future internet

MDPI

Article

Exploiting Blockchain Technology for Enhancing Digital Twins' Security and Transparency

Alessio Ferone *,† and Stefano Verrilli *,†

Department of Science and Technology, University of Naples Parthenope, 80143 Naples, Italy
* Correspondence: alessio.ferone@uniparthenope.it (A.F.); stefano.verrilli001@studenti.uniparthenope.it (S.V.)
† These authors contributed equally to this work.

Abstract: Blockchain technology has been applied in a wide range of domains and has seen major developments in the last years, notably when integrated with other emerging technologies. In this paper, we focus on the integration of blockchain and digital twins with the aim of enhancing the capabilities of both technologies. In particular, we demonstrate how blockchain can improve critical aspects of the security and transparency of digital twins by analyzing a real-world scenario and evaluating produced experimental data. This research identifies and addresses critical vulnerabilities in digital twins, particularly data integrity and transparency, through blockchain-based validation mechanisms and smart-contract integration. Various blockchain-related and digital twin-related technologies are employed to enable the repeatability of the suggested approach. Additionally, an in-depth analysis of such integration is provided to facilitate a symbiotic relationship between these technologies by addressing key challenges, such as scalability, interoperability, and performance, along with viable solutions that could advance their co-evolution in both academic research and industrial applications.

Keywords: blockchain; digital twin; surveillance; Ethereum; smart contract

Academic Editor: Qiang Qu

Received: 28 November 2024
Revised: 7 January 2025
Accepted: 8 January 2025
Published: 13 January 2025

Citation: Ferone, A.; Verrilli, S. Exploiting Blockchain Technology for Enhancing Digital Twins' Security and Transparency. *Future Internet* **2025**, *17*, 31. https://doi.org/10.3390/fi17010031

1. Introduction

In the current industrial landscape, digital twin technologies have emerged as a prominent application to modern day challenges, along with the Internet of Things (IoT), by enabling faster data generation and collection. A digital twin offers a robust framework for representing physical objects and systems, allowing for the simulation and analysis of their real-time behavior and performance. By leveraging digital twin technologies, industries can optimize operations, predict potential issues, and make data-driven decisions that enhance efficiency and reduce operational costs.

Meanwhile, IoT technologies have become a common solution in industrial applications, enabling faster, autonomous communication between interconnected devices. This network of devices collects and exchanges data over the internet, fostering automation and real-time decision-making across various sectors. The integration of IoT with digital twin technologies has further revolutionized industrial operations by enhancing the flow of data and facilitating more dynamic and informed control over processes.

The widespread use of digital representations in the industrial processes has resulted in a constant rise in the consumption and exchange of industrial data. In the broader landscape of IoT technologies, the cooperation between interconnected devices and the extended reflection of those interactions to the physical counterpart has further increased the impact on industrial operations and represented one of digital twins' main advantages

over classical smart devices [1]. Moreover, this seamless exchange of information between digital twins enhances control over system-generated data and enables the continuous monitoring of potential malfunctions within the created ecosystem. Ref. [2] emphasizes that the bidirectional communication between physical and digital realms ensures robust performance monitoring and rapid fault detection.

However, as emphasized in numerous studies addressing the security challenges of IoT technologies, digital twin systems are frequently targeted by potential attackers due to their often inadequate security measures and widespread deployment in environments with limited control. In particular, ref. [3] highlights vulnerabilities in the authentication protocols of IoT devices, which can be exploited in digital twin systems. Ref. [4] discusses common attack vectors, such as data spoofing and unauthorized access, that can compromise the integrity of digital twins.

As an extension of IoT smart devices, digital twin technologies can face similar risks, including the possibility of data tampering and theft, which could potentially lead to data breaches or leakage in the worst-case scenario. Studies such as [5,6] have examined real-world instances where such security lapses resulted in significant operational disruptions, underlining the need for enhanced protective measures.

A robust security ecosystem for digital twins must be established to facilitate a transparent information exchange between digital representations while minimizing the performance loss for the underlying physical devices. Ref. [7] proposes a multi-layered security framework that combines encryption, authentication, and anomaly detection techniques to address these challenges through third-party systems. Furthermore, the security solution should be resistant to direct attacks aimed at compromising the security system's infrastructure. Ref. [8] introduces EtherTwin, a blockchain-based system designed to ensure tamper-proof and verifiable interactions in digital twin ecosystems.

Many possible solutions have been proposed over the years, addressing the problem mainly by introducing internal controls over potentially unsafe information exchanges. Ref. [9] highlights the role of blockchain-based distributed ledgers in ensuring data immutability, while [3] underscores the importance of decentralized access control mechanisms. These solutions primarily focus on the integrity and confidentiality of data but often neglect the underlying security system's vulnerability to a direct attack.

This research proposes an innovative approach that leverages the adaptability potential of blockchain technology, a decentralized system that has experienced rapid adoption in recent years. Refs. [10,11] discuss the widespread adoption of blockchain across industries, highlighting its versatility in securing transactional and operational data. Beyond its initial application in cryptocurrencies, blockchain has found widespread use in various domains. Ref. [12] examines its integration into supply chains, showing how blockchain improves traceability and accountability. The decentralized nature of blockchain enables the secure and immutable storage of diverse data types, including documents, facts, packets, and transactions, as [13,14] demonstrate in their respective studies.

However, despite these advances, a significant gap remains in ensuring the transparent and secure integration of digital twin management systems with robust security frameworks. This research aims to address this challenge by pioneering an innovative approach that bridges digital twin ecosystems with blockchain technology. The intent is to create a seamless integration that not only enhances data security and traceability but also ensures transparency in operations without compromising system performance. By enabling real-time, secure interactions between digital twins and blockchain systems in a symbiotic relationship, this approach advances the state of the art, transforming how industrial ecosystems manage and secure their data. The following paragraphs explore a potential

application of this technology within the context of digital twin security, aiming to minimize communication between the physical system and its blockchain-based counterpart.

In Section 4, we evaluate the proposed system's performance, focusing on scalability and its contribution to enhanced security while ensuring the correct functioning of the solution proposed through multiple tests. Section 5 demonstrates its real-world application using a real system and the newly integrated security system.

2. Related Works

In recent years, substantial academic attention has been directed towards the intersection of blockchain technologies and digital twin systems, driven by the increasing need for enhanced data integrity, transparency, and security in industrial applications. The paper [15] provides a comprehensive survey on blockchain applications in industries, highlighting how blockchain's decentralized and immutable nature can address critical challenges in transparency, traceability, and security. Their work emphasizes the transformative potential of blockchain for ensuring accountability and trust in data sharing, particularly in resource-constrained environments.

Further, Ref. [16] explores the integration of digital twins and blockchain in the context of construction projects, emphasizing how blockchain enhances accountability in information sharing, thus enabling more trustworthy decision-making and reducing risks of data manipulation. Their framework offers a novel approach to mitigating conflicts of interest in large-scale projects by leveraging the blockchain's distributed ledger capabilities to ensure traceability and access control.

Another critical contribution is the work by [6], who conducted a systematic review on the integration of blockchain and digital twins in smart-built environments. They highlighted how blockchain enhanced the trustworthiness of digital twin systems by providing mechanisms for secure and verifiable data exchange, particularly useful in environments characterized by decentralized operations and distributed data.

Furthermore, Ref. [4] discusses digital twin security threats and countermeasures, emphasizing the necessity for robust security frameworks in digital twins. Their findings underscore the importance of blockchain's role in mitigating security risks, such as unauthorized access and data tampering, which are especially critical in industrial applications like manufacturing processes and smart cities.

Overall, these studies collectively underscore that the integration of blockchain and digital twins not only addresses technological challenges but also significantly enhances data security, accountability, and efficiency in industrial applications. They demonstrate that leveraging blockchain for digital twins helps to foster more trustworthy and secure systems in environments that demand high degrees of data transparency and integrity.

2.1. Consensus Algorithm

The consensus mechanisms employed in these blockchain types differ significantly. For public blockchains, consensus algorithms such as Proof-of-Work (PoW) and Proof-of-Stake (PoS) are typically used, where the emphasis is placed on computational power and economic incentives to maintain network integrity. On the other hand, private blockchains often rely on the Proof-of-Authority (PoA) consensus algorithm [17]. PoA operates by relying on a system of trusted validators, where a committee of pre-approved nodes is responsible for verifying transactions [18]. This approach assumes a higher level of trust among participants, which enables faster transaction validation and scalability [10].

The use of PoA is particularly advantageous in private blockchain settings, as the presence of a designated group of validators not only accelerates transaction processing but also addresses one of the fundamental challenges associated with blockchain technology—the

need for rapid data exchange in environments with increasing volumes of information. Hence, the application validity of one blockchain type over the other is the main aspect to consider while choosing a suitable infrastructure solution and thus determine the superiority of one consensus mechanism over the other.

2.2. Smart Contracts

Another prominent issue concerning blockchain technologies regards the lack of a programmable way of interacting with it. This lack of classical distributed ledgers coupled with the impossibility of modifying written transactions could lead to a loss of data integrity and, in the worst-case scenario, data loss.

The academic discourse in this area predominantly highlights the transformative potential of smart contracts in automating blockchain operations, thereby reducing or entirely eliminating the need for third-party intermediaries in transactions. The deployment of smart contracts within blockchain systems not only streamlines these processes but also significantly enhances transparency and facilitates the debugging of complex workflows, particularly in industrial environments [19,20].

Furthermore, the introduction of the auxiliary use of smart contract could also address the problem related to the blockchain unsuitability for data storage by introducing accessible data structures designed for this task. This prominent characteristic of smart contracts of storing a small quantity of data under the data structure's element could eliminate the need for off-chain storage solutions and integrate them into the digital twin blockchain system [21].

On the other hand, this kind of reasoning is only possible by considering the small quantity of data usually employed for controlling capabilities. In fact, since smart contracts and their data are stored on the blockchain itself, the transfer of related information is deemed to be at a slower speed compared to an off-chain solution. Also, in this case, the adopted data storage solution is tied to a trade-off problem, the speed of the off-chain solution or the trustworthiness of its on-chain counterpart.

In the context of integrating smart-contract technology with digital twin systems, several studies have suggested leveraging ERC721 and ERC20 standards to maximize the benefits of these systems [22]. By developing scripts on blockchain platforms that support smart contracts, it is possible to standardize procedures and improve their auditability and security characteristics. This standardization is particularly relevant for enhancing the interoperability of smart contracts and ensuring secure data exchange across various nodes in the network.

The ERC721 standard enables, in the digital twin context, the unique identification and traceability of individual twin instances. For instance, in industrial applications, each digital twin can be represented by an NFT, encapsulating key parameters of its physical counterpart. This approach guarantees tamper-proof tracking of system changes, enhancing accountability and reducing risks of data manipulation in shared environments. These tokens, being distinct and non-interchangeable, provide a reliable means to track and manage assets, thus preserving the integrity of the information associated with them.

Despite the described advantages, the implementation of a blockchain-based representation for digital twins also introduces certain challenges. One key concern is the potential for increased complexity and abstraction in an already complex system. The additional layer of blockchain could potentially complicate the digital twin infrastructure, making it more challenging to manage and operate.

Moreover, while the immutability of smart contracts is often regarded as a benefit for ensuring data integrity, it also introduces significant limitations. Once deployed, smart contracts cannot be altered, which reduces their flexibility and hinders the ability to make

iterative improvements. In response to these challenges, standards such as EIP-1967 [23] and EIP-2535 [24] have emerged as key advancements in smart-contract engineering [25]. For example, EIP-1967 formalizes a method for proxy storage slot allocation, enabling proxy contracts to delegate function calls to an implementation contract while securely referencing its address within a fixed storage slot.

