Smart Contract Powered Framework for the Next Generation Industry 4.0 Business Model

DENIS STEFANESCU, LETICIA MONTALVILLO, and AITOR URBIETA, Ikerlan Technology Research Centre, Spain
PATXI GALÁN-GARCÍA, Entrii, Spain
JUANJO UNZILLA, University of the Basque Country UPV/EHU, Spain

Blockchain stands as a crucial technology capable of enhancing transparency, security, and efficiency within various sectors. Particularly in Industry 4.0, blockchains can be employed to monitor the movement of goods and materials across the supply chain, ensuring data integrity, transparency, immutability, accountability, and industrial process interoperability. Industry 4.0 companies must interact with various external players, which often requires collaborations and partnerships to access new technologies, shared data and analytics, and required resources for their processes. However, current business process management in the industry is mostly centralized, untraceable, unreliable and lacks automation. Smart contracts hold the potential to tackle these concerns by offering a remarkable level of automation, transparency, and security, while also aiding in regulatory compliance. However, the adoption of smart contracts in Industry 4.0 is still limited due to various obstacles. The challenges facing the deployment of Distributed Ledger Technology (DLT) in Industry 4.0 are multifaceted, encompassing issues such as interoperability across different blockchain systems, scalability and performance constraints, limitations on ensuring data privacy, difficulties in accessing external data sources, and the significant transaction costs often associated with executing smart contracts. To tackle these obstacles comprehensively, we advocate for the development of an interoperable consortium blockchain framework tailored to meet the versatile business processes required by Industry 4.0 enterprises or groups. This proposed architecture, distinct from any specific internal system, incorporates private channels and ensures secure access to external data, thus addressing broader interoperability concerns beyond just smart contracts. Our goal is to establish a comprehensive DLT framework that assures data integrity and traceability throughout its lifecycle, from creation and processing to its final utilization for business insights. The effectiveness of our approach is demonstrated through its application in a real-world industrial scenario, undertaken in partnership with Fagor Automation, a global leader in the industrial sector.

CCS Concepts: • **Security and privacy** → *Distributed systems security*.

Additional Key Words and Phrases: Blockchain, Industry 4.0, Smart Contracts

1 INTRODUCTION

Industry 4.0 represents the progression of automation within manufacturing and various other sectors. This concept is marked by the implementation of technologies such as the Internet of Things (IoT), cloud computing, and data analytics to bolster productivity, efficiency, and innovation [1]. A primary factor propelling Industry 4.0 is the escalating digitalization of manufacturing and other industries. This process entails employing sensors, data analytics, and additional technologies to collect and examine real-time data from machinery, procedures, and

Authors' Contact Information: Denis Stefanescu, distefanescu@ikerlan.es; Leticia Montalvillo, lmontalvillo@ikerlan.es; Aitor Urbieta, aurbieta@ikerlan.es, Ikerlan Technology Research Centre, Arrasate-Mondragon, Spain; Patxi Galán-García, Entrii, Valencia, Spain, patxigg@deusto.es; Juanjo Unzilla, University of the Basque Country UPV/EHU, Bilbao, Spain, juanjo.unzilla@ehu.eus.

ACM 2769-6480/2024/8-ART
https://doi.org/10.1145/3686167

products. The analyzed data is then utilized to refine operations, enhance product quality, and decrease expenses [2].

In the context of Industry 4.0, blockchain stands as a disruptive technology [3]. Blockchain is a type of Distributed Ledger Technology (DLT) that enables secure and transparent recording of transactions and data. A blockchain is composed by a network of nodes that work together to validate and record transactions on a shared decentralized ledger. Within a blockchain system, transactions are assembled into blocks and cryptographically linked to the previous block in the sequence. This creates a secure and tamper-evident record of all transactions, since any attempt to alter a block would require changing all subsequent blocks in the chain, which is highly unachievable [4]. One of the key features of blockchain is decentralization. This characteristic makes it resistant to tampering and fraudulent activities. As a result, blockchain has the potential to amplify transparency, security, and efficiency across various industries, allowing them to function more competently, securely, and with reduced costs [5].

However, applying blockchain in resource constrained environments is not straightforward [6]. For instance, scalability is a significant drawback of blockchain technology [7]. As each node in a blockchain network must validate and record every transaction, the network can quickly become overwhelmed when transaction volume increases. This can result in slow transaction times and elevated fees. Additionally, blockchain technology's energy consumption is a concern [8]. The validation procedure for transactions within a blockchain network generally requires substantial computing resources and energy.

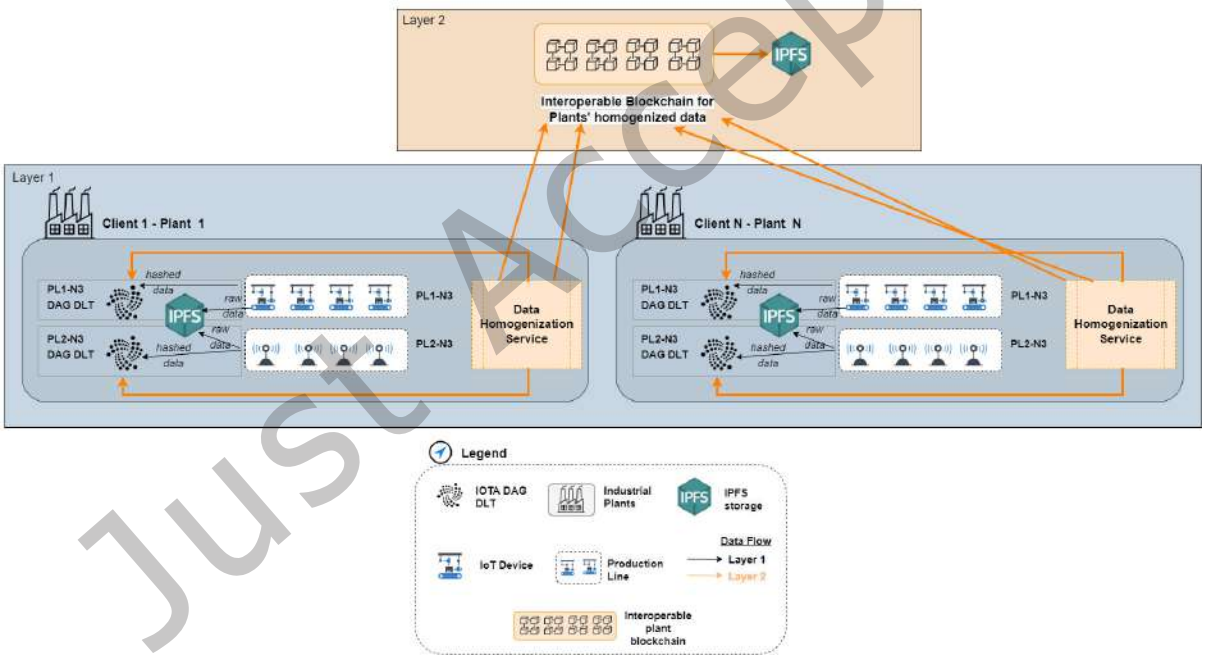

Fig. 1. Industrial DLT-based data processing and homogenization architecture

Therefore, in recent years, alternatives to blockchain technology such as Directed Acyclic Graphs (DAGs) have been developed to address some of these drawbacks. DAGs are a category of distributed ledger technology that enable faster and more efficient transactions in comparison to blockchains. They do not necessitate mining and lack blocks or chains, which allows for faster transaction times and reduced fees [9].

Within the framework of Industry 4.0, blockchain can be employed to monitor the flow of products and resources across the supply chain, facilitating enhanced transparency and efficiency. Furthermore, as shown in a previous work [10] and in Figure 1, blockchain and DAG-type DLTs can also be used to guarantee the integrity of the data throughout the whole process, from when the data is generated from the Industrial IoT (IIoT) devices until it is homogenized and shared across several plants. This approach provides robust protection, especially against data tampering attacks, while also providing transparency, immutability, accountability and interoperability to industrial processes. In addition, auxiliary tools such as the InterPlanetary File System (IPFS)[1] are used in order to reduce the storage burden of the DLTs by keeping the actual data in IPFS and only the IPFS references within the DLTs. Thus, this approach is highly efficient and enough to guarantee industrial data integrity.

However, processing and storing plant homogenized data does not imply the end of the cycle. Generated data must be exploited so enterprises can correctly manage and improve industrial processes and generate value [11]. Industry 4.0 companies are expected to interact with a great variety of external players, including suppliers, customers, and partners. This can include other businesses, government agencies, energy and materials providers, research institutions, and other industry groups. In particular, Industry 4.0 enterprises often engage in collaborations and partnerships to access new technologies, shared data and analytics, and required resources for their processes. Additionally, Industry 4.0 enterprises often rely on external service providers for specialized expertise in areas such as data analytics or cybersecurity.

Currently, the management of business processes in the industry is predominantly centralized, unverifiable, untrustworthy, and lacking automation [12]. This is where the notion of smart contracts emerges. Smart contracts are self-executing agreements with the conditions of arrangements between two or more parties embedded directly into lines of code. Within the scope of Industry 4.0, smart contracts hold the potential to address several pertinent issues, such as inadequate automation, traceability, or data manipulation [13]. By automating the completion and monitoring of transactions, smart contracts deliver a high level of automation, which helps to prevent time loss and human-induced errors. Smart contracts also contribute to enhancing transaction transparency and security [14]. As they are self-executing and transparent, disputing the terms of a smart contract or altering it without other parties' consent becomes challenging. Another issue that smart contracts resolve in Industry 4.0 is regulatory compliance [15]. By streamlining the completion and tracking of transactions, smart contracts can assist businesses in adhering to relevant regulations and standards.

