evolution viewpoint, J. Softw.: Evol. Proc. 32 (11) (2020), e2272.

[12] M. Garriga, S.D. Palma, M. Arias, et al., Blockchain and cryptocurrencies: a classification and comparison of architecture drivers, Concurr. Comp.: Pract. Ex. 33 (8) (2021), e5992.

[13] F. Wessling, C. Ehmke, M. Hesenius, et al., How much blockchain do you need? Towards a concept for building hybrid dapp architectures, in: WETSEB 2018-1st International Workshop on Emerging Trends in Software Engineering for Blockchain; 27 May–3 Jun 2018; Gothenburg, Sweden, IEEE, Piscataway, NJ, USA, 2018, pp. 44–47.

[14] M. Wöehrer, U. Zdun, Architectural design decisions for blockchain-based applications, in: The 3rd IEEE International Conference on Blockchain and Cryptocurrency (ICBC); 3–6 May 2021; Sydney, Australia, IEEE, Piscataway, NJ, USA, 2021, pp. 1–5.

[15] S.S. Panda, B.K. Mohanta, U. Satapathy, et al., Study of blockchain based decentralized consensus algorithms, in: TENCON 2019—2019 IEEE Region 10 Conference; 17–20 Oct 2019; Kochi, India, IEEE, Piscataway, NJ, USA, 2019, pp. 908–913.

[16] M. Castro, B. Liskov, Practical byzantine fault tolerance, in: Proceedings of the Third Symposium on Operating Systems Design and Implementation; 22 Feb 1999; New Orleans, LA, USA, USENIX Association, Berkeley, CA, USA, 1999, pp. 173–186.

[17] L. Gerrits, C.N. Samuel, R. Kromes, et al., Experimental scalability study of consortium blockchains with bft consensus for iot automotive use case, in: Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems, Association for Computing Machinery; 15–17 Nov 2021; Coimbra, Portugal, ACM, New York, NY, USA, 2021, pp. 492–498.

[18] G. Shapiro, C. Natoli, V. Gramoli, The performance of byzantine fault tolerant blockchains, in: 2020 IEEE 19th International Symposium on Network Computing and Applications (NCA); 24–27 Nov 2020; Cambridge, MA, USA, IEEE, Piscataway, NJ, USA, 2020, pp. 1–8.

[19] A. Ahmad, M. Saad, J. Kim, et al., Performance evaluation of consensus protocols in blockchain-based audit systems, in: 2021 International Conference on Information Networking (ICOIN); 13–16 Jan 2021; Jeju Island, Republic of Korea, IEEE, Piscataway, NJ, USA, 2021, pp. 654–656.

[20] C.N. Samuel, S. Glock, F. Verdier, et al., Choice of ethereum clients for private blockchain: assessment from proof of authority perspective, in: 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC); 3–6 May 2021; Sydney, Australia, IEEE, Piscataway, NJ, USA, 2021, pp. 1–5.

[21] G. Fenu, L. Marchesi, M. Marchesi, et al., The ico phenomenon and its relationships with ethereum smart contract environment, in: 2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE); 20–20 Mar 2018; Campobasso, Italy, IEEE, Piscataway, NJ, USA, 2018, pp. 26–32.

[22] Y. Wang, J.H. Han, P. Beynon-Davies, Understanding blockchain technology for future supply chains: a systematic literature review and research agenda, Supply Chain Manag.: Int. J. 24 (1) (2019) 62–84.

[23] Q.H. Lu, X.W. Xu, Adaptable blockchain-based systems: a case study for product traceability, IEEE Software 34 (6) (2017) 21–27.

[24] D. Di Francesco Maesa, P. Mori, Blockchain 3.0 applications survey, J. Parallel Distr. Comput. 138 (2020) 99–114.

[25] S.V. Akram, P.K. Malik, R. Singh, et al., Adoption of blockchain technology in various realms: opportunities and challenges, Security and Privacy 3 (5) (2020) e109.

[26] E.J. De Aguiar, B.S. Faiçal, B. Krishnamachari, et al., A survey of blockchain-based strategies for healthcare, ACM Comput. Surv. 53 (2) (2021) 1–27.

[27] X. Xu, I. Weber, M. Staples, Architecture for Blockchain Applications, 1 st, Springer, Cham, Switzerland, 2019.

[28] State of the dapps website. https://www.stateofthedapps.com/stats, 2021. (Accessed 14 March 2022).

[29] J. Polge, J. Robert, Y. Le Traon, Permissioned blockchain frameworks in the industry: a comparison, ICT Express 7 (2) (2021) 229–233.

[30] Ethereum 2.0 website. https://ethereum.org/en/eth2, 2021. (Accessed 14 March 2022).

[31] Hyperledger fabric. https://www.hyperledger.org/use/fabric, 2021. (Accessed 14 March 2022).

[32] D. Li, W.E. Wong, J. Guo, A survey on blockchain for enterprise using hyperledger fabric and composer, in: 2019 6th International Conference on Dependable Systems and Their Applications (DSA); 3–6 Jan 2020; Harbin, China, IEEE, Piscataway, NJ, USA, 2020, pp. 71–80.

[33] L. Marchesi, M. Marchesi, R. Tonelli, Abcde—agile block chain dapp engineering, Blockchain: Res. Appl. 1 (1) (2020) 100002.

[34] X. Xu, C. Pautasso, L.M. Zhu, et al., A pattern collection for blockchain-based applications, in: EuroPLoP '18: Proceedings of the 23rd European Conference on Pattern Languages of Programs; 4–8 Jul 2018; Irsee, Germany, ACM, New York, NY, USA, 2018, pp. 1–20.

[35] J. Sun, X. Yao, S. Wang, et al., Blockchain-based secure storage and access scheme for electronic medical records in ipfs, IEEE Access 8 (2020) 59389–59401.

[36] Regulation (eu) no 910/2014 of the european parliament and of the council. https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32014R0910&from=EN, 2014. (Accessed 14 March 2022).

[37] H. Huang, J.R. Lin, B.C. Zheng, et al., When blockchain meets distributed file systems: an overview, challenges, and open issues, IEEE Access 8 (2020) 50574–50586.

[38] D.C. Nguyen, P.N. Pathirana, M. Ding, et al., Integration of blockchain and cloud of things: architecture, applications and challenges, IEEE.Commun. Surv. Tutorials 22 (4) (2020) 2521–2549.

[39] Etherna blochain as a service. https://www.netservice.eu/en/products-and-solutions/etherna, 2021. (Accessed 14 March 2022).

[40] T. Adisorn, L. Tholen, T. Götz, Towards a digital product passport fit for contributing to a circular economy, Energies 14 (8) (2021) 2289.