Additionally, EIP-1822 [26] introduced the UUPS (Upgradable and Upgradable Proxy Standard), which aims to enhance the upgradability of smart contracts while preserving their immutability and state integrity. This framework allows developers to implement upgradable smart contracts without losing essential state or data. Traditional smart contracts are immutable once deployed on the blockchain, meaning any necessary bug fixes or feature enhancements require the deployment of a new contract address. The UUPS addresses this limitation by introducing a proxy mechanism that decouples contract logic from data storage. The proxy contract retains state variables, while the implementation contract contains the business logic. This separation ensures that contract upgrades can be applied without altering stored data, maintaining the system's integrity and ensuring continuity.

This approach minimizes the risk of storage collision and facilitates the upgrade process. Proxy patterns utilizing EIP-1967, such as the Transparent Proxy Pattern and the Universal Upgradable Proxy Standard (UUPS), have become foundational tools in modern contract development [27]. By enabling pseudo-upgradability through carefully designed versioning protocols, developers can achieve a balance between the immutability required for security and the adaptability demanded by dynamic application scenarios [28].

2.3. Digital Twins

Digital twins represent one of the most prominent technologies to empower the IoT infrastructure enabling data-driven approaches for industry management. Their implementation, however, faces significant challenges as pointed out by [29,30]: communication and data security constitute a major downside and can be extremely challenging to integrate into already established systems. Moreover, these authors highlight that the lack of a clear data standardization and diversity in source systems represent another significant inconvenience concerning the data exchange protocols used.

In the context of data security, particularly regarding communication with digital twins, Refs. [31,32] broadly explored blockchain integration as one of the possible solutions to address these challenges. The cooperative interaction between these two distinct technologies resembles an instance of technology fusion, a phenomenon which, as analyzed by [33], has frequently recurred throughout technological history.

Although digital twins provide industrial systems with a precious source of analytical information, Ref. [34] emphasizes that they lack the accountability and responsibility for shared information, which could in turn negatively impact the integrity of data. Additionally, these authors point out that data standardization remains a critical issue within the digital twin ecosystem. The diversity of systems and platforms used across industries often leads to the adoption of proprietary software solutions that handle data in ways that are not universally compatible. This fragmentation impedes the seamless integration of digital twins with other systems, hindering the effective exchange of information and limiting the scalability and interoperability of digital twin applications.

3. Proposed Approach

To address the challenges related to the digital twin's security system and avoiding potential information leakage, we proceeded by defining the key objectives to satisfy. The first is undoubtedly tied to the implementation of an access control system; this measure safeguards the information exchanged with the digital twins and ensures that data are

handled solely by authorized entities, thereby preventing their misuse for unauthorized purposes. The first proposed measure to ensure the security of exchanged data is based on the integration of blockchain accounts assigned to all authorized entities in the system. Each authorized entity requires a corresponding blockchain address granting as proof of his authenticity.

The validation system of communication sources relies on the introduction of a private backbone blockchain. This blockchain records the entities interacting with the digital twin, enhancing the overall security of the communication process. Due to the requirements of our system, no monetary transaction is performed over this backbone structure, and any transaction fee is totally excluded from the equation. Regarding the integrity of data within the security system, a validation component is integrated to ensure that received data correspond to the data sent by the originating entity. This measure is essential to avoid discrepancies and to guarantee the reliability and authenticity of the exchanged data.

To achieve this objective, the system incorporates a data validation mechanism within the smart-contract subsystem, complemented by a middleware component integrated into the communication infrastructure. By leveraging the autonomy of the introduced stub component within the digital twin subsystem, off-chain transaction signing is employed, thereby maintaining the locality of the operation and avoiding unnecessary data transfers. The integrated middleware component of the system manages the logical operations serving as a foundation for the administrative control and management of the digital environment and its full description is represented in Figure 1.

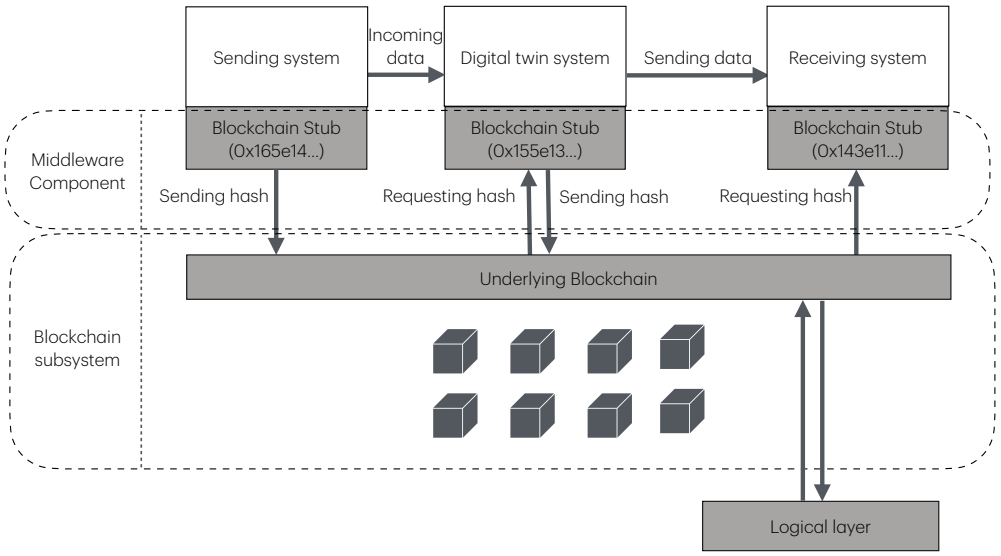


Figure 1. Diagram of the proposed architecture for the communication between the system's components and the underlying blockchain.

To further enhance the security of data transmitted to the blockchain component, a hashing mechanism is employed. The hash of the data is computed and transmitted alongside the signed transaction. In a hypothetical communication, the digital twin only accepts pre-validated information on the blockchain through the middleware component. Additionally, data transmission from the digital twin occurs only after undergoing an additional layer of validation. This approach ensures that the logical layer retains only the "fingerprint" of the shared information, enriching the overall data locality while minimizing the stored volume of data on the blockchain.

In order to guarantee a deterministic and programmatic approach to the digital twin's information and the underlying blockchain, a smart-contract-based subsystem is adopted to

manage the logic governing the system's components. Furthermore, to allow an incremental upgradability of the proposed solution, the EIP-1967 design pattern is adopted over the system logic through the implementation of an UUPS proxy.

The long-term usability of the system is strictly related to those characteristics, as they are essential for ensuring the solution's adaptability to evolving security requirements. Those factors play a crucial role in the system's future development and ensure a sustained operational efficiency.

In addition to the specific security measures previously outlined, it is essential that the proposed system incorporate robust administrative functionalities. These capabilities enable continuous monitoring and dynamic modification of communication flows within the digital environment. This characteristic is crucial for detecting and diagnosing potential security vulnerabilities by allowing proper interventions in order to address them.

Furthermore, to enhance the overall security framework, it is necessary to establish an internal security measure within the system to prevent unauthorized communication with digital twins. This measure aims to isolate internal issues within individual sections of the system and prevent them from spreading throughout the digital environment, which could otherwise compromise the operational integrity of other connected components.

To implement these desired capabilities, the system architecture employs the Principle of Least Privilege (PoLP) approach. The latter is achieved through a clear structural separation within the smart-contract design, explicitly distinguishing administrative functionalities handled by the *Manager sub-system* from lower-level operations handled by the *Storing sub-system*. This modular architecture results in the development of two distinct smart contracts that, while operating independently, act as a cohesive entity with clearly defined roles and responsibilities.

An additional advantage provided by the combined implementation of the upgradable proxy pattern and role-based division within the smart-contract architecture is the ability to continuously develop and update managing tasks without losing stored information in the storage contract. This interesting feature of the proposed system enables the seamless integration of new business logic into the existing framework without disrupting the natural flow of information in the system. The path followed by data regarding the interaction with this component is illustrated in Figure 2.

The final aspect of the proposed solution to address regards the performance of the validation system employed, since the integration of blockchain technology into the system architecture may introduce delays. Specifically, the performance bottlenecks could originate from the concentration of requests towards a particular node potentially resulting in the congestion of the network and delays in the processing of the requests.

To address the challenge related to the uneven distribution of requests, the system involves the deployment of a load-balancing mechanism to evenly distribute the number of incoming requests across multiple nodes, as shown in Figure 3. The load balancer acts as a gateway for the middleware component and provide the appropriate address to communicate with the blockchain. Regarding the consensus protocol employed by the network nodes, which plays a critical role in determining system performance, a detailed analysis is presented in the results section (see Section 4).

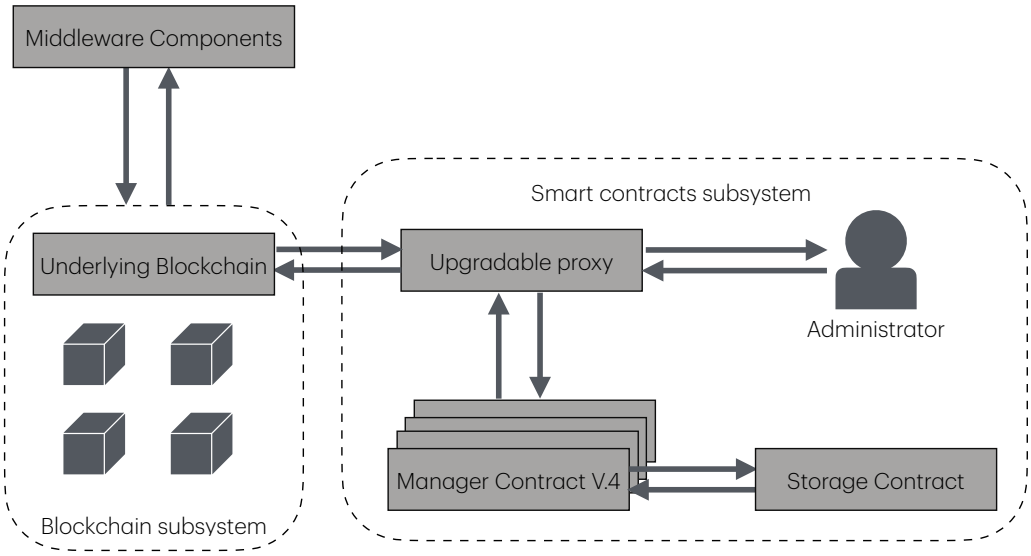


Figure 2. Architecture used for the implementation of the smart contracts' subsystem. The system's hashes are stored into a low-privilege contract while managing functionalities and access control capabilities are accessed through a manager contract.

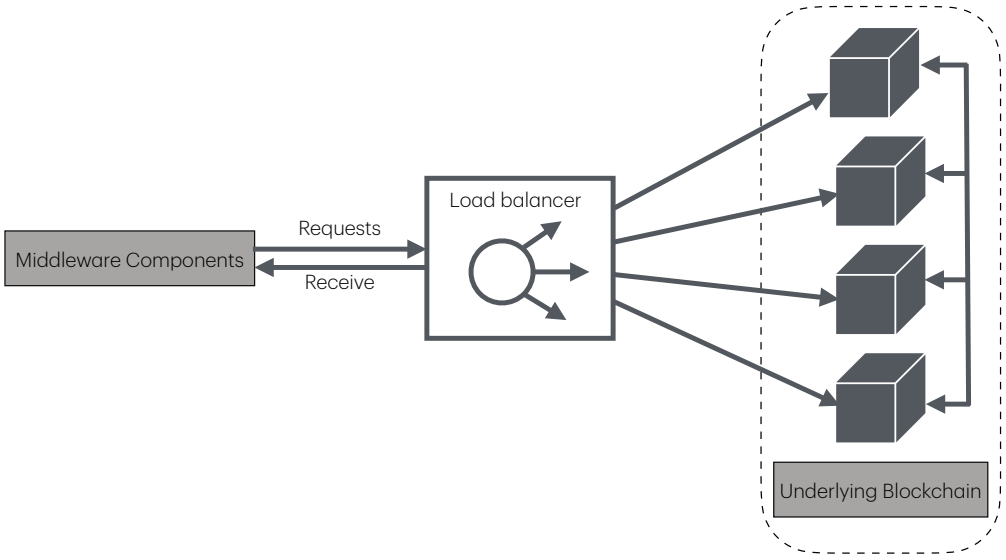


Figure 3. The solution proposed to allow a balanced flow of requests from the system's actors to the nodes of the blockchain.