Nonetheless, smart contract business processes adoption by Industry 4.0 enterprises is still uncommon due to several challenges that still need to be solved [11] [16]:

- Lack of interoperability. It is difficult for different organizations to interoperate since there are many different smart contract platforms and data structures.
- Scalability and performance limitations. Smart contracts can be heavy on blockchain networks, which can slow down the performance in high-transaction situations.
- Limited data privacy. Smart contracts are executed on a distributed ledger, which means that all of the data used in the contract is public and transparent. This can be a problem for organizations that handle sensitive or confidential information.
- Limited access to external data. Smart contracts can only work with data that is stored on the ledger on top of which they are written.
- Fees. Some smart contract-compatible blockchains require fees for each transaction. This may incur high costs for enterprises.

Therefore, we aim to solve the aforementioned challenges and extend the architecture from Figure 1 by providing an interoperable and customizable smart contract platform that can be adapted to any business process that could be needed by an Industry 4.0 enterprise or group. The proposed platform is separated from any

[1] https://ipfs.tech/

internal architecture that an enterprise could have, implements private channels and has the ability to access any external data in a secure manner. Thus, we intend to achieve a holistic DLT architecture, where data integrity and traceability are guaranteed throughout the whole process: from when the data is generated, processed and homogenized (as in Figure 1), up until it is exploited for business purposes. Finally, we evaluate the proposed platform by implementing an Industry 4.0 use case.

Specifically, we provide the following contributions:

- Design of an interoperable, customizable and fee-less smart contract platform. This contribution addresses the challenges of interoperability, scalability, and cost. Different organizations, using different smart contract platforms, often find it hard to cooperate due to their varying data structures. Our proposed design allows for adaptable interfaces to accommodate various business processes, facilitating collaboration among diverse enterprises. Scalability issues are addressed by the customizability of our platform, enabling it to handle high-transaction situations without impairing performance. Moreover, the challenge of cost is targeted by ensuring our platform does not require transaction fees, which can often present a substantial burden to businesses.
- Private channel implementation for data privacy. This contribution directly counters the challenge of limited data privacy. As smart contracts execute on distributed ledgers, all data used becomes transparent and accessible, which can be problematic for sensitive information. By implementing private channels, our platform ensures data privacy for organizations, mitigating this concern.
- Incorporation of oracles for external data access. Addressing the challenge of limited access to external data, our platform incorporates oracles, enabling the smart contracts to access data outside the ledger. This feature expands the operational capability of the smart contracts, ensuring their successful execution.
- Implementation of the proposed platform along with a specific use case. Here, we translate our solutions into practice by developing the platform and demonstrating its functionality in a real-world Industry 4.0 use case developed in collaboration with a leading industrial company (Fagor Automation). This allows us to better understand how our platform can meet the challenges and needs of real Industry 4.0 enterprises.
- Evaluation of the platform in terms of performance and security. Lastly, our study examines the platform's effectiveness in overcoming the identified challenges.

The remainder of the paper is organized as follows. In Section 2 we provide an introduction to the technical concepts that appear in this paper. In Section 3 we analyze the related work and perform a comparison against our own. In Section 4 we present the proposed consortium blockchain architecture and explain its components. In Section 5 we describe the implementation of the proposed solution. In Section 6 we discuss the results of the work. Finally, in Section 7 we present the conclusions and future research opportunities.

2 BACKGROUND

In this section, we describe the key blockchain related concepts and frameworks that appear in this work.

2.1 Smart Contracts

Smart contracts are self-executing agreements that are written directly into lines of code. Although they were conceived in 1994 by Nick Szabo [17] and thus are not a recent concept, they became popular with the rise of blockchain approximately less than one decade ago. Consequently, they are typically implemented on blockchains, which allow for secure and transparent execution of the contracts. The code and the agreements contained therein exist over a decentralized network and are stored across a shared, replicated and immutable database. This allows for the removal of intermediaries and the creation of trust-less, automatized schemes that can cover a wide range of applications [18].

Smart contracts must be written in a programming language that is compatible with the blockchain platform they are being implemented on. The most commonly used language for this purpose is Solidity, which is used for Ethereum-based smart contracts [19]. However, many blockchain platforms typically allow the development of smart contracts in many different languages. For example, the Hyperledger Fabric[2] blockchain allows the development of smart contracts (also known in the Fabric ecosystem as "chaincodes") in NodeJS, Go and Java. Another recently popularized language for smart contract development is Rust.

2.2 Blockchain Oracles

An oracle serves as a software or a service that connects a blockchain smart contract to external data sources. It allows smart contracts to access and utilize data that is not stored on the blockchain, such as information from APIs or other external systems, thus enabling the development of decentralized applications capable of interacting with the real world.

Specifically designed for blockchain networks, blockchain oracles supply external data to smart contracts and initiate smart contract execution when specific conditions are fulfilled. They can be classified into two primary categories: off-chain oracles and on-chain oracles [20].

Off-chain oracles do not belong to the blockchain network and lack their own blockchain. They gather data from external sources like APIs or web pages and transmit it to the smart contract on the blockchain. This data can trigger smart contract execution or update the contract's state. However, this kind of oracle may introduce a single point of failure within a blockchain structure.

Conversely, on-chain oracles are part of the blockchain network and possess their own blockchain. They execute smart contracts and update their state based on external data. Additionally, they provide external data to the smart contract and initiate its execution when specific conditions are satisfied. On-chain oracles are generally considered more secure due to their inherent security advantages from decentralized systems.

2.3 Permissioned - Consortium Blockchains

The first blockchains (e.g., Bitcoin, Ethereum) were public and permissionless. However, these blockchains are typically slow when processing data and are not scalable. Moreover, they are not suitable for all use cases, such as business consortia. As a response to the aforementioned issues, private and permissioned blockchains have emerged. Enterprise-oriented permissioned blockchains are commonly called "consortium blockchains".

A consortium blockchain represents a form of blockchain network commonly employed when a group of organizations, including businesses or government agencies, need to exchange information and collaborate on shared objectives while retaining a certain control over network access and validating process [21]. This type of blockchain contrasts with public blockchains such as Bitcoin or Ethereum, where the ledger is accessible to everyone, and anyone can become a validator.

Consortium blockchains typically utilize the same foundational technology as public blockchains, such as cryptography for securing the network and a distributed consensus mechanism for transaction validation. However, consortium blockchains usually employ a different consensus mechanism compared to public blockchains. Due to their permissioned nature, consortium blockchains typically use lightweight consensus mechanisms such as the Practical Byzantine Fault Tolerance (PBFT) [22] consensus, which is designed to be more efficient and faster than the Proof of Work (PoW) algorithm that is commonly used in public blockchains such as Bitcoin. PBFT works by having a designated leader, known as the primary, that is responsible for ordering and broadcasting transactions to the rest of the network. The other nodes then verify the transactions and reach consensus on their validity. PBFT, however, has relatively low scalability since there is a great amount of constant communication between the nodes.

[2]https://www.hyperledger.org/use/fabric

2.4 Blockchain Interoperability

Blockchain interoperability refers to the ability of distinct blockchain networks to communicate and interact with each other. This can be achieved through intermediary gateways or through the use of compatible parallel chains [23].

In parallel chain architectures, a leading chain governs the entire network and offers out-of-the-box compatibility between the rest of the parallel chains that compose the network. Each parallel chain serve a distinct purpose, and includes its unique features and capabilities. Moreover, these types of protocols typically support different types of consensus mechanisms and can be used to create private or public networks. Even though parallel chain protocols offer a straightforward, simple approach to interoperability, they are still limited when interacting with other blockchains that are built by different companies. Furthermore, parallel chain approaches are typically complex. Currently, the most known parallel chain platform is Polkadot [24].

Gateway-based interoperability is a method of connecting different blockchain networks by using an intermediary gateway connector that acts as a bridge between them. This approach allows for the transfer of assets and data between the different networks, enabling interoperability [25]. A gateway can be implemented in many ways, from a simple API connector to more complex systems such as intermediary node networks, to blockchain oracles that are programmed for interoperability tasks. A primary benefit of gateway-based interoperability is its ability to facilitate asset and data transfers between distinct blockchain networks without necessitating intricate mechanisms or extensive cooperation between various blockchain providers. Nonetheless, gateway-based interoperability has its drawbacks, such as the presence of centralized points of failure. Consequently, if the gateway network is compromised, all dependent networks may be impacted. Despite these drawbacks, gateway interoperability remains a straightforward and efficient approach to achieving interoperability. However, the security issues of these mechanisms must be carefully analyzed and prevented.

3 RELATED WORK

In this section, we review previous research on the use and implementation of smart contracts in Industry 4.0 environments. Currently, several noteworthy Industry 4.0 initiatives incorporate smart contracts in industrial settings. This literature analysis aims to examine the state of research concerning smart contract usage in Industry 4.0, while identifying major trends and implementation challenges.

Ye and König [26] explored a payment framework for the construction industry, employing smart contracts connected to digital building information models. Payments, in this context, were set to automatically execute upon condition fulfillment. Although insightful, the work's scope was narrowed to a single business case and the smart contracts were restricted in accessing additional external information. In relation to our work, this highlights the necessity for smart contracts that can access a more comprehensive set of data, enhancing their flexibility and applicability.

Demertzis et al. [27] utilized smart contracts to implement a bilateral traffic control agreement, the detection of anomalies in this setup was based on an AI-trained model. This proposal was aimed at enabling a decentralized and secure platform to carry out transactions in critical infrastructural networks, eliminating the requirement of a central authority. Their work contributes to the understanding of how enhanced security in smart contracts can be achieved. However, their approach heavily relies on AI, which might not be a feasible or necessary solution for all scenarios. This points to the need for developing smart contract systems, such as the one proposed in this work, that focus on inherent security and decentralization mechanisms without necessarily relying on AI.

