An Enterprise Architecture Approach to Semantic Blockchain Interoperability

***Abstract*—Blockchain technology has revolutionized the way data is stored and accessed in a decentralized manner. However, the lack of interoperability between such systems is an ongoing challenge hindering their wider adoption. This document proposes a two-part solution composed of activities that aim to enhance semantic interoperability between homogeneous and heterogeneous blockchain systems. The first part are the design-time activities that consist of constructing an Archimate model, extracting its Resource Description Framework (RDF) ontology, and assessing its correctness utilizing a semantic reasoner. The second part are the runtime activities that involve leveraging the resulting ontology in a supply chain management application to validate transactions among participants in a network of systems. The evaluation results are promising, demonstrating that a shared ontology can support a transparent and accurate transaction validation approach. Thus, this work is a significant step in proving that distributed ledger technologies can benefit from enterprise architecture techniques to improve their interoperability.**

I. Introduction

Distributed ledger technology (DLT) refers to networks of computers that store databases in a distributed way, distinguishing them from traditional databases confined to a single server. This decentralized structure enhances security by making data tampering more challenging for malicious entities. Blockchains store data in a chain of blocks and are a specific type of DLT. Each block contains multiple transaction records, and once appended to the chain, a block becomes immutable. This mechanism establishes a permanent and transparent ledger of all transactions conducted in a network. In addition to being used to record and track monetary transactions, blockchain technology can potentially support applications in several other domains, including supply chain management, voting systems, and identity verification [1].

Private or permissioned blockchains are distributed ledger systems where access to the network and validation of transactions is limited to a select group of entities. These blockchains provide enterprises with several advantages in comparison with permissionless blockchains. Firstly, data confidentiality is enhanced since sensitive information is shared only among trusted participants. Secondly, they are highly scalable due to reduced network congestion and faster transaction processing since the number of participants is reduced compared to public blockchains. Additionally, private blockchains offer increased control and governance, enabling businesses to set rules and consensus mechanisms that align with their specific requirements. Furthermore, their usage enhances compliance, which makes them a valuable asset for various use cases in industries with high regulation [2].

Interoperability, which refers to the ability of different systems and components to work together efficiently, is a quality attribute that has four layers: the technical (further subdivided into functional and structural), the semantic (including interpreting information format and its meaning), the organizational, and the legal [3]. Regarding specifically the semantic layer of interoperability, on the one hand information format pertains to the structure and representation of data exchanged between systems. This includes data types, encoding methods, and communication protocols. On the other hand, understanding the meaning embedded in the information involves interpreting the data context, semantics, and relationships, ensuring that the exchanged information is correctly understood by the systems consuming it.

Over the last few years, there have been many attempts to introduce a solution that will mitigate the interoperability issues of blockchain systems [4]. However, few solutions focus on the semantic layer. While each layer has its significance, the semantic layer will play a key role in increasing interchain communication efficiency [5]. By building systems on a common ground of understanding, actions conducted among systems with a high volume of cross-chain transactions become faster, as there is less friction to translate concepts from one blockchain to the other.

The objective of this work is to explore enterprise architecture (EA) mechanisms to produce an ontology [6] that individual systems can leverage to build plugins, which are small and flexible components executing their business logic on top of a base architecture that includes the necessary elements to achieve interoperability. EA offers standardizable methods in the form of models that can facilitate the intercommunication of distributed ledger technology systems in the semantic layer by reducing ambiguity in valuable processes such as interchain transactions.

A proof-of-concept (POC) presents the practical aspects of this approach. Its scope is a network of permissioned blockchains owned by organizations with interoperation needs with fellow members of a supply chain consortium. The decision to restrict the POC to private blockchains comes from the fact that systems involving public blockchains operate with some inherently different concerns and requirements as mentioned above. Therefore, designing an EA model targeting both would complicate the solution prematurely.

The results drawn from the conducted experiments are encouraging, showing that the integrity of the transaction validation process is enhanced when following the proposed approach.

In summary, this thesis introduces an EA model that supports semantic interoperability between blockchain systems and implements a POC that will leverage the ontology extracted from it. By proving the applicability and effectiveness of the proposed solution, the aim is to show that EA techniques can be valuable for solving interoperability issues in blockchain systems.

This paper is structured as follows. Section II gives an overview of the architecture and use case that support the POC, followed by Section III, which describes the proposed solution in detail. Next, Section IV provides the evaluation strategy and its outcome, and Section V conducts a comprehensive literature review. Lastly, the conclusions drawn from this study are discussed in Section VI, and Section VII outlines future work.

II. Preliminaries: Cacti and Supply Chain

This section presents important knowledge concerning the proposed solution. Specifically, it provides an in-depth analysis of Hyperledger Cacti [7], as it forms the basis of the architecture that supports business logic plugins (BLPs) used for the interoperation of supply chain-related entities. Additionally, it describes the use case that will demonstrate the capabilities of the proposed solution along the remainder of the paper.

A. Hyperledger Cacti

Blockchain gateways are intermediary systems that facilitate interoperability between different blockchain networks. They act as bridges, enabling data transfer across disparate blockchain platforms, allowing them to exchange information and assets. One example of a middleware framework that adopts the concept of blockchain gateways is Hyperledger Cacti. Built on the Hermes model [8], Cacti is an open-source solution that provides a fault-tolerant middleware layer supporting blockchain interoperability (BI).