3.1. Implementation

In the next paragraphs, the methodology and practical implementation strategies undertaken to fulfill the required objectives are analyzed.

3.1.1. Blockchain Creation

The architectural requirements outlined in this research necessitate a blockchain protocol that supports robust smart-contract functionality while allowing for future system upgrades. To meet these criteria, we adopted the Ethereum protocol due to its well-established presence in the blockchain industry and its comprehensive support for smart-contract development.

The blockchain infrastructure for this study was implemented using Hyperledger Besu [35], a highly regarded Ethereum client known for its enterprise-grade capabilities. Besu offers exceptional flexibility in configuring private networks, making it particularly

suitable for applications requiring secure, permissioned environments. Additionally, the rich ecosystem provided by the Hyperledger suite ensures seamless integration with analytical tools and complements the experimental framework.

Hyperledger Besu was chosen for its compatibility with enterprise-level blockchain requirements, particularly in the context of digital twin systems. Its robust support for smart contracts and permissioned configurations aligns perfectly with the security, scalability, and reliability demands of such systems. Moreover, Besu's interoperability with the broader Ethereum ecosystem ensures long-term adaptability, enabling smooth integration with existing blockchain tools and frameworks.

Beyond addressing performance and security objectives, Hyperledger Besu enhanced the reproducibility of the proposed blockchain architecture through a docker-based solution. This capability facilitated straightforward deployment across diverse business-oriented blockchain applications, ensuring the scalability, versatility, and practicality of the framework in real-world scenarios.

It is important to note that while Hyperledger Besu was used in this implementation for its ease of use and rich feature set, the proposed system is not dependent on this specific software. Any blockchain platform that supports robust smart-contract functionality and permissioned configurations can be utilized, as the general design of the approach is platform-agnostic. In order to communicate with the blockchain, each actor involved in the system requires a pre-authenticated address, already available at the moment of network creation or added afterward. Once the communication is established without rejection from the network, the encapsulated stub proceeds to validate its communications with the digital twin by sending a signed message through the blockchain. Each validated transaction is then stored in the isolated blockchain and recorded in the shared ledger.

3.1.2. Virtualization

In alignment with the requirements outlined in the preceding paragraph, the primary objective of our proposed architecture was to establish a secure and isolated security system while enabling the reproducibility of the proposed solution. To fulfill these requirements for the designed backbone blockchain, we decided to introduce an additional layer of abstraction by containerizing each component within the blockchain. This decision was executed through the employment of the Docker engine [36], facilitating the creation of blockchain nodes and the integrated load balancer. The additional system layer ensured a controlled and secure interaction between components while maintaining them in an isolated virtual environment.

Using the Docker-provided images for Hyperledger Besu nodes, our approach involved configuring each node to align with the validation requirements of the network. The abstracted configuration was then encapsulated in Dockerfile format, creating a modular and flexible solution suitable for various network tasks. Following the virtual environment configuration of the security subsystem, an entry point for network tasks was created through port binding on the load balancer's host machine. The configured solution was then deployed as a component in the security solution ready to serve any request coming from the involved actors of the system.

3.1.3. Smart Contracts

After establishing the foundational blockchain protocol, the next phase involved defining the implementation of smart contracts to be utilized within the validation system. For this development, Solidity was selected as the programming language, given its wide recognition as a core language for smart contract programming on blockchain platforms [37]. Solidity's robust features make it particularly well suited for constructing secure and reliable

validation mechanics while ensuring alignment with industry standards. Moreover, the integration of standardized contracts such as EIP-1967 further enhanced the development experience of the proposed solution.

The development of the manager contract followed this path by integrating into its deployment an upgradable proxy, which served as a stepping stone to integrate additional features in previous iterations of the solution. In terms of its own logic, the manager contract served as an on-chain security measure by enabling component communications and providing administrative access to designed network addresses. To establish precise directionality between entities within the system, each approved communication was recorded in the manager contract as a sender–receiver tuple within a stored list. This structured representation of interactions ensured that all communications within the producer–consumer framework complied with the predefined security protocols.

Furthermore, the administrative contract was uniquely designated as the sole authorized entity for direct communication with the on-chain storage counterpart. This storage component was configured to accept transactions exclusively from the manager contract which initiated the connection and subsequently retrieved the required hash data. By implementing a decoupled smart-contract architecture, the system ensured a robust data preservation and recovery policy, minimizing the risk of data loss while providing an interface-based communication between parts. This modular approach also allowed for seamless updates to the manager contract without compromising the security infrastructure, thus maintaining the essential security support needed. Pseudocode describing the functionalities provided by the manager contract to communicate with the storage contract is provided below in Algorithm 1:

Algorithm 1 Manager communication logic with the connected storage contract

```
1:  procedure MappingExists(_key)
2:      return corresponding[_key].exists
3:  procedure getAdminList
4:      Return AdminContractInstance.getAdmins()
5:
6:  procedure StoreInterface(Hash)
7:      if MappingExists(msg.sender) is false then
8:          revert
9:      Call StoreVal(Hash, msg.sender) on StorageContractProxyInstance
10: procedure RetrieveInterface(_receiver)
11:     if correspondings[msg.sender].to ≠ _receiver then
12:         revert
13:     Call RetrieveVal(msg.sender) on StorageContractProxyInstance
```

Moreover, for reproducibility reasons, the code employed to perform the administrative functionalities for the system's topology management is provided through the same framework as the previous pseudocode exemplification Algorithm 2.

The corresponding pseudocode representing the implementation logic of a "Storage" smart contract is provided, as a direct counterpart to the previously defined manager contract, in Algorithm 3. The contract serves as a secure data repository, with functionality for storing and retrieving hashed values. It also incorporates access control mechanisms to ensure that only authorized entities, such as a manager proxy, can interact with the contract's critical functions. The pseudocode highlights the logical flow of these operations and the modifiers that enforce security.

Algorithm 2 Manager contract administrative functionalities

```
1:  procedure NEWCORRESPONDING(_sender, _receiver)
2:      if MappingExists(_sender) is true then
3:          revert
4:      Set correspondings[_sender].to ← _receiver
5:      Set correspondings[_sender].exists ← true
6:  procedure DELETECORRESPONDING(_sender)
7:      if MappingExists(_sender) is false then
8:          revert
9:      Set correspondings[_sender].exists ← false
10: procedure UPDATEENTITY(_sender, _receiver)
11:     if MappingExists(_sender) is false then
12:         revert
13:     Update correspondings[_sender].to ← _receiver
```

Algorithm 3 Storage Contract Operations

```
1:  procedure INITIALIZE
2:      Set owner ← msg.sender
3:  procedure ISCALLER
4:      require(ManagerProxyAddress == msg.sender)
5:      Continue
6:  procedure ISOWNER
7:      require(owner == msg.sender)
8:      Continue
9:  procedure AUTHORIZEUPGRADE(newImplementation) override
10:     if IsOwner condition is not met then
11:         revert
12: procedure STOREVAL(Hash, Storer)
13:     if IsCaller condition is not met then
14:         revert
15:     if ManagerProxyAddress ≠ msg.sender then
16:         revert
17:     Set CheckSum[Storer] ← Hash
18: procedure RETRIEVEVAL(fromPK)
19:     if IsCaller condition is not met then
20:         revert
21:     if  then ManagerProxyAddress ≠ msg.sender
22:         revert
23:     Return CheckSum[fromPK]
24: procedure SETCALLER(CallerAddress)
25:     if IsOwner condition is not met then
26:         revert
27:     Set ManagerProxyAddress ← CallerAddress
```

The complete architecture regarding the developed smart contracts is provided in the form of a class diagram in Figure 4. This representation highlights the symbiotic relationship established between the contracts and the integration, in terms of pattern utilization, of the UUPS.

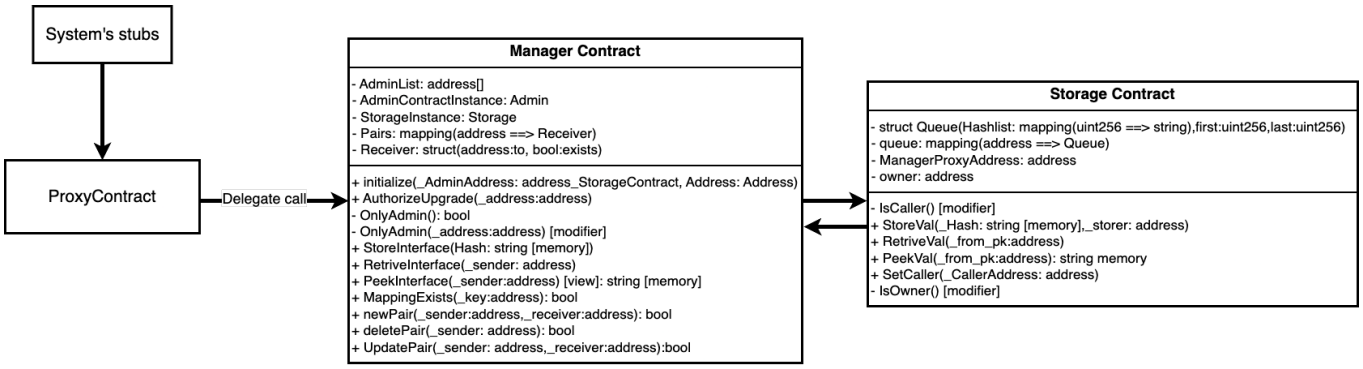


Figure 4. Relationship, in terms of contract's functions, established in the component, clarifying the role division enforced in the system.

3.1.4. Stub Components

To achieve the validation objectives designated for the middleware component within the system, multiple implementations of the underlying logic were developed to communicate with the digital twin and the data senders and receivers. Specifically, in this study, stubs were created in different programming languages, including Java 11 and C++ 11 with Python 3-bound code and are available in the paper repository. Although the task of stub development might appear dependent on the specific characteristics of each system's actor, consistent logic was applied across all implementations in order to guarantee coherence of the system as a whole.

Thus, the implementation preserved core logic integrity across diverse programming environments, aligning the middleware's behavior consistently with the system's broader validation requirements. In particular, each stub component of the system presented a corresponding implementation of the hash storing and retrieving while implementing a system's consistent hashing procedure in order to guarantee the autonomy of the solution within the containing environment.

To ensure consistent functionality across all system stubs, each component was designed to communicate autonomously with the blockchain subsystem, enabling seamless data exchange with the digital twin. This interaction was facilitated by the hashing validation system, which served as an intermediary component to authenticate communication within the architecture. The hashing validation system not only verified the legitimacy of exchanges but also ensured robust security by systematically rejecting any unauthorized communication. The final diagram examining a communication interaction between a sending system and the digital twin is represented in Figure 5, which describes the data flow during a canonical communication.

In order to allow interoperability by blockchain operators, we decided to implement each of the component through the utilization of the Web3 functionalities. In fact, over its vast use in different blockchain applications, the Web3 library also presents multiple implementations in different programming languages and a large following in the blockchain community. Due to its flexibility, the Web3 library represents a suitable solution to provide a shared implementation procedure while adopting a common logic between stubs.