Garrocho et al. [28] focused on applying smart contracts for machine-to-machine communications within industrial settings. They proposed a smart contract middleware for achieving secure, decentralized industrial communication. Their findings, however, suggest the block times were overly high, rendering the approach

impractical. This emphasizes the criticality of scalability and performance in the development of our smart contract platform.

Dixit et al. [29] designed a simple smart contract-based access control mechanism for Industry 4.0, where the industrial data was stored in an IPFS decentralized database with secure smart contract-based access. This highlights the potential of utilizing smart contracts beyond mere access control, towards the automation of intricate business processes.

Ravi et al. [30] implemented a Hyperledger Fabric blockchain for supply chain management with the purpose of enhancing data integrity, provenance transparency, privacy, and security. Their objective was to assess if permissioned blockchains could make supply chains less susceptible to corruption. However, like the other works discussed, their focus was on a specific case, indicating the need for a more versatile, use case-agnostic smart contract platform.

In Table 1, we provide a comparison between our work and the rest of the related works. Based on the main challenges that we defined in the introduction for the current smart contract platforms, we analyze if the studied proposals are permissioned and fee-less, what is the main purpose of each proposal, if they provide interoperability with other business DLTs, if they provide data privacy via private channels or other mechanism, if the smart contract can access external data and if they offer relatively high performance. We consider a proposal to have a high performance capacity whether it satisfies the needs for which it was designed in terms of throughput and scalability.

As it can be seen in the related works, applying smart contracts in industrial environments in order to decentralize and automatize business processes is still an uncommon practice and has many issues. First, some proposals run up against the typical performance limitations of blockchains. Second, the processed data within the smart contracts is typically raw machine data that is decentralized and secured only in the last phase of the chain, i.e., at the end, when it is involved in the business processes (via smart contracts). Finally, the proposed smart contracts are very limited, as they cannot access external data. And finally, overall, the main issue of the related works is that they aim at designing smart contract schemes for very specific cases, thus greatly limiting the range of applications. Thus, we aim to provide a smart contract-based platform that is efficient, interoperable and not limited only to a few use cases.

Table 1. Related works comparison

	Permissioned and fee-less	**Smart contracts purpose**	**Interoperability**	**Privacy**	**External data accessibility**	**High performance**
[26]	Not specified	Business purposes	×	×	×	Not specified
[27]	✓	IoT data processing	×	✓	×	Not specified
[28]	✓	IoT data processing	×	×	×	×
[29]	✓	Identity management	×	✓	×	✓
[30]	✓	Supply chain traceability	×	✓	×	✓
This work	✓	Business purposes	✓	✓	✓	✓

4 THE ARCHITECTURE

The overall architecture is designed for multiple industrial enterprises to interact with external entities in a secure, fast, cheap and straightforward manner. Specifically, the design of the architecture addresses the challenges mentioned in Section 1 regarding:

- Compatibility or interoperability. The smart contracts and their associated data has to be compatible.
- Performance and scalability. The system must be able to support a relatively high number of simultaneous operations.
- Data privacy. Data privacy must be guaranteed within the blockchain, since business agreements are typically private.
- Access to external data. In order to perform complex business agreements, smart contracts must be able to access external data.
- Incurred costs. A blockchain-based system for business processes should not lead to a considerable increase in costs for companies.

Therefore, in order to tackle these challenges, we establish six main design characteristics. Some of the characteristics that we defined such as the DLT type, permissions level and consensus are required in any DLT design, as stated in the work presented by M. Tabatabaei et al. [31]. The aforementioned work provides a comprehensive survey of DLT architectures, advancing in the standardization of DLT design. Apart from the three characteristics that we mentioned above, we include three more that are not typically mandatory; however, we take them into account in order to solve all the challenges that we mention in the introduction.

(1) The type of the DLT. We analyze the existing DLT types and choose the most appropriate type for the given requirements. There are many type of DLTs that have their own particular advantages and drawbacks.
(2) The consensus mechanism. The consensus algorithm is the core of the DLT, since it prevents malicious behavior in the network. Moreover, consensus has a great impact in the DLT's performance, thus it has to be carefully designed.
(3) The DLT access and permission level (public, private, permissioned, permissionless). We establish whether the ledger will be public or private and permissioned or permissionless. This design choice impacts several aspects of the DLT such as the level of privacy and its overall performance (i.e., private blockchains are typically much faster than public blockchains).
(4) Data privacy mechanisms. We design data privacy mechanisms within the blockchain, in order to enable private agreements between enterprises.
(5) External data access mechanisms. Smart contracts that are unable to access external data are strictly limited, especially when it comes to business agreements. For example, many business agreements require access to external price charts. Therefore, we design an approach that enable smart contracts to use external data.
(6) The interoperability approach. As stated above, the smart contracts and their data must be compatible. Thus, we design an interoperability approach in order to cover a broad range of applications.

Figure 2 shows the overall diagram of the architecture. Within Layers 1 and 2 (depicted in blue and orange) we have the starting context that was explained at the beginning of the paper, in which several industrial plants share a common interoperable Polkadot-based blockchain that contains the homogenized IIoT data. This Polkadot based layer assures seamless interactions between several plants when performing data homogenization and processing. The data homogenization service is based on the Eclipse Unide model, as shown in previous work [10].

Within the green frame we have the proposed platform, in which many types of enterprises connect with each other and execute automatic agreements based on the blockchain smart contract technology. Industrial plants are connected to this platform via their shared homogenized data blockchain since their data is useful for the

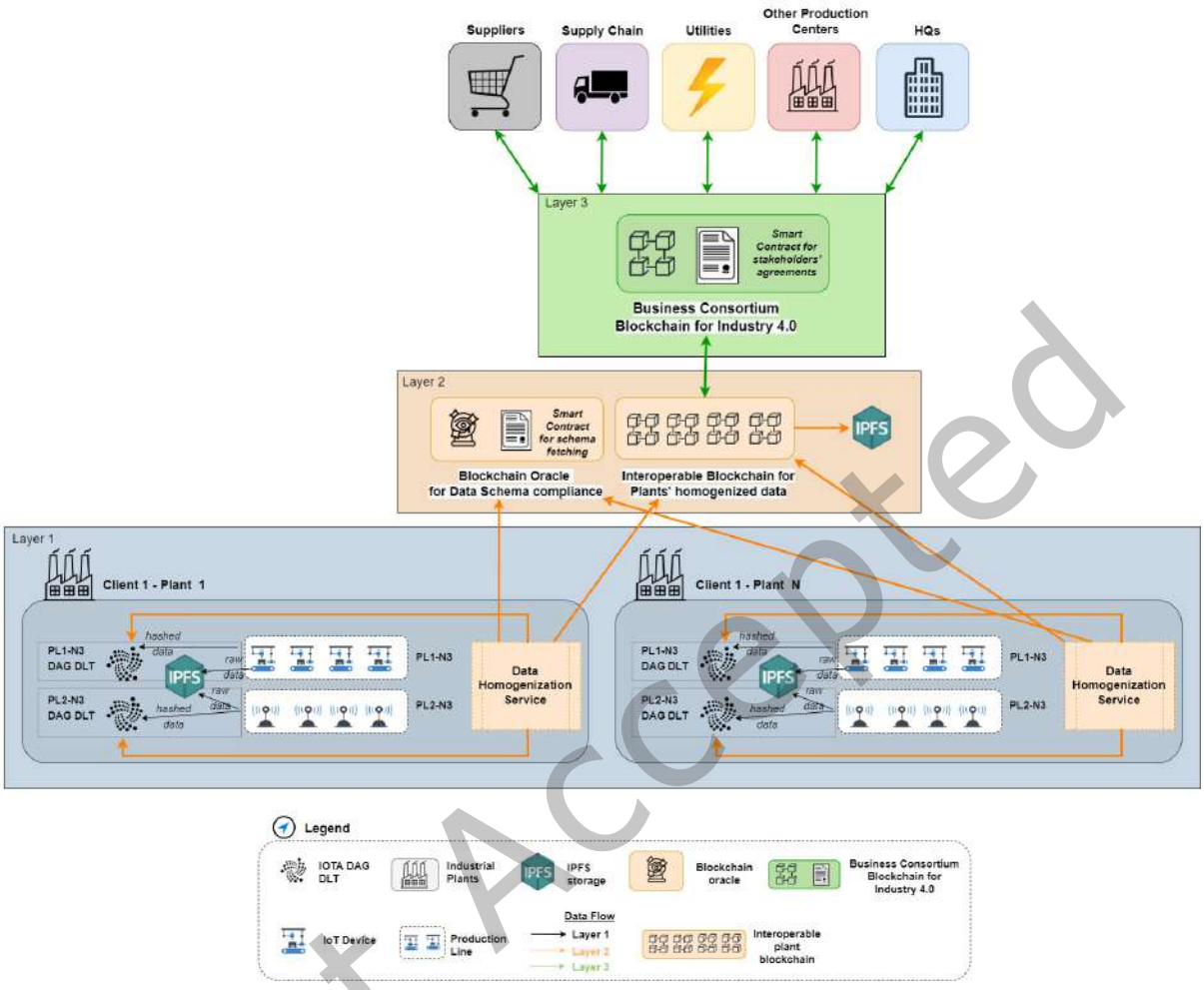

Fig. 2. Overall diagram of the architecture. In white, the starting context [10]. In green, the proposed platform. Arrows represent the data flow.

execution of the agreements, such as, for example, a pay-per-use agreement for industrial machines with external providers. However, each enterprise has total control over the amount of data that it shares for the execution of smart contract agreements and processes.

A smart contract blockchain platform greatly improves business processes between industrial plants and external suppliers, energy (and other utilities) providers, other production centers and business headquarters by providing a secure, transparent, and decentralized way to manage transactions and interactions between different parties. Smart contracts streamline numerous manual processes currently needed to oversee interactions, including payment tracking, transaction verification, and agreement enforcement. Furthermore, the decentralized nature of blockchain minimizes the risk of fraud and guarantees real-time access to identical information for all involved parties.

4.1 DLT type

In this subsection, we discuss the type of DLT that should be implemented in the proposed business-oriented architecture for Industry 4.0. As mentioned above, there are some alternative solutions to blockchain, the most popular and widely adopted alternative being DAG-based DLTs. However, in this architecture, we adopt a smart contract capable blockchain over the other available options.

One of the main advantages of a blockchain over other types of DLTs, such as DAGs, is its ability to provide a higher level of security, immutability and decentralization. This is achieved through the use of consensus algorithms and cryptographic techniques that make it extremely difficult for any one party to alter or tamper with the record of transactions. Other DLTs have not proven yet to be as decentralized or as secure as blockchains. For example, the IOTA DAG platform[3], currently has a centralized coordinator node, and it has experienced many security issues over the last few years [32].

Another advantage of a blockchain is its ability to facilitate smart contract functionality, which is crucial in this work. Smart contracts are stored and replicated on the blockchain network. This allows for the automation of certain processes and the ability to easily track and verify the execution of the contract. In an industrial context, this enables the efficient and secure execution of transactions between different parties in the business ecosystem.

Finally, even though DAGs are known to be much faster than the most popular blockchains, such as Bitcoin, there are some business-oriented permissioned blockchain solutions that also offer a high transaction processing speed. For example, the Hyperledger Fabric blockchain can achieve hundreds or even up to 1,000 transactions per second. Nonetheless, in business environments where digital contracts are executed, the required throughput is not as high as it might need to be at lower levels where the IIoT devices are located [33].

4.2 Network accessibility and permissions

In this subsection, we define whether a public-permissionless or private-permissioned (consortium) blockchain is the most appropriate choice for the proposed scheme.

While public blockchains have many advantages, such as being totally decentralized and providing transparency, they may not be the best choice for the proposed architecture of industrial plants and other entities collaborating and conducting business transactions. Overall, the drawbacks of public blockchain can be summarized as follows [34]:

- Lack of privacy. Public blockchains are open to anyone, which means that sensitive business information may be exposed to competitors or malicious actors.
- Slow transaction processing. Public blockchains have inefficient consensus algorithms, which can lead to longer transaction processing times.
- Lack of scalability. With a large number of participants, public blockchains can become congested, leading to a lack of scalability.
- Lack of governance. With anyone participating, it can be difficult to make decisions about the network and manage it effectively.

On the other hand, consortium blockchains are a more suitable choice in this type of business environment due to the following reasons [34]:

- Increased security. By limiting the number of participants, the risk of malicious actors infiltrating the network is reduced. This ensures that only trusted entities can access and validate transactions.
- Faster transaction processing. With a smaller number of participants, the consensus mechanism can be faster, leading to faster transaction processing times.

[3]https://iota.org

- Better scalability. As the number of participants is limited to the necessary parties only, the network can handle a larger volume of transactions, making it more scalable.
- Better privacy. By allowing only specific participants to access the network, sensitive business information can be kept private.
- Better governance. With a smaller group of participants, it is easier to manage and make decisions about the network.
- Smart contracts capabilities. Most consortium blockchains also provide the ability to execute smart contracts, which can be used to automate business processes and streamline workflows.
- Customization. Consortium blockchains can be customized so that they comply with the specific requirements and necessities of the businesses that are involved.

Overall, a public blockchain is not appropriate for this type of environment because it lacks privacy, has slow transaction processing, lacks scalability and lacks proper governance. A consortium blockchain, on the other hand, is a better option as it allows for a more controlled, private, and efficient network with the ability to provide custom features that are needed in this scenario.

4.3 External Data Access

As stated before, one of the significant challenges that smart contracts are facing in blockchain systems is accessing trustworthy external data. Thus, in this work we make use of blockchain oracles in order to enable smart contracts access external data and improve the usability of the architecture, since oracles provide more reliability, automation, transparency, and interoperability. The oracle mechanism involves the identification and establishment of trusted data sources, the appointment of a set of nodes as oracles, and the use of smart contracts to enforce conditions based on the data provided by the oracles. By having a trusted set of oracle nodes retrieve data from trustworthy sources, we assure the accuracy of the information used in the smart contracts. This, in turn, allows for the automation of contract execution and enforcement, adding transparency to the process as all data can be audited.

In the proposed consortium Industry 4.0 environment, oracles are intended to be used to retrieve relevant information such as market prices, product availability, weather conditions, etc. For example, in an Industry 4.0 supply chain, the oracles can retrieve trustworthy external information on the current price of electricity. Furthermore, in a manufacturing setting, oracles can also retrieve information on the availability of raw materials, triggering the smart contract to order more materials or change production schedules based on the availability. In this way, oracles provide real-time data that can be used to automatically enforce the terms of smart contracts and expand their usability.

4.4 Interoperability

In this subsection, we design an interoperability connector gateway to enable communication between the business blockchain and other blockchains that contain the data from each business. An interoperability gateway connector for the proposed blockchain platform is a software component that allows different blockchain networks to communicate and share information with each other. Connectors enable transferring of assets or data between networks and execution of smart contracts across multiple networks. Thus, the connector acts as a bridge between the different networks, allowing them to interoperate and work together seamlessly. In the case of industrial plants, we would have to connect the proposed business blockchain to the blockchain that contains homogenized plant data, as specified in [10]. In conclusion, this interoperability solution is crucial since smart contracts need data in order to function correctly.

The interoperability gateway that we design includes the following components:

- Management and monitoring. This component is responsible for managing the connector itself. It handles aspects such as configuration, logging, and monitoring the health and performance of the adapter.
- Adapter. This component is responsible for connecting to different blockchain networks and translating the data and commands between them. It can support multiple blockchain protocols according to the needs of the architecture.
- Rules engine. This component is responsible for enforcing any business logic or rules that need to be applied when data is exchanged between the different blockchain networks. The rules engine can be used to enforce compliance rules or to implement smart contract functionality.
- Mapping engine. This component is responsible for mapping data and assets between different networks. It handles aspects such as converting data formats, mapping smart contract functions, and handling any data validation or transformation that is needed.
- Communication layer. This component is responsible for securely communicating with other networks. It handles aspects such as encrypting and decrypting data, establishing secure connections, and handling network failures.
- API. The connector needs an API to enable external applications to actually communicate with it.

Figure 3 depicts the architecture of the interoperability gateway, along with its components. We divide the gateway into three layers (L1, L2 and L3). The middle layer (L2) includes the core components that we described above and provide the basic functionalities of the gateway: the rules engine, the mapping engine and the communication layer. Below (L3) we have the adapter itself, which connects to the core parts from L2 and enables the actual communication between the DLTs. Finally, on the top (L1) we have the monitoring system, which acts as the user interface of the whole gateway, as explained above.

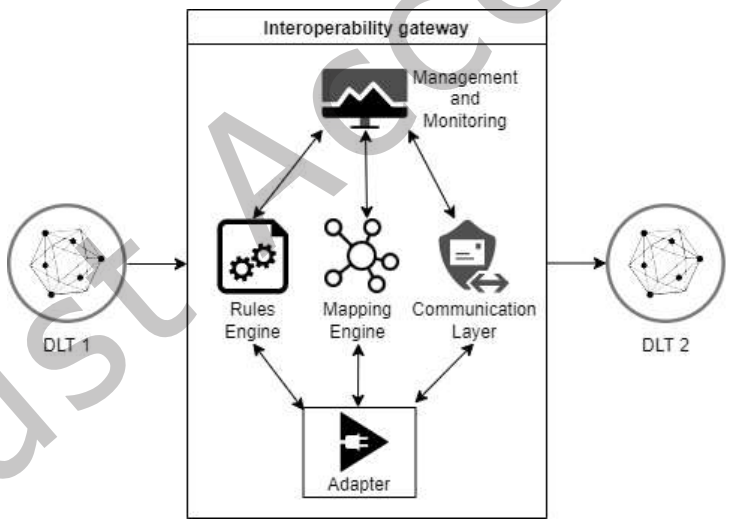

Fig. 3. Interoperability gateway scheme

4.5 Data privacy

Data privacy is a major concern in consortium blockchains as they involve multiple parties sharing sensitive information on a common platform. Private channels help address this issue by allowing communication between a select group of participants [35]. This feature enables organizations to securely and privately exchange sensitive

information without revealing it to the entire network. Specifically, data shared within a private channel is encrypted and solely accessible to channel members, adding another layer of security. This approach contributes to the preservation of sensitive information confidentiality and fosters trust among organizations. Private channels utilize a mix of symmetric and asymmetric encryption, including Advanced Encryption Standard (AES) for data encryption, Elliptic Curve Digital Signature Algorithm (ECDSA) for digital signatures, and Transport Layer Security (TLS) for secure network communication.

An alternative privacy method in consortium blockchains is homomorphic encryption [36]. This encryption technique allows for computations on encrypted data without the need for prior decryption. It offers privacy protection by enabling the processing of sensitive information while remaining encrypted, preventing unauthorized access. However, homomorphic encryption is still in its early development stages and faces several challenges, such as high computational overhead and a narrow scope of practical use cases.