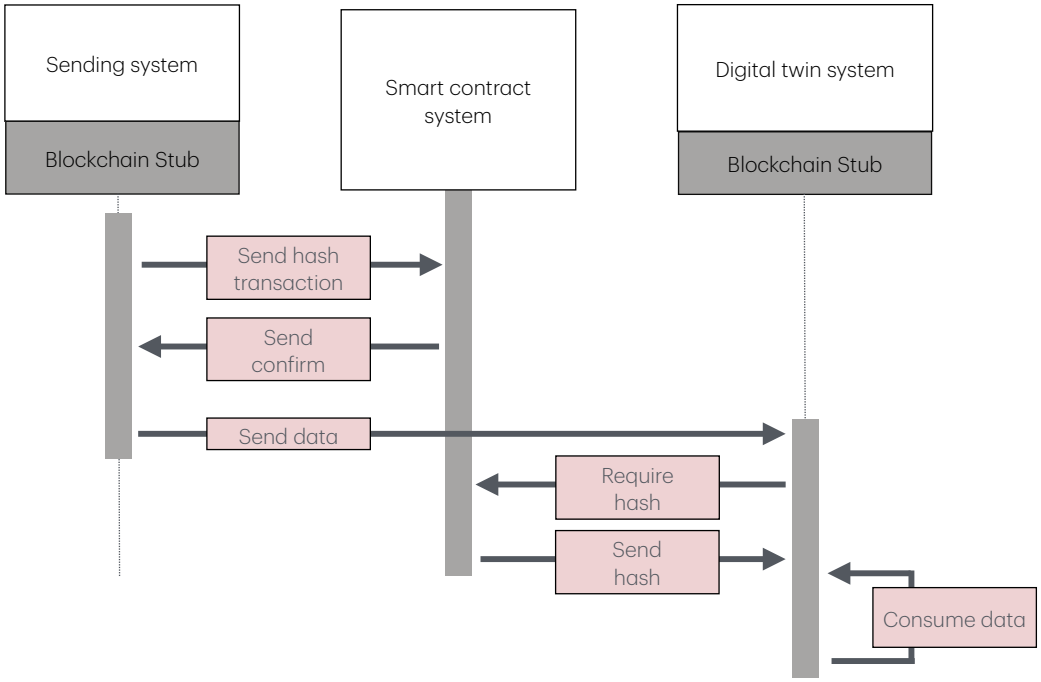


Figure 5. Sequence diagram of the communication steps in a data transmission.

4. Experimental Results

In this section, we present the experimental evaluation of the implemented system, using the practical application described in Section 5 as a testbed. The experimental results were derived from transaction logs collected from the blockchain network's nodes and Docker container consoles, providing a comprehensive view of the system's operational dynamics. The evaluation focused on two primary aspects: the security and administrative functionalities of the system's components and the performance of the underlying blockchain framework. Initially, we conducted an in-depth analysis of the data flow during communication with the digital twin system, examining both *sending* and *receiving* operations to ensure reliability and consistency. Additionally, security tests simulated scenarios involving unauthorized data submissions and directionality mismatches to validate the system's resilience and the effectiveness of its smart-contract mechanisms.

Moreover, to assess the framework's performance and robustness, we employed Hyperledger Caliper, a blockchain benchmarking tool, to conduct a series of experiments. These experiments measured critical metrics such as transaction throughput, latency, and network scalability under varying conditions.

4.1. Security Analysis

This section provides an analysis of the tests conducted to evaluate the security integrity of the proposed system. The examination focused on the security robustness of smart contracts and supporting stub components, analyzing their functionalities within the framework of established security requirements. Multiple tests were executed, and detailed results were systematically recorded through logging outputs to validate the operational logic and security measures implemented.

The smart contracts serve as the main mechanism for enforcing security policies in the system. Unauthorized transactions involving actors different from those designed for the communications can potentially lead to the malfunctioning of the system. Specifically, the manager contract is crucial in securing the system, acting as a gateway for managing data hashes and facilitating secure communication protocols.

To assess the access control mechanisms, two separate account addresses were instantiated for testing, one with administrative privileges and the other without. Both accounts attempted to access restricted functions within the manager contract to update the list of authorized communications. The transaction logs from these attempts provided insights into the system's response to authorized and unauthorized access requests.

Results showed that transactions initiated by the authorized administrative account were processed successfully across all tested operations, as anticipated, while attempts initiated by the unauthorized account were appropriately denied, consistent with the system's communication protocols and the access level assigned to the secondary address. This outcome validates that access control mechanisms effectively restrict administrative functionalities, aligning with the security objectives of the system.

Following the verification of the secure access to administrative functionalities within the contracts, the next phase of analysis focused on evaluating the communication dynamics between data-sending actors and the digital twin system. This specific test was designed to examine the system's resilience against unauthorized communication attempts that could potentially alter the data representation within the digital environment.

Similarly to the access control testing performed on the smart contract's administrative functions, this test assessed the system's response by initiating transactions from two distinct addresses: one authorized as a valid data sender and another unregistered. The anticipated outcome was an accepted transaction for the authorized address and a rejection for the unauthorized one.

For this phase of testing, access restrictions provided by the backbone blockchain were temporarily disabled to enable an isolated assessment of the management component's behavior in handling secure communications. Without this adjustment, the network validators would automatically reject any transaction originating from a sender lacking a registered address, preventing an authorized update of the digital twin's information. This approach allowed for a focused analysis of the system's internal security mechanisms independent of external blockchain-based access controls.

Also, in this case, the results of that testing phase were favorable, aligning with the expected behavior of the manager contract's security protocols. When data transmissions were initiated from both addresses, the authorized address successfully completed a transaction recorded by a blockchain node, indicating that the manager contract's internal security systems approved the communication. On the other hand, for the second address, a revert message was recorded by the blockchain's nodes assessing the invalid nature of the communication route followed.

In order to complete the tests performed over the manager's account security requirements, a last test was performed to assess the capability of the latter to validate communication routes following a correct directionality. In this last scenario, both addresses were previously authenticated in the system with a communication path initialized from the first account and received by the second one. In that test, we performed a reversed transmission initialized by the second account and received from the first one.

According to the security logic, although a valid communication existed between the two involved addresses, the directionality of that transmission should result in a rejection from the manager contract. Although the results showed that the transmission was correctly signed and sent, the outcome result was a rejection by the gateway contract indicating an unauthorized communication between parties.

This series of tests confirmed our expectations regarding the manager contract's behavior, thereby confirming the security requirements outlined in Section 3.1 for the overall smart-contract architecture. In the next phase of analysis, we extended the validation of these security requirements to the system's stub components, examining potential

vulnerabilities in the communication framework from an alternative perspective within the proposed solution.

4.2. Stub Testing

In order to guarantee the interoperability between the hosting system component and the blockchain's stub integrated into each of the actors involved, we now analyze the steps followed by the security measures introduced. Moreover, in the last part of this discussion, we focus on the security aspects regarding the consumption of invalidated data from the digital twin's system.

Following a canonical path of communication between the system's actors, after a channel initialization, a sender started to communicate with the digital twin while sending data hashes to the storing contract through the manager gateway. The path followed by the signed transactions was recorded by the validating nodes, which traced the steps followed by the system in the update of the digital twin's information.

Within the provided information, each functional aspect of the associated sub-module could be identified, enriched with specific information regarding the processed data. The hash values corresponding to the generated JSON files were recorded as input parameters, followed by the receipts associated with the submissions of hashes to the blockchain.

While data were received from the direct communication channel connecting the sender module and the digital twin, hashes were required from the blockchain's smart contract to be compared with the one obtained locally. The collected logs confirmed that the data transmission by the auxiliary sender sub-module were successfully received by the digital twin and were ready to be subsequently processed by the Hosting system.

As the last step of the message validation life cycle, the received information was correctly validated by the sub-system, and the digital twin software proceeded to update the information of the model, updating the underlying model. This fundamental test allowed us to assert the correct behavior of the system under correct conditions, though it left open important questions regarding the security protocols put in place.

In order to evaluate the security requirements of the communication process, we carried out a behavioral analysis using as an example a transaction unknown to the validation system forwarded to the digital twin system only by standard communication means. The expected results described by the transaction life cycle should imply a discarded transaction from the system. The following test simulated the concurrent transmission of both unconfirmed data and confirmed data to the digital twin system.

The purpose of that test was mainly to evaluate the system's robustness in preventing interference by a potential malicious actor attempting to exploit the primary transmission channel to compromise the digital twin's management mechanism. It was expected that the receiver system would maintain its intended functionality by disregarding any unconfirmed messages, thereby continuing to receive an additional hash value and validate subsequent incoming data.

In the specified test scenario, we configured two kinds of sender actors: a fraudulent source only communicating with the digital twin through direct messages, and a canonical source acting as intended for the system. Both actors involved proceeded to send syntactically correct messages to the digital twin's system, and the response of the latter was registered through specific logging messages in the stub.

Through this analysis, we were able to assert the formulated hypothesis, obtaining evidence of the predicted behavior. Any fraudulent transmission attempt resulted in a mismatch between hash values received from the sources, leading the receiver module to reject the received data for the digital twin update. Following this rejection, canonical system operation resumed, allowing the validation cycle to continue as intended without

causing service interruptions. The provided tests were essential to assess the security requirements for the components of the system mostly subjected to possible attacks and provided an opportunity to closely examine the actual functionality of the final system in different scenarios.

Moreover, it is possible to further analyze the recorded results with respect to the CIA Triad model to better highlight the design choices from a security strategy point of view. The aforementioned model comprised three essential aspects:

- Confidentiality ensures that only authorized users have access to information;
- Integrity ensures information non-repudiation and authenticity;
- Availability ensures reliable access to information.

In this regard, the choice of the blockchain as the underlying distributed ledger for secure information exchange aimed to follow the model prescriptions. In fact, the massive use of encryption, both for saving data and service access, naturally enforces confidentiality and integrity. In addition, by its distributed architecture, the decentralized nature of blockchain ensures a high degree of availability.

For what concerns the systems components, considering the analysis summarized in Table 1, it is clear that the proposed approach guarantees confidentiality, allowing only authorized transactions to be propagated and hence recorded on the blockchain as a hash value (thus enforcing integrity). The availability of the system is strongly connected to the availability of the blockchain, because the smart contracts are also stored on the blockchain and hence their availability is also guaranteed.

Table 1. Summary of security analysis for different components, showcasing potential threats and system responses.

Tested Component	Possible Threat	Recorded Behavior	Tested Scenario
Manager contract	• Illegal operations • Data leaks • Digital twin attack	Transaction reverted from the manager contract	Updating admin list from an unauthorized address
	• Illegal communications • Digital twin invalidation • System malfunction	Transaction reverted as not registered as a valid path	Sending hash data from an unauthorized address
	• Information misuse • Data loss • Coherence invalidation	Authorized actors, but communication reverted	Invalid communication direction from/to valid addresses
Stub components	• Unauthorized update • Data invalidation • System jamming	Discarded communications without system interruptions	Data sent directly to the digital twin system bypassing hash validation

4.3. Performance Benchmarks

Although the provided tests for security requirements were essential to ensure the system's reliability, performance measurements were equally crucial in asserting the system's overall usability. The tests' main objective in this paragraph concerned the evaluation of multiple performance metrics regarding the blockchain backbone while providing a reproducible environment for the results shown.

The provided tests leveraged the benchmarking capabilities of Hyperledger Caliper in order to provide accurate and extensive results on the deployed blockchain network. To achieve the following results, we proceeded by deploying in the digital twin system a self-hosted private blockchain network through docker configuration files as defined in Section 3.1. With the objective of registering performance metrics concerning transactions traveling through the network, we introduced the additional docker container of the Caliper image directly connected to the main network.

The whole network was then deployed onto a suitable physical environment consisting of an IMac machine equipped with 10 cores of computational power and 32 GB of RAM and updated to the MacOS Monterey operating system. Full access to all the computational resources described was then granted to the docker network, and it was connected with the external environment through port binding on the host machine. Moreover, the communication network between containers was managed through a local connection on the specified machine to eliminate dependencies on the variable speed of the external network.