In the case of the proposed Industry 4.0 consortium blockchain, private channels offer a simpler and more practical solution for privacy protection compared to homomorphic encryption. Private channels allow for secure communication between a select group of participants, which meet the privacy requirements for most use cases in industry 4.0. Additionally, private channels are easier to implement and require less computational overhead compared to homomorphic encryption, making them a more feasible option.

4.6 Consensus

When it comes to choosing a consensus algorithm for a consortium blockchain, there are many factors to consider, such as performance, security, scalability, and reliability. In this work, we propose the use of Raft [4].

Raft is a leader-based consensus algorithm that is widely used in distributed systems. It provides a high level of fault tolerance, which means that even if some nodes in the network fail, the blockchain can continue to operate normally. Additionally, Raft has proven to be highly scalable, making it suitable for use in large, complex networks.

When a system starts or the leader fails, a new leader is elected. The leader accepts client requests, appends new entries to its log, and replicates the log entries to follower nodes. Once a log entry is replicated to the majority of nodes, it is marked as "committed", and the nodes execute the operation. Furthermore, to maintain storage efficiency, the algorithm employs log compaction and snapshots, which involve compressing the current log and discarding old log entries.

One of the key advantages of Raft over other consensus algorithms is its simplicity. Unlike other consensus algorithms, such as PBFT, which can be difficult to implement and understand, Raft is straightforward and well-documented. This makes it easier to build and maintain a Raft based blockchain.

Furthermore, Raft is generally considered to have better performance than PBFT, particularly in scenarios with larger networks. PBFT requires more message exchanges and processing time due to its complex message validation process, which becomes a bottleneck in large networks.

Another important factor in choosing a consensus algorithm is security. Raft provides strong security guarantees. In particular, Raft's leader-based approach makes it difficult for attackers to disrupt the consensus process, helping to prevent attacks such as the "double spend" issue. However, in comparison with PBFT, Raft is better suited for scenarios where the primary concern is handling node failures and network partitions rather than malicious behavior. Nonetheless, in the proposed network the presence of a large number of malicious nodes is unlikely since it is a permissioned consortium network between enterprises.

In conclusion, with its combination of reliability, scalability, and security, Raft provides a strong foundation for secure and efficient data sharing among the various stakeholders in the network.

[4] https://raft.github.io/

5 IMPLEMENTATION

In this section we describe the implementation process of the presented solution. We build the architecture on top of Hyperledger Fabric (v2.2), which is an open-source, consortium oriented blockchain that has smart contract capabilities, private channels, customizable consensus and zero fees. This platform is widely used by companies, researchers and developers to create efficient consortium blockchains for a wide range of applications [37].

Running a network with private channels, oracles for trustworthy external data retrieval, smart contracts (chaincodes), and a custom developed connector for interoperability purposes involves several steps. First, we set up a network that includes the required number of organizations, peers, and channels using a modified version of the *network.sh* script from Fabric. Then, we establish the oracle service to allow the overall blockchain architecture and its chaincodes to interact with external data sources when needed. Furthermore, private channels need be created in order to limit the access of private data that is sent between the organizations. We also implement the custom connector that we defined in the previous section to enable interoperability with other blockchains. In this case, we need to achieve interoperability with the DLTs that are being employed in the lower levels of the architecture (i.e., within the industrial plants and production lines). This implementation was carried out using the NodeJS programming language. In order to avoid centralization, we set the interoperability connectors to function on top of the decentralized oracles network.

Finally, the architecture is deployed on a set of physical or virtual machines and tested in a development environment before being rolled out to production. Once the network is live, we monitor its privacy, security and performance capabilities in order to validate the effectivity of the solution and detect possible failures and improvement opportunities. However, it is pertinent to highlight that, in this implementation, speed, although important, is not paramount. The aim is not to deliver the fastest possible solution, but rather to ensure a system that performs reliably and effectively within industrial settings.

To facilitate replication and further development, we have included all the code for both Hyperledger Fabric with Raft consensus and Hyperledger Caliper (the tool we will later use to present the performance analysis) in a GitHub repository[5]. The repository also contains an exhaustive manual for setting up the network and configuring the Raft consensus. This comprehensive documentation ensures that researchers and developers can follow the steps and set up a similar environment for their own testing and development purposes.

The implementation has been carried out on a laptop with Ubuntu operating system, an i7 processor, 16 GB of RAM and a 512 GB SSD drive. As for end users to interact with the network, the requirements are significantly lower as they would interact with the blockchain through applications and interfaces that abstract away most of the technical complexities of the network itself.

5.1 Use case - Fagor Automation machines monitoring

We implement a real-world use case resulting from a partnership between the IKERLAN research center[6] and Fagor Automation[7], a worldwide leading industrial company with extensive experience in the development and manufacture of products for the automation and control of machines.

In this use case, we aim to monitor three parameters from the Computer Numerical Control (CNC) machines that belong to Fagor Automation: installed software, validation code, and parametrization settings. Moreover, we also consider the operation mode of the CNC, specifically the switches between User and Setup/Administrator modes.

The CNC should always operate in User mode as recommended by Fagor Automation. However, a switch to the Setup mode is required to adjust parameters or change the program. If, after these adjustments, the CNC does not

[5] https://github.com/denisro2010/fabriccaliper
[6] https://www.ikerlan.es/
[7] https://www.fagorautomation.com/

return to User mode and remains in Setup mode, this could lead to a potential breach of warranty conditions. The access to the Administrator mode is only granted when enabled by a validation code. Therefore, any "unauthorized" mode changes can also be detected and treated as possible warranty violations. In Administrator mode, the entire disk is unprotected against writing and any modifications can be made, including software installations.

All illegal changes, including these unauthorized mode changes, are registered and processed throughout the industrial plants, and then shared with the proposed business blockchain within a smart contract. This smart contract establishes the warranty status based on a cumulative score given by the level of severity of the detected changes.

To determine whether an installed software is critical or not, we retrieve an external "critical software" list from the machine vendor using trustworthy blockchain oracles. For this task, we set up the official Fagor Automation CNC simulator[8] in the evaluation machine.

As depicted in Figure 4, we implement the script that monitors the industrial machines and detect suspicious changes related to the three aspects that are mentioned above: installed software, validation code and parametrization settings. When changes are detected, the machine data and the generated changes log is processed through the production lines DLT, and then sent to the industrial plant, where the severity score is calculated.

The process begins with an interaction with the blockchain oracles, which provide meaningful data to the smart contracts (chaincodes). The oracles provide a "critical software list", which contains various software that are deemed critical for the machine's functionality and integrity. Since the critical software list is external, it can be modified only by the CNC owner - in this case, Fagor Automation.

The maximum score limit is set at 10. The scoring is calculated based on several factors:

- Trivial Changes (Score: 1): These changes include modifications in system files or processes that are not deemed critical for the operation of the machine, non-critical software updates, or slight tweaks in parametrization settings. These are changes that won't significantly affect the machine's performance or compromise its security.
- Moderate Changes (Score: 5): This category includes changes that can potentially affect the machine's performance but are not likely to compromise its security. For instance, changes in the validation code that do not affect critical processes, installation of software that is not part of the critical software list, or moderate adjustments in parametrization settings.
- Severe Changes (Score: 10): These are critical changes that could potentially lead to a breach of the machine's warranty terms. Examples include the installation or update of critical software without proper validation, alterations in the validation code that affect critical processes, or substantial modifications in parametrization settings that can drastically impact the machine's performance.

The score considers the nature of the change: if there is a modification, deletion, or addition of critical system files or processes, it could attract a higher score. The number of changes is also a factor: if numerous alterations occur in a short time span, the score increases. Furthermore, the score is influenced by the nature of the software itself: changes in some software may be more detrimental than others, based on its role and importance within the system. It is important to notice that in specific cases, such as repeated install and uninstall of the same software, the score would be altered only once.

After detecting and scoring these changes, a log is generated for each machine at the plant level. This log contains information about the changes and the identity of the machine, and is shared with the Fabric business consortium blockchain. This phase includes several functions:

- getCriticalSWListFromOracle(): This function calls upon the blockchain oracles to retrieve an up-to-date critical software list, which is essential for the CNC's operation and security.

[8]https://hmielite.fagorautomation.com/

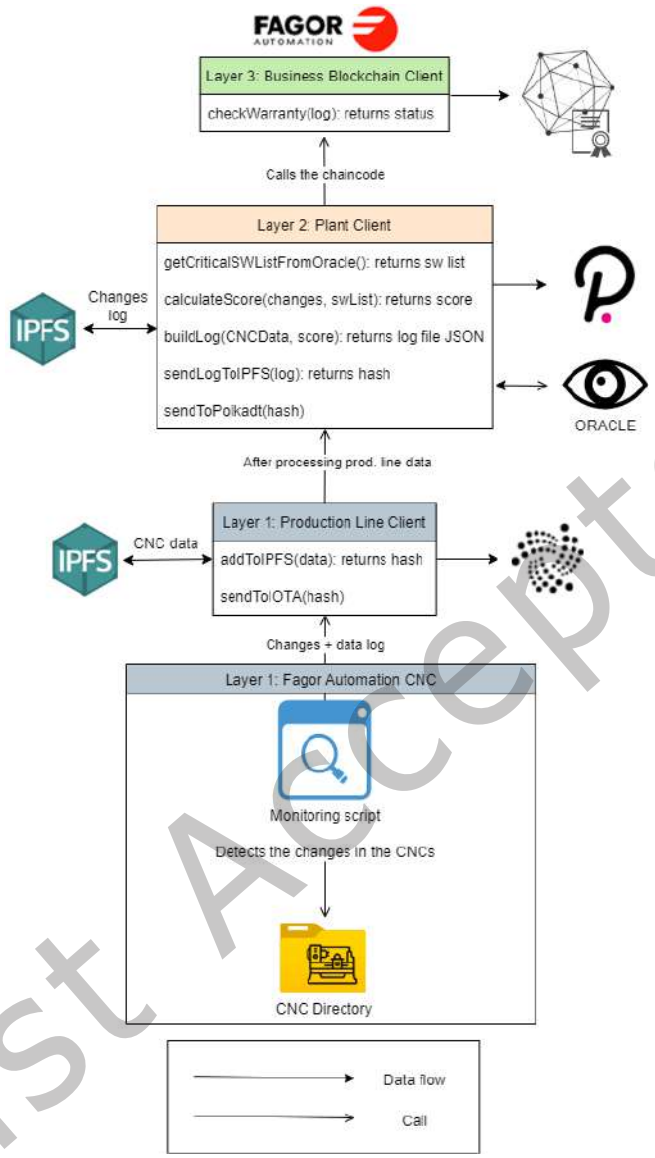

Fig. 4. Fagor Automation use case diagram

- calculateScore(changes, swlist): It computes a severity score for changes based on predefined criteria, where trivial, moderate, and severe changes are assigned scores of 1, 5, and 10, respectively.
- buildLog(CNCData, score): Generates a log file in JSON format containing the details of the changes along with the calculated scores.
- sendLogToIPFS(log): The generated log is sent to the IPFS network, receiving a unique hash that represents the stored data.

- sendToPolkadot(hash): This function sends the IPFS hash to the Polkadot blockchain for further processing and secure storage.

After sharing the information from this log with the business consortium blockchain, the chaincodes allow the manufacturer of the machines (Fagor Automation) to check the status of the warranty. If the changes detected have been assessed as illegal and critical by exceeding the established score threshold, the chaincode triggers the revocation of the warranty.

The chaincodes are designed to consider several situations that could arise during operation. For example, when multiple changes occur concurrently, the chaincode performs a sequential assessment, ensuring that each alteration is registered and scored accurately. It keeps track of the total score, and if subsequent changes cause the score to exceed the threshold, it triggers the warranty's revocation.

In cases of discrepancies in the scoring process, the chaincode employs checks to validate the consistency and accuracy of the input data. Each time a change is scored, the chaincode cross-verifies the calculated score against predefined rules based on the nature of the alteration. If the calculated score does not match the expected score for that particular change, the chaincode flags this discrepancy for manual review. This ensures that errors or inconsistencies in the scoring process do not go unnoticed and can be rectified promptly.

This entire process ensures transparency and accountability in the warranty management process, as any critical changes are logged, scored, and recorded on a blockchain that multiple parties can access and verify. The chaincode that checks the warranty status based on the generated logs from the industrial plants is shown in Algorithm 1.

Algorithm 1 Check Warranty Based on Score Chaincode

```
1:  procedure CheckWarranty(jsonPayload)
2:      warrantyDataJSON ← jsonPayload
3:      totalScore ← warrantyData.TotalScore
4:      if totalScore ≥ 10 then
5:          warrantyStatus ← ``Warranty is void"
6:      else
7:          warrantyStatus ← ``Warranty is valid"
8:      end if
9:      return warrantyStatus
10: end procedure
```

6 DISCUSSION

Despite the numerous advantages blockchain and smart contracts offer, their adoption in Industry 4.0 has been hindered by various obstacles. In the ensuing discussion section, we explore the potential of our proposed solution to revolutionize business processes within Industry 4.0, and discuss the analysis and implementation of our proposed platform in a realistic industrial case scenario. Furthermore, we emphasize the importance of data integrity and traceability in achieving a holistic DLT architecture and analyze the security and performance metrics of the implementation.

6.1 Security analysis

In this subsection, we discuss the threats to validity in terms of security within the implementation of an Industry 4.0 smart contract-based consortium blockchain with Hyperledger Fabric that interacts with external plant blockchains (Polkadot) located in each industrial plant via gateway API connections (Polkadot{.js}) and Fabric

chaincodes. The complex nature of this system presents multiple challenges that might arise. By examining each component, their interactions, and potential vulnerabilities, we aim to provide a comprehensive understanding of the threats that must be addressed to maintain the system's overall reliability.

To identify the possible threats, we begin by dissecting the components and their interactions, such as the Hyperledger Fabric blockchain, the smart contracts (chaincodes in Fabric), the Polkadot blockchains and the gateway connections for interoperability. We then assess the communication and data sharing between components to identify possible attack vectors and scrutinize external dependencies for potential vulnerabilities.

Below we provide a detailed list of possible security threats of the architecture based on the current literature on this topics [38] and how these risks are mitigated:

- Tampering: The system utilizes cryptographic hashing functions and digital signatures to ensure data integrity. Both Polkadot and Hyperledger Fabric blockchains implement secure consensus algorithms and permissioned network characteristics. Additionally, data is stored in an append-only, distributed ledger, making unauthorized data tampering nearly impossible.
- Repudiation: The use of digital signatures and an immutable, append-only distributed ledger ensures that all transactions and actions are attributable to their respective users or nodes. This design eliminates the possibility of repudiation, providing a reliable audit trail.
- Information Disclosure: Data confidentiality is preserved through encryption, both at rest and in transit. All communication between nodes, as well as between the Polkadot and Hyperledger Fabric blockchains, is secured using TLS. Access control mechanisms are also in place to ensure that only authorized users and nodes can access sensitive data.
- Denial of Service (DoS): The distributed nature of the system provides inherent resilience against DoS attacks. Both Polkadot and Hyperledger Fabric blockchains implement measures to prevent or mitigate the impact of such attacks, including rate limiting, transaction validation, and blacklisting of malicious nodes.
- Elevation of Privilege: The system incorporates a well-defined set of permissions, enforcing the principle of least privilege.
- Smart contract vulnerabilities: Smart contracts are developed using secure coding practices and undergo rigorous testing, including unit testing, integration testing, and formal verification, to minimize the risk of vulnerabilities.
- Insecure APIs: The gateway API connections between the Polkadot and Hyperledger Fabric blockchains follow industry best practices in API security, such as proper authentication, input validation, and rate limiting.
- Consensus mechanism attacks: Both Polkadot and Hyperledger Fabric have secure consensus algorithms that are designed to resist attacks such as Sybil and Eclipse. The use of a consortium blockchain, where membership is controlled, further reduces the risk of consensus-related attacks, as malicious nodes are less likely to gain a majority control of the network.
- External dependency vulnerabilities: The system's dependencies, such as the Polkadot{.js}, are carefully vetted and monitored for security vulnerabilities. Regular updates and patches are applied to minimize the risk posed by these external components.

6.2 Performance analysis

In this section, we present the performance analysis of our implementation using Hyperledger Caliper[9], a blockchain benchmarking tool. Caliper was selected due to its ability to produce a set of comprehensive performance reports and its direct compatibility with Hyperledger consortium oriented DLTs. The performance

[9]https://www.hyperledger.org/use/caliper

evaluation framework aims to measure various important metrics including transaction throughput, latency, resource usage, and chaincode execution time, which are critical in consortium blockchains [39]. Hyperledger Caliper is configured to simulate a variety of workloads representing different transaction sizes and volumes. This is done to evaluate the performance of our implementation under different operating conditions.

To provide a robust assessment, we also conduct a comparative analysis with other configuration, namely Hyperledger Fabric using the Practical Byzantine Fault Tolerance (PBFT) consensus. This comparison is essential as it allows us to explore the specific advantages and potential limitations of our Fabric implementation with Raft consensus, relative to other well-established consensus mechanisms and platforms. Understanding how different consensus algorithms affect performance under similar conditions is particularly relevant in consortium blockchains, where efficiency, security, and scalability are crucial.

In this research, we have limited our testing to the Hyperledger environment because other consortium DLTs assessed such as R3 Corda were not compatible with Caliper. This decision ensures the quality and credibility of our results by using a homogenized benchmarking platform for all tests. Furthermore, it is important to note that the objective of the proposed architecture is not to achieve the utmost efficiency, but to ensure adequate performance in the specific deployment context. Therefore, we do not seek to develop the most efficient architecture possible, as this is not the primary focus of our research and implementation.

We perform comparisons between our implemented Fagor Automation use case scenario and a basic scenario using the default chaincodes from Hyperledger, in order to examine the potential impact on performance that a more advanced scenario would have.

To provide a clearer understanding of our solution, we also perform a comparison with other state-of-the-art solutions. Specifically, Table 2 summarizes the technical comparison analysis of the other proposals. The table evaluates several aspects: whether the solution includes a performance evaluation, the replicability of the implementation based on the details provided in the respective papers, the platform used for implementation, and the type of comparative analysis performed against our solution. This detailed comparison allows us to understand how our implementation stands relative to other solutions, ensuring a comprehensive analysis of performance, replicability, and platform usage.

The tests were run multiple times under each scenario to ensure the robustness and consistency of the results. The data recorded by Caliper is then analyzed for each metric under the different scenarios. Transaction throughput and latency are analyzed to understand the capacity of our network and the responsiveness of transactions, respectively.

Table 2. Technical comparison analysis

	Performance Evaluation	**Replicable**	**Implementation Platform**	**Comparison**
[26]	No	No	Not specified	Not possible
[27]	Yes	No	Private Ethereum	Not possible
[28]	Yes	No	Private Ethereum	Theoretical
[29]	Yes	No	Private Ethereum	Theoretical
[30]	Yes	Yes	Hyperledger	Practical

6.2.1 Results - Practical comparison. In this subsection we present the results of the benchmarks. We provide a practical comparison against the solution presented in [30], wile regarding proposals [28] and [29] we perform only a theoretical discussion due to their lack of replicability.