4.3.1. Performance Metrics

Evaluating the performance of a blockchain network is critical and requires a thorough analysis, as it cannot rely on a single factor. Assessing network quality extends beyond simply measuring transaction validation speed; a comprehensive examination of all available metrics is essential. As discussed in Section 2.1, a critical aspect of network transaction speed and overall quality concerns the choice of a suitable consensus algorithm for the designed problem. This section provides a quantitative analysis of several suitable protocols to facilitate a data-driven approach in selecting the optimal solution. More generally, network quality is assessed by identifying specific characteristics of the network and then selecting those that best align with the requirements of the intended application. A short summary of the included characteristics:

- Finality: capability of blockchain protocols to reach consensus between nodes without creating additional branches in the network.
- Vitality: assessed through the maximum number of failing validation nodes before reaching the collapse of the network's validation capabilities.
- Throughput: speed at which transactions are evaluated and accepted by the network and inserted in blockchain blocks.
- BFT (Byzantine fault tolerance): capability of the network to reach consensus in case of partial validation node failing.
- Scalability refers to the network's ability to sustain its validation rate as its configuration change. Scalability is further divided into horizontal and vertical scalability. Horizontal scalability measures the validation rate as the number of network validator changes, while vertical scalability assesses the validation rate in response to variations in available hardware resources.

While the characteristics outlined above provide valuable insights into the metrics relevant for our testing phase, it is crucial to distinguish consensus protocols based on their specific use cases. In this section, we analyzed several algorithms suitable for our objectives, focusing in particular on the Clique, IBFT 2.0, and QBFT protocols. A short description of the main characteristics of each of those protocols is provided:

- Clique: This protocol is designed for environments similar to Ethereum's mainnet. It allows for faster block addition and can operate with just one validator, making it suitable for testing and demonstration purposes. However, it does not guarantee immediate finality, which means forks can occur if the network experiences issues or if there are insufficient validators available [38].
- IBFT 2.0 (Istanbul Byzantine Fault Tolerance): An evolution of the original IBFT protocol, IBFT 2.0 requires at least four validators to ensure Byzantine fault tolerance. It guarantees immediate finality, meaning once a block is added, it cannot be altered or removed from the chain. This makes it suitable for production environments where reliability is crucial [39].
- QBFT (Quorum Byzantine Fault Tolerance): Developed as an enterprise-grade version of IBFT, QBFT also requires four validators and offers immediate finality. It incorporates a more efficient leader election process and reduces communication rounds

needed for consensus compared to IBFT, making it better suited for larger networks with higher scalability needs [38].

The unique characteristics, along with the primary similarities and differences of each protocol, are summarized in Table 2. This table serves as a key reference in order to validate our selection of the most appropriate consensus mechanism for our peculiar use case.

Table 2. Comparison of consensus protocols: Clique, IBFT 2.0, and QBFT.

Feature	Clique	IBFT 2.0	QBFT
Consensus type	Proof of Authority (PoA)	Byzantine fault tolerance (BFT)	Byzantine fault tolerance (BFT)
Fault tolerance	Up to 50% of nodes	Up to 33% of nodes	Up to 33% of nodes
Performance	High transaction throughput	Moderate transaction throughput	Moderate to high transaction throughput
Block finality	Probabilistic	Immediate	Immediate
Use cases	Private networks	High-security applications	High-security applications
Strengths	Simple, fast, effective in trusted environments	Strong finality, fault tolerance	Strong finality, improved performance in BFT environments
Limitations	Lacks finality, vulnerable if over 50% nodes fail	Lower throughput compared to PoA	Similar limitations as IBFT 2.0

4.3.2. Benchmarks on Throughput

The initial test conducted on our network configuration focused on evaluating the network throughput by incrementally increasing the Transactions Per Second (TPS) sent to the network for validation. In this analysis, we examined both the throughput and latency performance across different consensus protocols to identify the most optimal protocol in terms of transaction approval rate.

The transaction load began at a baseline of 100 TPS and progressively increased until it reached 700 TPS. The transactions involved were associated with the "send hash" functionality, which is designed to enable multiple concurrent connections to the digital twin system. This setup served as a practical scenario for assessing the network's ability to handle varying levels of transaction demands while maintaining system efficiency.

As shown in Figure 6a, the results of the tests indicated that the throughput values were generally similar across the three analyzed algorithms, with Clique exhibiting greater instability in the number of transactions approved. Regarding the latency in Figure 6b, the Clique algorithm slightly outperformed the others, showing a momentary equality when reaching 600 transactions per second. In comparison, both QBFT and IBFT 2.0 demonstrated nearly identical performance, with comparable latency times, reinforcing their similarity in terms of overall performance.

Although the Clique protocol showed some important advantages in terms of validation speed, its inability to provide a consistent and stable environment for the digital twin's update requests made this protocol not well suited for our purposes in terms of reliability. In the following tests, we still provided measurements utilizing that consensus protocol as a comparison with the IBFT 2.0 and the QBFT counterparts.

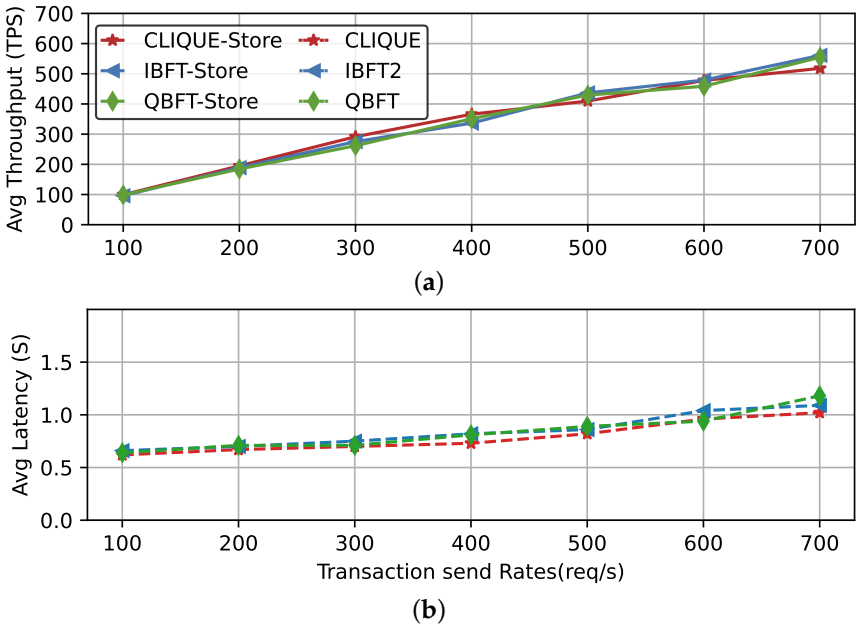


Figure 6. The plot in (**a**) shows measurements of the average throughput due to a progressive increase in transaction sending rate. Instead, the plot in (**b**) highlights the registered latency of the three networks in the same scenario. (**a**) Throughput plot with respect to the consensus protocols. (**b**) Latency plot with respect to the consensus protocols.

The next test aimed to evaluate the scalability characteristics of the consensus protocols under consideration, with particular emphasis on their horizontal scalability. To investigate that behavior, we assessed the network's performance by gradually increasing the number of validators, starting from a baseline of 4 and scaling up to a maximum of 14. During that process, transactions were sent at a rate of 400 transactions per second, for a total of 2000 transactions. The obtained results regarding the transaction throughput and the effective latency are outlined in Figure 7a,b, respectively.

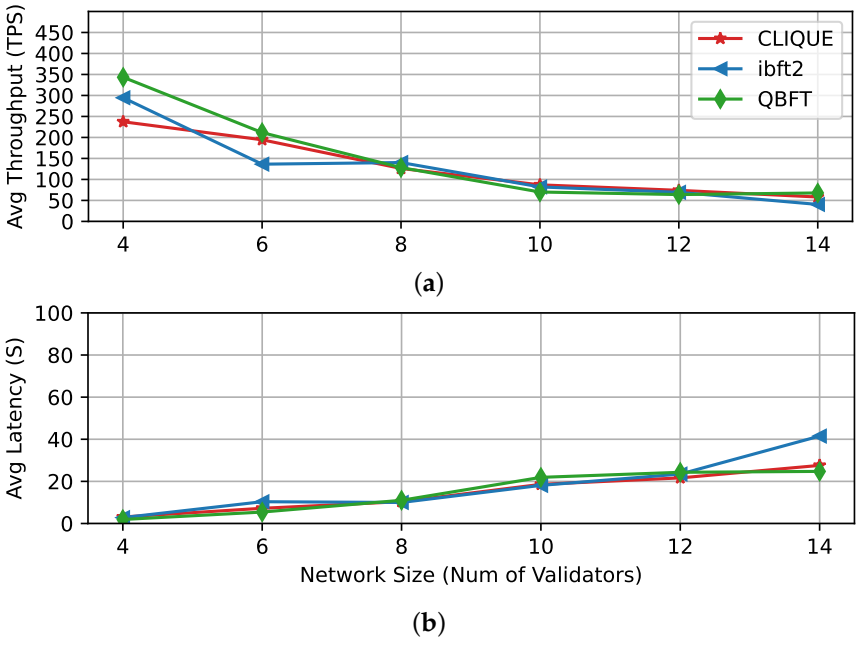


Figure 7. Performance of the different consensus protocols in increasing network's size configurations. (**a**) Throughput plot with respect to the horizontal scalability. (**b**) Latency plot with respect to the horizontal scalability.

In this scenario, the Clique algorithm also reaffirmed its position as the protocol with the lowest latency, although it demonstrated a lower transaction approval rate compared to the others, due to the probabilistic nature of its consensus mechanism. Instead, the IBFT 2.0 and the QBFT algorithms exhibited similar behaviors in terms of throughput, as indicated by the corresponding measurement plots. However, a steady increase in latency was observed for IBFT 2.0 as the network size grew, reaching its peak at the maximum size of the network. In fact, the latency recorded by the IBFT 2.0 algorithm consistently remained lower than that of the QBFT, establishing a lower bound for its latency in comparison.

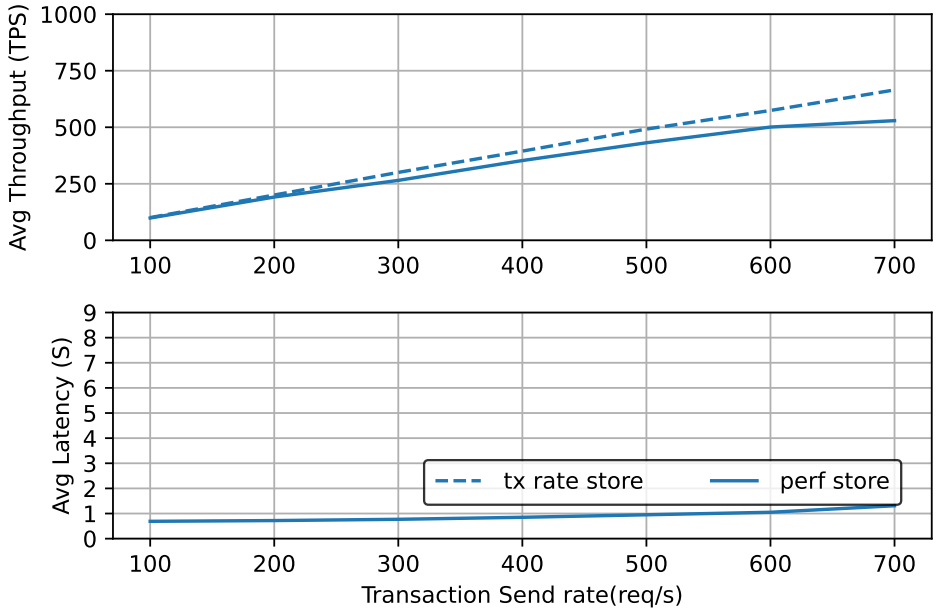
The narrow latency gap between these two consensus protocols necessitates further analysis in the subsequent sections to determine the most suitable protocol for the specific security requirements of our system's environment. To clarify the obtained benchmark results and designate a suitable consensus algorithm for our system's purposes, we conducted an additional test on these two protocols based on varying transaction send rate configurations. The test highlighted the measurements obtained in a simulated utilization of the real system described in Section 5 through its canonical use.

Moreover, we tested the capabilities of the network to serve in both sending and receiving hash modalities. The objective of these tests was to evaluate the system's performance under high transaction loads. Specifically, assuming a transaction rate of 60 transactions per second from the different instances of the sender devices, the analysis determined whether the system could support up to N occurrences in the real environment.

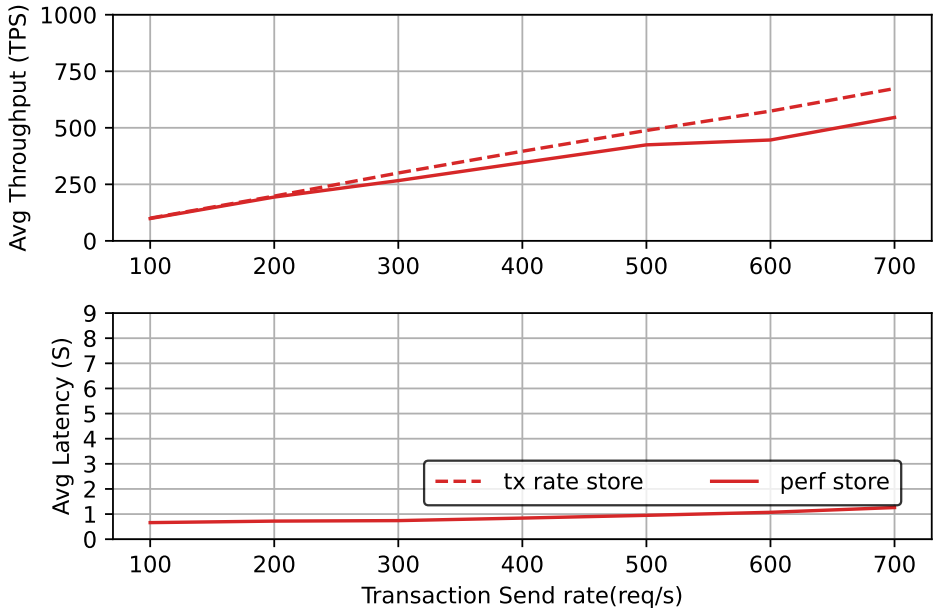
Analyzing the performance of the QBFT and the IBFT 2.0 algorithms, as presented in Figure 8, it was observed that both algorithms exhibited similar latency and throughput characteristics. However, QBFT demonstrated greater resilience to increasing transactions per second compared to IBFT 2.0. These benchmarks illustrated the number of transactions successfully validated (represented by solid lines) in relation to the number of transactions sent per second (represented by dashed lines). Through these tests, the architecture of our blockchain network could be effectively defined.

Summarizing the results, the series of tests conducted in this paragraph on the underlying blockchain framework and the associated consensus protocols led to the selection of the QBFT algorithm as the underlying algorithm for our solution. Although the Clique protocol initially appeared promising for the offered transaction speed, it demonstrated instability early in the scaling and stress tests. In contrast, both QBFT and IBFT 2.0 maintained comparable performance across the benchmark suite. However, under conditions of increased network load, subtle but significant performance distinctions emerged between the two protocols, ultimately favoring QBFT.

Additionally, the network architecture proposed was configured to operate with six nodes. This specific number of validators was chosen based on observed optimal performance outcomes of the QBFT protocol, guaranteeing enhanced fault tolerance and network resilience. The provided benchmarking results demonstrated the suitability of the QBFT consensus protocol compared with the proposed requirements for the real-time digital twin updates, maintaining a constant transaction throughput even during peak loads while minimizing the overall latency. These metrics directly correlated with the system's objective to provide a secure yet responsive digital twin framework, ensuring its scalability for high-demand industrial scenarios like supply-chain monitoring and other IoT-enabled solutions.



(**a**)



(**b**)

Figure 8. Average throughput and average latency of the two consensus protocols compared to the effective number of transactions sent to the system. The storing functionality was tested for both QBFT (**a**) and IBFT2 (**b**) consensus protocols. (**a**) QBFT protocol plots. (**b**) IBFT protocol plots.

It should be noted that other DLT models with different degrees of security and performance could be evaluated for implementing the proposed approach. With regard to DAG (Direct Acyclic Graph)-based models, although performance in terms of throughput and scalability is on average better than blockchain models [40], they have a lower degree of decentralization which results in a lower degree of availability. In addition, the consensus mechanism is generally entrusted to fewer nodes, which reduces data integrity. Iota and Nano (which, however, does not support smart-contract execution) fall into this category. As for blockchain-based models, one possible alternative is Hyperledger Fabric, which performs on average better than Ethereum (of which Besu is a client) [41]. However, Besu

has two strengths that made us prefer it: better performance in terms of transaction rate and the possibility of integration with the public Ethereum blockchain.

5. Working Example

This section presents a practical application of the proposed framework, illustrated through the case study of the AiWatch system [42]. The primary objective of this modular project is to develop a machine learning-powered system for the protection of physical environments (e.g., rooms, hallways) in real-time scenarios. By integrating the proposed architecture into a practical implementation, this study aimed to validate the obtained results while verifying the feasibility of the proposed design and highlighting its potential application in real-world systems.

In the context of this application, each physical space is represented by its corresponding digital twin, while cameras serve as data-gathering agents for the virtual model by capturing and transmitting information on detected entities. The produced data are subsequently forwarded to an elaborating device, which is responsible for processing and analyzing the received information to enable effective system operation. As a last step of this procedure, the elaborated data are then retransmitted from different cameras into a final destination server employed for the update of the environmental digital twin represented.

To facilitate efficient and reliable communication among devices within the system, the integration of Apache Kafka serves as the primary communication channel. By managing the system's data queues through Kafka's topic-based infrastructure, the architecture enables the security system to concentrate on validation tasks at the component level. The system adopts the JSON format as the communication standard, and with each data transmission within Kafka topics, the message is formatted, ensuring consistent data exchange standards across the system's components.

The AiWatch system leverages real-time surveillance data to update digital twins, serving as a proper test case for validating the proposed blockchain-based security framework. By capturing and processing continuous streams of data from surveillance cameras, AiWatch updates the digital twin models with actionable insights in real time. This dynamic and complex data flow highlights several critical challenges, including ensuring the integrity and authenticity of transmitted data, maintaining system reliability under high transaction loads, and providing scalability to accommodate increasing data volumes or system expansions. The application of the blockchain-based proposed solution addresses these challenges by offering a decentralized and tamper-proof layer of data validation while providing a local autonomy for the actors involved resulting in enhancing both the security and transparency of the system.

Within the AiWatch application, each component involved is assigned a specific identifier that is used consistently throughout this discussion. Specifically, the term *Tracker* refers to the camera modules responsible for the initial processing of image frames, while the final destination server, tasked with managing digital twin information, is designated as *Ditto*. The latter is chosen as a direct reference to the *Eclipse Ditto* 1.1.0 software, which empowers the module's functionality.

The focus of this discussion lies in integrating the proposed security system into the pre-existing AiWatch architecture. To achieve this objective, the development of specialized sub-modules for both Tracker and Ditto was required, while the specified smart contracts and the underlying blockchain infrastructure were provided to support the system's functionality. A visual schematic description of the system is provided in Figure 9, where the proposed integration is included.

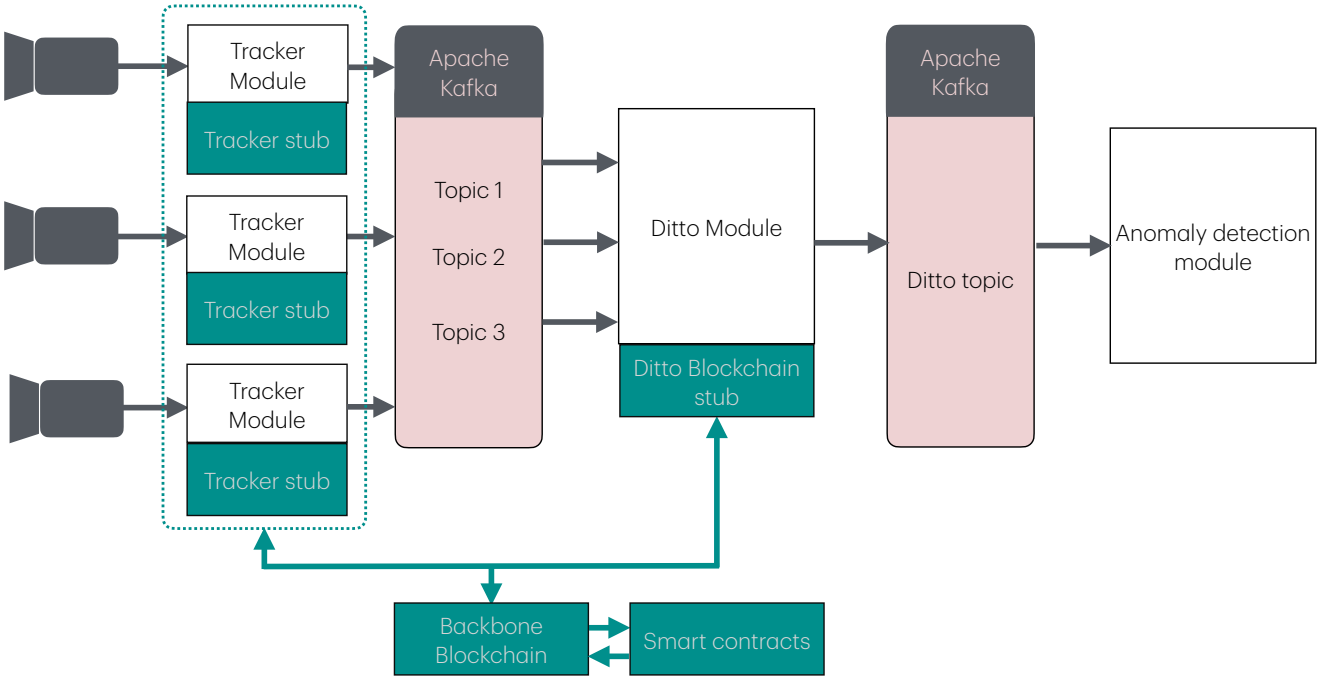


Figure 9. Diagram of the AiWatch system architecture, emphasizing the integration points for blockchain-based security features. Black arrows indicate data flow, while the blockchain serves as a tamper-proof layer for communication validation.

For the purposes of this analysis, further exploration of the general functionalities provided by the AiWatch system was postponed in order to focus on the implementation of the security system for managing the digital twin's information. The objective was to achieve a secure and reusable interface for interacting with the digital twin, thereby ensuring consistent and reliable use of the collected data. In particular, the interaction with the digital twin's information with respect to the *Anomaly Detection* module was ignored for it minimal or no risks in those communications. With this foundational definition of the proposed integration points in place, the focus then shifted to the development of the individual sub-modules.

Stubs' Development

The legacy code written in C++ and provided by the developers of *Tracker* presents a complex structure originated from the necessity of extracting key information from the captured frames corresponding to the junctions of the recorded entities in the scene. A critical function of that module is the transformation of extracted data into structured JSON files, followed by their transmission to a designated Apache Kafka topic via an appropriate wrapper. The proposed hash validation architecture integrates with this process at the precise moment prior to data transmission through the Kafka topic. At this stage, the connection to the blockchain is initialized, and the data are stored within the smart contracts. However, due to compatibility issues with the C++ implementation of the Web3 library—used as a standard framework for the stub modules—a Python- wrapped solution was introduced. This solution, executed within the C++ code, was incorporated to enable communication with the system's unified blockchain infrastructure and allow access to the validation system's functionalities.