First we evaluate the throughput of the network, as shown in Figure 5. The blue bars represent the TPS of the proposed solution from the paper when applied to the specific scenario from Fagor. The data suggest that the proposed solution scales well with an increasing number of nodes, showing only a slight decrease in TPS from 141 at 10 nodes to 128 at 30 nodes, which is already a considerable number of participants for a consortium network.

In contrast, the purple bars represent the performance of the work presented in [30], which has a sightly lower throughput capacity that the proposed solution, mainly due to a higher setup complexity designed for the supply chain management field.

For a richer comparison, the orange bars indicate the performance of the same proposed solution under a basic scenario. Interestingly, the performance under the basic scenario does not surpass the Fagor scenario and follows a similar trend of slight performance decrease as the number of nodes increase, from 142 TPS to 131 TPS.

The turquoise bars denote the TPS of the basic Fabric implementation with PBFT using the Fagor scenario. These results are consistently lower than the proposed solution for the same scenario across all node setups, which suggests that the proposed solution offers an advantage over a basic Fabric PBFT setup when tailored to the specific needs of the Fagor scenario.

Finally, the gray bars show the TPS of Fabric PBFT under a basic scenario. In this case, the TPS starts at 108 with 10 nodes and decreases to 99 with 30 nodes. The decrease in TPS as the number of nodes increases is consistent across all tested setups, which could be attributed to the increased complexity and communication overhead that comes with a larger network.

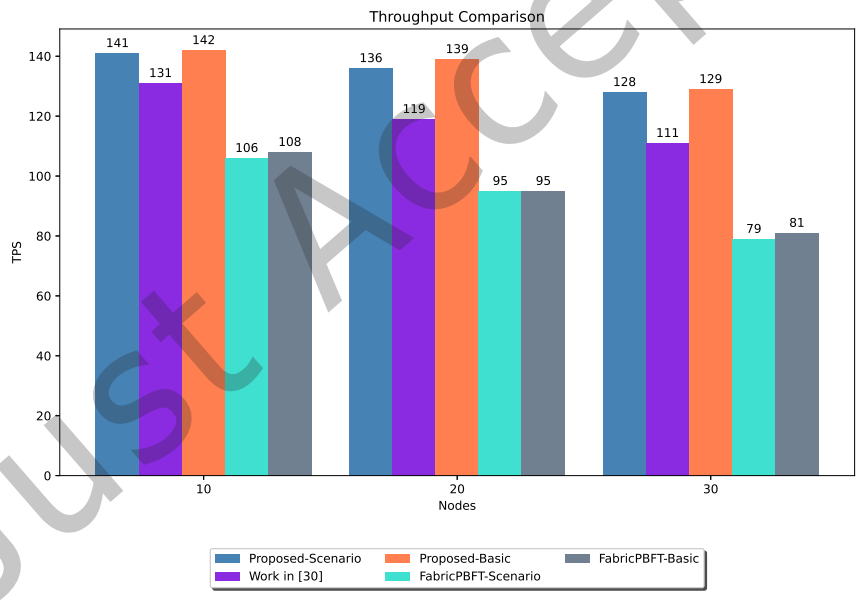

Fig. 5. Throughput (TPS) benchmarks

Apart from throughput, as shown in Figure 6, we also measured latency, which is crucial for the responsiveness of transactions. Specifically, it averaged at 0.97 seconds when testing the Fagor scenario with 10 nodes, 1.56 seconds with 20 nodes and 2.02 seconds with 30 nodes. When implementing a basic default scenario, the latency had very small variations.

In contrast, similar to the throughput results, the work presented in [30] and represented by the purple bars shows a higher latency due to the more complex setup designed for the supply chain management field.

In the case of Fabric with PBFT, the latency was significantly higher, and kept growing exponentially with the number of nodes, primarily due to the escalating complexity and overhead required for consensus. The PBFT algorithm mandates extensive inter-node communication, which increases quadratically with the number of nodes, leading to slower transaction processing. This scalability issue is a common challenge in distributed systems using Byzantine Fault Tolerant mechanisms, where ensuring reliability and security significantly impacts performance at larger scales.

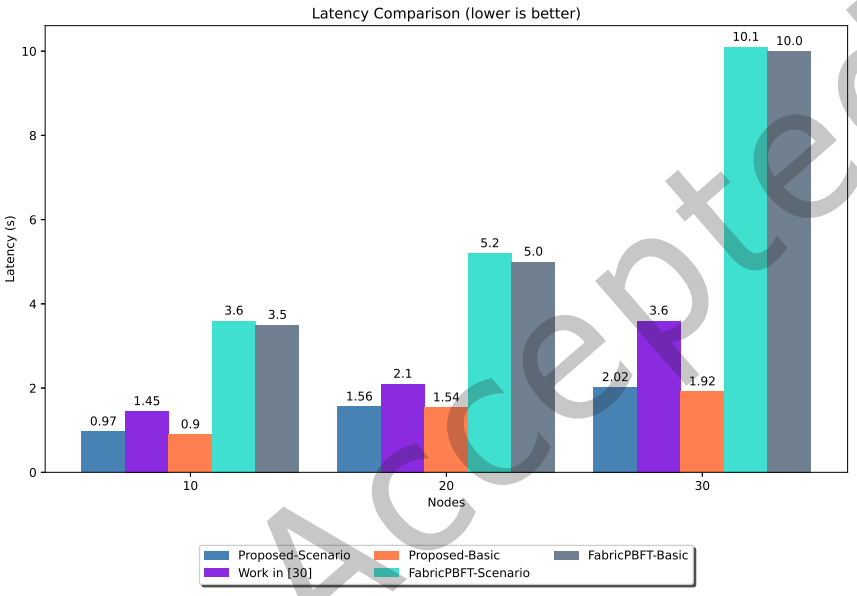

Fig. 6. Latency benchmarks

To ensure the robustness of our blockchain network for business applications, we conducted performance measurements focusing on the throughput (TPS) and network latency of the oracle network and interoperability adapters. These components are crucial for accessing external data and promoting network interoperability. The measurement process was straightforward compared to previous assessments, reflecting their simpler operational context within our architecture.

The results are presented in Table 3, which outlines the performance metrics for both the oracle network and the interoperability adapters. As indicated, both components exhibit very high throughput and minimum latency, confirming that they do not act as bottlenecks within the implemented blockchain business network.

6.2.2 Results - Theoretical comparison. In terms of performance, this proposal exhibits notable advantages compared to the frameworks discussed in the articles by [28] and [29] The comparison focuses on key metrics such as transaction throughput, latency, and scalability.

The transaction throughput of the proposed framework is significantly higher due to the use of Hyperledger Fabric and the Raft consensus algorithm. These components enable parallel transaction processing, resulting in higher throughput compared to the traditional proof-of-work mechanisms used in [28]. The framework in [29],

Table 3. Oracles and interoperability adapters performance

Module	Avg throughput (TPS)	Avg latency (s)	Max latency (s)
Oracles	804	0.5	1
Interoperability adapters	1040	0.1	0.3

while designed to be scalable, also suffers from lower throughput because it relies on less optimized blockchain architectures for transaction processing.

Latency is another critical area where the proposed framework shows improvements. The optimized consensus mechanism and the use of private channels in Hyperledger Fabric reduce communication overhead, leading to lower latency. In contrast, framework in [28] experiences high latency due to the time required for block confirmations, which affects the real-time communication needs of industrial environments. The framework in [29] also faces latency issues, though not as severe as those in [28], yet still higher than in the proposed framework.

Scalability is a significant strength of the proposed framework. The integration of private channels and oracles for external data allows the system to handle a larger number of transactions and nodes efficiently. The framework in [28], despite its decentralized nature, struggles with scalability due to the limitations of the blockchain network in managing high transaction volumes and the associated latency in block confirmations. The framework in [29] is designed with scalability in mind, but its performance is hindered by inherent blockchain transaction delays.

7 CONCLUSIONS AND FUTURE WORK

The research detailed in this paper offers a comprehensive examination into the potential advantages and difficulties in integrating blockchain and smart contract technologies in the context of Industry 4.0. These advanced technologies present compelling resolutions to prevalent issues encountered in business process management, including centralization, lack of traceability, unreliability, and insufficient automation.

The proposed solution consists of an architecture based on an interoperable consortium smart contract blockchain on top of the Hyperledger Fabric platform. This architecture is poised to address the existing challenges in process management, creating a paradigm shift in how Industry 4.0 enterprises operate. It enables the creation of private channels for secure communication and provides controlled access to external data sources.

This architecture achieves a high degree of automation, enabling processes to execute with little to no human intervention. It also promotes transparency, enabling stakeholders to monitor and audit operations in real-time. Security is fundamentally ingrained into our system, protecting sensitive data and transactions from potential threats.

The practical effectiveness of our solution has been validated through our implementation process. The presented use case demonstrated that our architecture can proficiently manage the execution of automatic agreements based on smart contract technology. It guarantees compatibility with existing systems, high-performance levels, rigorous data privacy, and affordability. All these aspects contribute to creating a system that not only meets but surpasses the requirements of Industry 4.0 enterprises.

Looking forward, this study opens up several avenues for future research and enhancement of the proposed solution. An aspect we introduced in our work is the idea of standardization in DLT design. Future efforts could concentrate on establishing a comprehensive framework for standardizing DLT designs, thereby creating a universal standard.

Another significant area of exploration is the scalability of the platform. As businesses evolve and grow, it is important for our platform to scale alongside them. This would involve the implementation of new or improved consensus algorithms or sharding techniques.

Finally, it would be beneficial to explore how machine learning algorithms could be incorporated to predict potential system changes and flag them before they occur, providing a proactive solution to maintaining the integrity and performance of CNC machines. This includes studying patterns of past system changes and their impacts, which would allow us to accurately predict future changes and take preventative action.