As illustrated in Figure 10 the introduction of the blockchain communicating sub-module through the Python-integrated portion of the code (on the right) allowed the

Tracker component to seamlessly communicate with the validating network while keeping its autonomy in its standard functionalities.

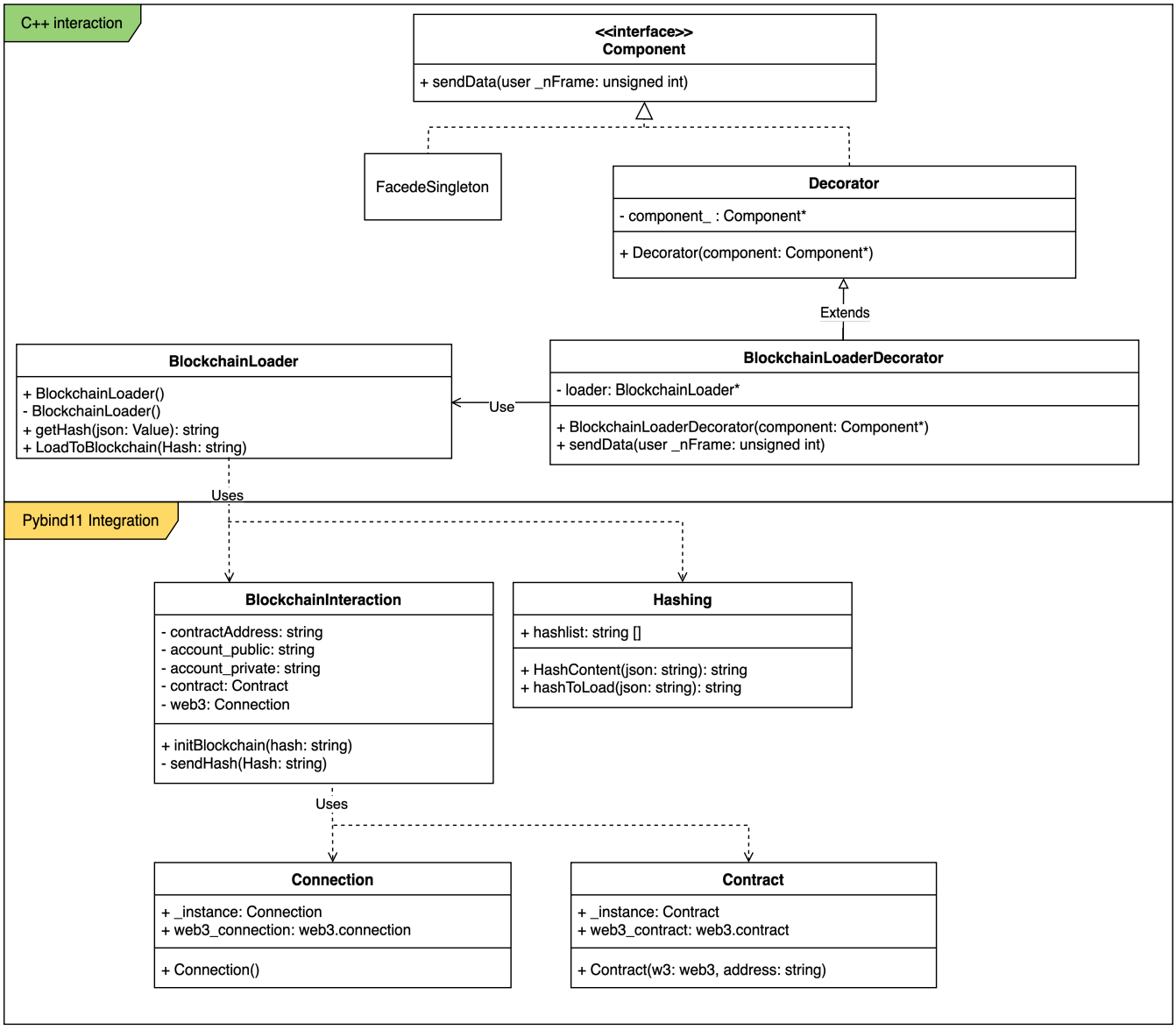


Figure 10. Portion of the class diagram extracted from the Tracker's code. The highlighted subsection provides a clarification over the mounting point selected for the introduction of the blockchain-related functionalities by leveraging the introduction of a Python section for the integration of the sub-module.

Instead, the proposed integration of the Ditto sub-module involved the extension of its connectivity capabilities through the development of a dedicated Java software component for filtering incoming messages. This integration methodology was chosen to enable more advanced functionalities required for interaction with blockchain technologies. In fact, at the time of this work, the module included a component responsible for receiving messages via the transmission medium and subsequently converting the corresponding messages into the Ditto Protocol format. This transformation process was achieved through the use of a JavaScript mapping function. However, the original implementation exhibited significant limitations, particularly in terms of extensibility and the ability to incorporate external libraries. To address the encountered issues, the integration of the sub-module involved the elaboration of a newly introduced mapping component. This was achieved

by incorporating the functionalities of the sub-module into a custom mapper implemented in Java, which retained the original core functionality while extending its capabilities. The redesigned component built upon the original mapper and is visually represented through the scheme in Figure 11.

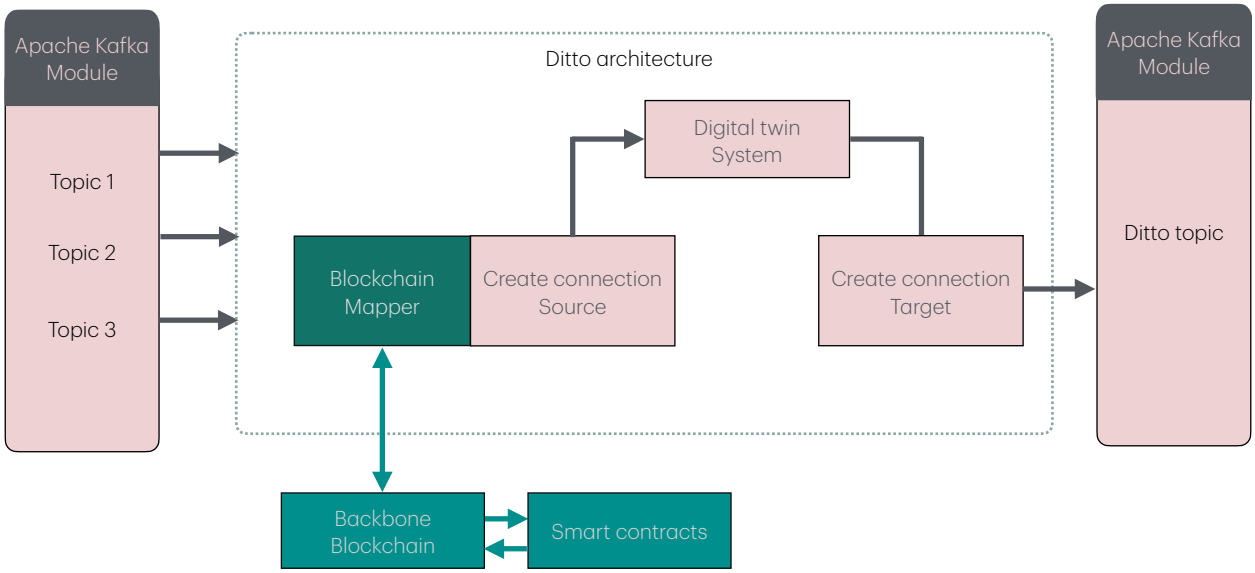


Figure 11. Schematic path followed by data in the AiWatch system with particular emphasis over the Ditto sub-components and the integrated mapper.

To further underline the symbiotic and transparent integration of blockchain communication, the proposed implementation ensured that all blockchain-related interactions were natively embedded within the Ditto sub-module. By avoiding reliance on external or third-party software, the design guaranteed seamless integration of blockchain functionalities directly into the core operational workflows. This tight coupling enhanced the transparency of data flow and reduced potential points of failure, enabling robust and secure communication. The approach capitalized on the inherent strengths of Ditto's modular architecture while preserving the autonomy and integrity of the blockchain connection mechanism.

The final result was developed following the standard procedures outlined in the Eclipse Ditto documentation. This approach allowed for the integration of the newly introduced functionalities and enabled the implementation of the Web3 connection channel. The provided class model in Figure 12 outlines the primary components of the sub-module code, offering critical insights into the structural organization and functionality. Furthermore, it highlights the modular division of responsibilities within the validation system, ensuring clarity for the various tasks performed by the component.

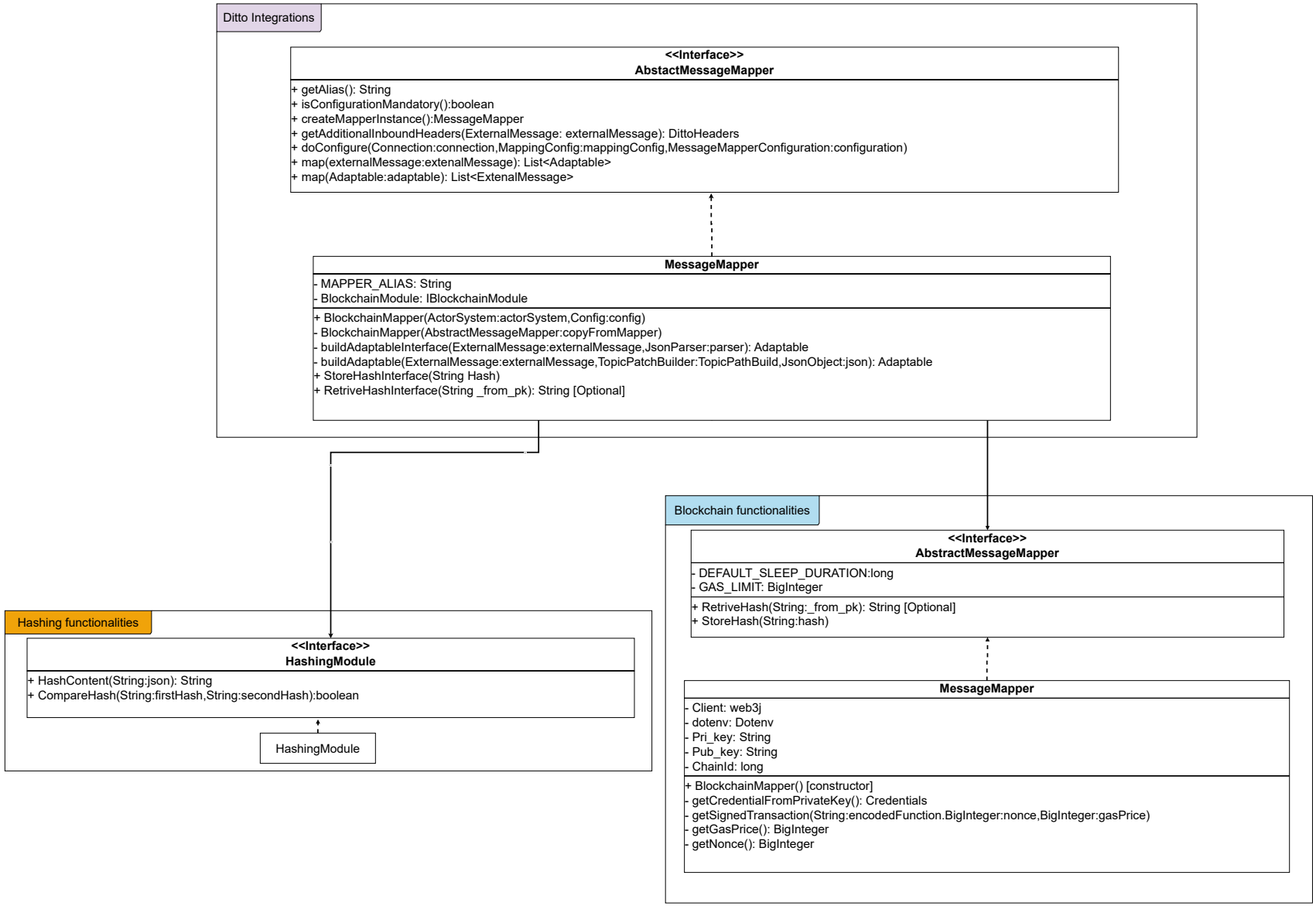


Figure 12. Complete class diagram of the developed Ditto sub-module.

6. Discussion

The results provided by this study demonstrated the viability of integrating blockchain technology with digital twins to enhance security, transparency, and performance in industrial applications. The implemented framework leveraged blockchain's tamper-proof nature to address critical vulnerabilities in digital twin systems, such as data integrity and unauthorized access. Security tests validated the robustness of the architecture proposed, effectively preventing unauthorized communication and ensuring data consistency. Moreover, the proposed case study illustrated the practical applications of this framework, showcasing how it could be deployed in real-world scenarios. For instance, fraudulent data transmissions were effectively intercepted, ensuring the reliability of the information used for decision-making in real time. Performance benchmarks also highlighted the scalability and adaptability of the system, particularly with the utilization of the QBFT consensus protocol, which demonstrated superior throughput and resilience under high transaction loads. Despite these advantages, challenges such as interoperability with existing digital twin solutions and the computational overhead of blockchain integration remain areas for further exploration. Future research could investigate lightweight blockchain models or Layer 2 scaling solutions to optimize system performance.

Author Contributions: Conceptualization, A.F. and S.V.; methodology, A.F. and S.V.; software, A.F. and S.V.; validation, A.F. and S.V.; formal analysis, A.F. and S.V.; writing—original draft preparation, A.F. and S.V.; writing—review and editing, A.F. and S.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Source code is available at the following link: https://github.com/StefanoVerrilli/Blockchain_Integration_AiWatch, accessed on 9 January 2025.

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

References

1. Qi, Q.; Tao, F.; Hu, T.; Anwer, N.; Liu, A.; Wei, Y.; Wang, L.; Nee, A.Y. Enabling technologies and tools for digital twin. *J. Manuf. Syst.* **2021**, *58*, 3–21. [CrossRef]
2. Sasikumar, A.; Vairavasundaram, S.; Kotecha, K.; Indragandhi, V.; Ravi, L.; Selvachandran, G.; Abraham, A. Blockchain-based trust mechanism for digital twin empowered Industrial Internet of Things. *Future Gener. Comput. Syst.* **2023**, *141*, 16–27.
3. Gehrmann, C.; Gunnarsson, M. A Digital Twin Based Industrial Automation and Control System Security Architecture. *IEEE Trans. Ind. Inform.* **2020**, *16*, 669–680. [CrossRef]
4. Karaarslan, E.; Babiker, M. Digital twin security threats and countermeasures: An introduction. In Proceedings of the 2021 International Conference on Information Security and Cryptology (ISCTURKEY), Ankara, Turkey, 2–3 December 2021; pp. 7–11.
5. Suhail, S.; Hussain, R.; Jurdak, R.; Oracevic, A.; Salah, K.; Hong, C.S.; Matulevičius, R. Blockchain-based digital twins: Research trends, issues, and future challenges. *ACM Comput. Surv. CSUR* **2022**, *54*, 1–34. [CrossRef]
6. Sadri, H.; Yitmen, I.; Tagliabue, L.C.; Westphal, F.; Tezel, A.; Taheri, A.; Sibenik, G. Integration of blockchain and digital twins in the smart built environment adopting disruptive technologies—A systematic review. *Sustainability* **2023**, *15*, 3713. [CrossRef]
7. Thakur, G.; Kumar, P.; Deepika; Jangirala, S.; Das, A.K.; Park, Y. An Effective Privacy-Preserving Blockchain-Assisted Security Protocol for Cloud-Based Digital Twin Environment. *IEEE Access* **2023**, *11*, 26877–26892. [CrossRef]
8. Putz, B.; Dietz, M.; Empl, P.; Pernul, G. Ethertwin: Blockchain-based secure digital twin information management. *Inf. Process. Manag.* **2021**, *58*, 102425. [CrossRef]
9. Salim, M.M.; Comivi, A.K.; Nurbek, T.; Park, H.; Park, J.H. A blockchain-enabled secure digital twin framework for early botnet detection in IIoT environment. *Sensors* **2022**, *22*, 6133. [CrossRef]
10. Fan, C. Blockchain-Based Design for Performant Peer-to-Peer Energy Trading Systems. Ph.D. Thesis, University of Alberta, Edmonton, AB, Canada, 2023.

11. Berdik, D.; Otoum, S.; Schmidt, N.; Porter, D.; Jararweh, Y. A survey on blockchain for information systems management and security. *Inf. Process. Manag.* **2021**, *58*, 102397. [CrossRef]
12. Ferone, A.; Della Porta, A. A blockchain-based infection tracing and notification system by non-fungible tokens. *Comput. Commun.* **2022**, *192*, 66–74. [CrossRef]
13. Dutta, P.; Choi, T.M.; Somani, S.; Butala, R. Blockchain technology in supply chain operations: Applications, challenges and research opportunities. *Transp. Res. Part E Logist. Transp. Rev.* **2020**, *142*, 102067. [CrossRef] [PubMed]
14. Minoli, D.; Occhiogrosso, B. Blockchain mechanisms for IoT security. *Internet Things* **2018**, *1*, 1–13. [CrossRef]
15. Al-Jaroodi, J.; Mohamed, N. Blockchain in industries: A survey. *IEEE Access* **2019**, *7*, 36500–36515. [CrossRef]
16. Lee, D.; Lee, S.H.; Masoud, N.; Krishnan, M.; Li, V.C. Integrated digital twin and blockchain framework to support accountable information sharing in construction projects. *Autom. Constr.* **2021**, *127*, 103688. [CrossRef]
17. Hao, Y.; Li, Y.; Dong, X.; Fang, L.; Chen, P. Performance analysis of consensus algorithm in private blockchain. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, China, 26–30 June 2018; pp. 280–285.
18. Pahlajani, S.; Kshirsagar, A.; Pachghare, V. Survey on private blockchain consensus algorithms. In Proceedings of the 2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT), Chennai, India, 25–26 April 2019; pp. 1–6.
19. Lin, S.Y.; Zhang, L.; Li, J.; Ji, L.L.; Sun, Y. A survey of application research based on blockchain smart contract. *Wirel. Netw.* **2022**, *28*, 635–690. [CrossRef]
20. Terzi, S.; Zacharaki, A.; Nizamis, A.; Votis, K.; Ioannidis, D.; Tzovaras, D.; Stamelos, I. Transforming the supply-chain management and industry logistics with blockchain smart contracts. In Proceedings of the 23rd Pan-Hellenic Conference on Informatics, Nicosia, Cyprus, 28–30 November 2019; pp. 9–14.
21. Sultana, T.; Almogren, A.; Akbar, M.; Al-Zuair, M.; Ullah, I.; Javaid, N. Data sharing system integrating access control mechanism using blockchain-based smart contracts for IoT devices. *Appl. Sci.* **2020**, *10*, 488. [CrossRef]
22. Nielsen, C.P.; Da Silva, E.R.; Yu, F. Digital twins and blockchain–proof of concept. *Procedia CIRP* **2020**, *93*, 251–255. [CrossRef]
23. ERC-1967. ERC-1967: Proxy Storage Slots. 2019. Available online: https://eips.ethereum.org/EIPS/eip-1967 (accessed on 29 December 2024).
24. ERC-2535. ERC-2535: Diamonds, Multi-Facet Proxy. 2020. Available online: https://eips.ethereum.org/EIPS/eip-2535 (accessed on 29 December 2024).
25. Bui, V.C.; Wen, S.; Yu, J.; Xia, X.; Haghighi, M.S.; Xiang, Y. Evaluating upgradable smart contract. In Proceedings of the 2021 IEEE International Conference on Blockchain (Blockchain), Melbourne, Australia, 6–8 December 2021; pp. 252–256.
26. ERC-1822. ERC-1822: Universal Upgradeable Proxy Standard (UUPS). 2019. Available online: https://eips.ethereum.org/EIPS/eip-1822 (accessed on 29 December 2024).
27. Salehi, M.; Clark, J.; Mannan, M. Not so immutable: Upgradeability of smart contracts on ethereum. In Proceedings of the International Conference on Financial Cryptography and Data Security, Brač, Croatia, 1–5 May 2022; Springer: Berlin/Heidelberg, Germany; pp. 539–554.
28. Malik, S.; Bandara, H.D.; van Beest, N.R.; Xu, X. Smart contracts' upgradability for flexible business processes. In Proceedings of the International Conference on Business Process Management, Utrecht, The Netherlands, 11–15 September 2024; Springer: Berlin/Heidelberg, Germany; pp. 55–70.
29. Fuller, A.; Fan, Z.; Day, C.; Barlow, C. Digital twin: Enabling technologies, challenges and open research. *IEEE Access* **2020**, *8*, 108952–108971. [CrossRef]
30. Botín-Sanabria, D.M.; Mihaita, A.S.; Peimbert-García, R.E.; Ramírez-Moreno, M.A.; Ramírez-Mendoza, R.A.; Lozoya-Santos, J.d.J. Digital twin technology challenges and applications: A comprehensive review. *Remote Sens.* **2022**, *14*, 1335. [CrossRef]
31. Celik, Y.; Petri, I.; Rezgui, Y. Leveraging BIM and blockchain for digital twins. In Proceedings of the 2021 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), Cardiff, UK, 21–23 June 2021; pp. 1–10.
32. Götz, C.S.; Karlsson, P.; Yitmen, I. Exploring applicability, interoperability and integrability of Blockchain-based digital twins for asset life cycle management. *Smart Sustain. Built Environ.* **2020**, *11*, 532–558. [CrossRef]
33. Koppu, S.; Somayaji, S.R.K.; Meenakshisundaram, I.; Wang, W.; Su, C. Fusion of blockchain, IoT and artificial intelligence—A survey. *IEICE Trans. Inf. Syst.* **2022**, *105*, 300–308. [CrossRef]
34. Godager, B.; Onstein, E.; Huang, L. The concept of enterprise BIM: Current research practice and future trends. *IEEE Access* **2021**, *9*, 42265–42290. [CrossRef]
35. Hyperledger Besu. Hyperledger Besu. 2024. Available online: https://besu.hyperledger.org (accessed on 29 December 2024).
36. Docker, I. Docker: The Open Platform for Distributed Applications. 2023. Available online: https://www.docker.com (accessed on 26 November 2023).
37. Foundation, E. Solidity: Smart Contract Programming Language. 2023. Available online: https://soliditylang.org (accessed on 26 November 2023).

38. Hyperledger Besu. Clique Proof of Authority Consensus. 2024. Available online: https://besu.hyperledger.org/private-networks/concepts/poa (accessed on 29 December 2024).
39. Saltini, R.; Hyland-Wood, D. IBFT 2.0: A safe and live variation of the IBFT blockchain consensus protocol for eventually synchronous networks. *arXiv* **2019**, arXiv:1909.10194.
40. Kahmann, F.; Honecker, F.; Dreyer, J.; Fischer, M.; Tönjes, R. erformance Comparison of Directed Acyclic Graph-Based Distributed Ledgers and Blockchain Platforms. *Computers* **2023**, *12*, 257. [CrossRef]
41. Ucbas, Y.; Eleyan, A.; Hammoudeh, M.; Alohaly, M. Performance and Scalability Analysis of Ethereum and Hyperledger Fabric. *IEEE Access* **2023**, *11*, 67156–67167. [CrossRef]
42. Spoleto, A. J.; Staiano, A.; Hauber, G.; Lettiero, M.; Barra, P.; Camastra, F. Deep learning-based robust head pose estimation. In Proceedings of the 32nd Italian Workshop on Neural Networks—WIRN2024, Vietri Sul Mare, Italy, 5–7 June 2024.