ACKNOWLEDGMENTS

The European Commission financially supported this work through Horizon Europe program under the HAVEN project (grant agreement number 101137636). It was also partially supported by the Department of Economic Development, Sustainability and Environment of the Basque Government under the ELKARTEK 2023 program, project BEACON (with registration number KK-2023/00085).

REFERENCES

[1] H. Lasi, P. Fettke, H. G. Kemper, T. Feld, M. Hoffmann, Industry 4.0, Business and Information Systems Engineering 6 (4) (2014) 239–242. doi:10.1007/s12599-014-0334-4.

[2] M. Kerin, D. T. Pham, A review of emerging industry 4.0 technologies in remanufacturing, Journal of cleaner production 237 (2019) 117805.

[3] P. Selvaprabhu, et al., An examination of distributed and decentralized systems for trustworthy control of supply chains, IEEE Access 11 (2023) 137025–137052.

[4] R. Stephen, A. Alex, A review on blockchain security, in: IOP Conference Series: Materials Science and Engineering, Vol. 396, IOP Publishing, 2018, p. 012030.

[5] J. Golosova, A. Romanovs, The advantages and disadvantages of the blockchain technology, in: 2018 IEEE 6th workshop on advances in information, electronic and electrical engineering (AIEEE), IEEE, 2018, pp. 1–6.

[6] D. Stefanescu, L. Montalvillo, P. Galán-García, J. Unzilla, A. Urbieta, A systematic literature review of lightweight blockchain for iot, IEEE Access (2022).

[7] I. S. Rao, M. Kiah, M. M. Hameed, Z. A. Memon, Scalability of blockchain: a comprehensive review and future research direction, Cluster Computing (2024) 1–24.

[8] H. Alshahrani, N. Islam, D. Syed, A. Sulaiman, M. S. Al Reshan, K. Rajab, A. Shaikh, J. Shuja-Uddin, A. Soomro, Sustainability in blockchain: A systematic literature review on scalability and power consumption issues, Energies 16 (3) (2023) 1510.

[9] O. Ledesma, M. Sánchez, P. Lamo-Anuarbe, Exploring dag-based architecture as an alternative to blockchain for critical iot use cases, cs & it conference proceedings 13 (24) (2023).

[10] D. Stefanescu, P. Galán-García, L. Montalvillo, J. Unzilla, A. Urbieta, Industrial data homogenization and monitoring scheme with blockchain oracles, Smart Cities 6 (1) (2023) 263–290. doi:10.3390/smartcities6010013.
URL https://www.mdpi.com/2624-6511/6/1/13

[11] T. M. Fernandez-Carames, P. Fraga-Lamas, A Review on the Application of Blockchain to the Next Generation of Cybersecure Industry 4.0 Smart Factories, IEEE Access 7 (2019) 45201–45218. doi:10.1109/ACCESS.2019.2908780.

[12] G. Culot, F. Fattori, M. Podrecca, M. Sartor, Addressing industry 4.0 cybersecurity challenges, IEEE Engineering Management Review 47 (3) (2019) 79–86.

[13] G. Prause, Smart contracts for smart supply chains, IFAC-PapersOnLine 52 (13) (2019) 2501–2506.

[14] I. A. Omar, R. Jayaraman, K. Salah, M. C. E. Simsekler, I. Yaqoob, S. Ellahham, Ensuring protocol compliance and data transparency in clinical trials using blockchain smart contracts, BMC Medical Research Methodology 20 (1) (2020) 1–17.

[15] P. Cuccuru, Beyond bitcoin: an early overview on smart contracts, International Journal of Law and Information Technology 25 (3) (2017) 179–195.

[16] W. Zou, D. Lo, P. S. Kochhar, X.-B. D. Le, X. Xia, Y. Feng, Z. Chen, B. Xu, Smart contract development: Challenges and opportunities, IEEE Transactions on Software Engineering 47 (10) (2021) 2084–2106. doi:10.1109/TSE.2019.2942301.

[17] N. Szabo, Smart contracts: building blocks for digital markets, EXTROPY: The Journal of Transhumanist Thought 18 (2) (1996).

[18] W. Zou, D. Lo, P. S. Kochhar, X.-B. D. Le, X. Xia, Y. Feng, Z. Chen, B. Xu, Smart contract development: Challenges and opportunities, IEEE Transactions on Software Engineering 47 (10) (2019) 2084–2106.

[19] C. Dannen, Solidity programming, in: Introducing Ethereum and solidity, Springer, 2017, pp. 69–88.

[20] L. Breidenbach, C. Cachin, B. Chan, A. Coventry, S. Ellis, A. Juels, F. Koushanfar, A. Miller, B. Magauran, D. Moroz, S. Nazarov, A. Topliceanu, F. Tramèr, F. Zhang, Chainlink 2.0: Next Steps in the Evolution of Decentralized Oracle Networks (2021).
URL https://research.chain.link/whitepaper-v2.pdf
[21] O. Dib, K.-L. Brousmiche, A. Durand, E. Thea, E. B. Hamida, Consortium blockchains: Overview, applications and challenges, International Journal On Advances in Telecommunications 11 (1&2) (2018) 51–64.
[22] M. Castro, B. Liskov, Practical byzantine fault tolerance, OSDI 99 (1999) 173–186.
[23] A. Lohachab, S. Garg, B. Kang, M. B. Amin, J. Lee, S. Chen, X. Xu, Towards Interconnected Blockchains: A Comprehensive Review of the Role of Interoperability among Disparate Blockchains, ACM Comput. Surv. 54 (7) (2021). doi:10.1145/3460287.
URL https://doi.org/10.1145/3460287
[24] G. Wood, POLKADOT: VISION FOR A HETEROGENEOUS MULTI-CHAIN FRAMEWORK, Tech. rep., Eindhoven University of Technology (TU/e) (2016).
URL https://polkadot.network/PolkaDotPaper.pdf
[25] R. Belchior, A. Vasconcelos, S. Guerreiro, M. Correia, A Survey on Blockchain Interoperability: Past, Present, and Future Trends, ACM Comput. Surv. 54 (8) (10 2021). doi:10.1145/3471140.
URL https://doi.org/10.1145/3471140
[26] X. Ye, M. König, Framework for automated billing in the construction industry using bim and smart contracts, in: E. Toledo Santos, S. Scheer (Eds.), Proceedings of the 18th International Conference on Computing in Civil and Building Engineering, Springer International Publishing, Cham, 2021, pp. 824–838.
[27] K. Demertzis, L. Iliadis, N. Tziritas, P. Kikiras, Anomaly detection via blockchained deep learning smart contracts in industry 4.0, Neural Computing and Applications 32 (23) (2020) 17361–17378.
[28] C. T. B. Garrocho, C. M. S. Ferreira, A. S. S. Júnior, C. F. M. da Cunha Cavalcanti, R. A. R. Oliveira, Industry 4.0: Smart contract-based industrial internet of things process management, in: Anais Estendidos do IX Simposio Brasileiro de Engenharia de Sistemas Computacionais, SBC, 2019, pp. 137–142.
[29] A. Dixit, W. Asif, M. Rajarajan, Smart-contract enabled decentralized identity management framework for industry 4.0, in: IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society, IEEE, 2020, pp. 2221–2227.
[30] D. Ravi, S. Ramachandran, R. Vignesh, V. R. Falmari, M. Brindha, Privacy preserving transparent supply chain management through hyperledger fabric, Blockchain: Research and Applications 3 (2) (2022) 100072. doi:https://doi.org/10.1016/j.bcra.2022.100072.
URL https://www.sciencedirect.com/science/article/pii/S2096720922000136
[31] M. H. Tabatabaei, R. Vitenberg, N. R. Veeraragavan, Understanding blockchain: definitions, architecture, design, and system comparison, arXiv preprint arXiv:2207.02264 (2022).
[32] M. Conti, G. Kumar, P. Nerurkar, R. Saha, L. Vigneri, A survey on security challenges and solutions in the iota, Journal of Network and Computer Applications (2022) 103383.
[33] Y. Liu, K. Wang, Y. Lin, W. Xu, Lightchain: A lightweight blockchain system for industrial internet of things, IEEE Transactions on Industrial Informatics 15 (6) (2019) 3571–3581. doi:10.1109/TII.2019.2904049.
[34] D. Guegan, Public blockchain versus private blockchain, HAL SHS (2017).
[35] O. Choudhury, I. Sylla, N. Fairoza, A. Das, A blockchain framework for ensuring data quality in multi-organizational clinical trials, in: 2019 IEEE International Conference on Healthcare Informatics (ICHI), 2019, pp. 1–9. doi:10.1109/ICHI.2019.8904634.
[36] W. She, Z.-H. Gu, X.-K. Lyu, Q. Liu, Z. Tian, W. Liu, Homomorphic consortium blockchain for smart home system sensitive data privacy preserving, IEEE Access 7 (2019) 62058–62070. doi:10.1109/ACCESS.2019.2916345.
[37] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, et al., Hyperledger fabric: a distributed operating system for permissioned blockchains, in: Proceedings of the thirteenth EuroSys conference, 2018, pp. 1–15.
[38] S. Soni, B. Bhushan, A comprehensive survey on blockchain: Working, security analysis, privacy threats and potential applications, in: 2019 2nd international conference on intelligent computing, instrumentation and control technologies (ICICICT), Vol. 1, IEEE, 2019, pp. 922–926.
[39] W. Choi, J. W.-K. Hong, Performance evaluation of ethereum private and testnet networks using hyperledger caliper, in: 2021 22nd Asia-Pacific Network Operations and Management Symposium (APNOMS), IEEE, 2021, pp. 325–329.

Received 22 January 2024; revised 9 July 2024; accepted 17 July 2024