By implementing the protocols and mechanisms introduced by Hermes, Cacti enables secure and reliable communication between diverse blockchain networks, opening up opportunities for cross-chain transactions and the execution of smart contracts, which are self-executing contracts with the terms of the agreement written in a programming language like Solidity [9]. These contracts automatically execute and enforce themselves when predetermined conditions are met, without the need for intermediaries or third parties.

The main components of Cacti are the following:

- *Core Framework*: The core framework forms the foundation of Cacti. It provides the necessary infrastructure and interfaces for developing blockchain integration modules. It includes the main runtime, messaging protocols, and a plugin system for extensibility.
- *Connector*: Connectors are modules that enable communication and interaction with various blockchain platforms. Each connector is designed to interface with a specific blockchain platform, such as Avalanche [10], Cosmos [11], and Corda [12], to name a few. Connectors allow Cacti to interact with different blockchains simultaneously.
- *Consortium Management*: Cacti includes a consortium management component, which facilitates the creation and management of consortium networks. It allows multiple blockchain platforms to collaborate and communicate securely within a shared network.
- *Plugin System*: Cacti employs a plugin system that allows developers to extend its functionality by creating custom connectors and modules. This extensibility enables integration with new blockchain platforms or the addition of custom features and functionalities.
- *APIs and SDKs*: Cacti provides a set of application programming interfaces (APIs) and software development kits (SDKs) that simplify the integration process and allow developers to build applications on top of the framework. These APIs and SDKs offer standardized interfaces for accessing the functionality provided by Cacti.

Figure 1 depicts the high-level interactions of components utilizing Cacti. The client communicates with multiple BLPs via Representational State Transfer (REST) APIs. The BLPs then send the data to Cacti Core, which serves as the central component responsible for coordinating the exchange of data between the plugins and the connectors via connector APIs. Lastly, each connector is responsible for interfacing with the respective blockchain.

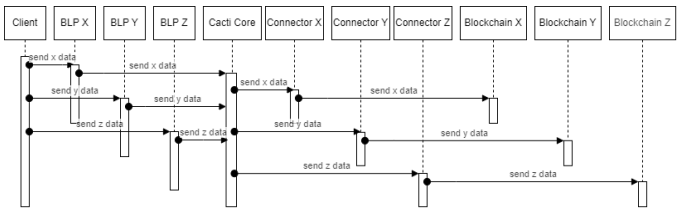

Fig. 1: UML sequence diagram of an architecture supported by Hyperledger Cacti

B. Supply Chain Use Case

The use case chosen for the POC focuses on the supply chain domain. This choice comes from the fact that this business sector inherently utilizes ambiguous terms to describe processes, making it adequate for demonstrating the value of the proposed solution.

The use case consists of three sequential data transfers among three entities, where each entity leverages a different type of blockchain for business-specific reasons irrelevant to the study. A data transfer is the process of copying information from one DLT to another, and it involves specific data (e.g., manufacturer information) in most cases. This operation can involve multiple steps, including an optional intermediate processing step that may manipulate or analyze the data before sending it to the target chain. Data transfers are vital for ensuring the interoperability and exchange of valuable information across blockchains [13].

The data model comprises a business role, function, item, and process. That is, a *Supplier* may *Supply* a *Bamboo Harvest* to a *Manufacturer* in the context of a *Product Sale* and so on. There is an obvious concern for preserving privacy through the entire interoperation process among the three entities, as in most cases, the transferred data includes sensitive business information.

Figure 2 depicts a sample flow of transactions taking place among the three entities of the use case as a Business Process Model and Notation (BPMN) diagram [14]. To achieve interoperability between the three entities, Cacti is used. When an event occurs in one system, a smart contract is invoked. The Cacti Core then interacts with the appropriate Connector (Fabric, Besu, or Quorum) to update the respective blockchain.

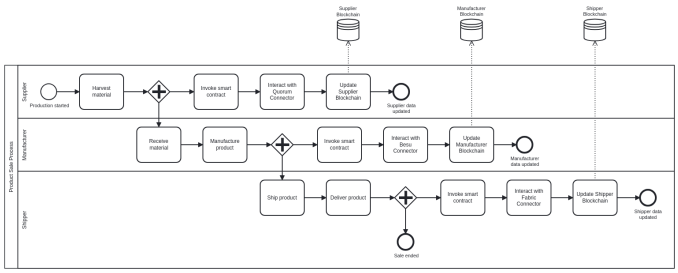

Fig. 2: Supply chain use case BPMN diagram

III. Ontologically-Guided Blockchain Interoperability

This section presents a solution for achieving interoperability between two or more blockchain systems by leveraging a common ontology that derives from an EA model. The approach is divided into design-time and runtime activities of the interoperation process to ensure standardization. The former are the activities that take place before the interoperation process has begun. The latter are the activities that take place during the interoperation process. In order to aid the reproduction of this approach, the complete source code of its implementation is published in a public repository.

A. Design-Time Activities

At design time, a model is created using Archimate, a modelling language for EA [15]. This model is the foundation for producing a shared vocabulary as an RDF/XML ontology which represents the structural and conceptual aspects of the Archimate model in a machine-readable format [16]. Lastly, the HermiT reasoner is used to verify the extracted ontology [17].

1) Enterprise Architecture Model Design: Ideally, this process is conducted by consortium stakeholders to ensure alignment among participating entities. As they possess rich insights about the meaning of terms and the specific format of messages of each business area, close inter-collaboration determines the scope and contents of the EA model. Therefore, this should be an iterative process, and the model may change significantly as the knowledge grows. Furthermore, the

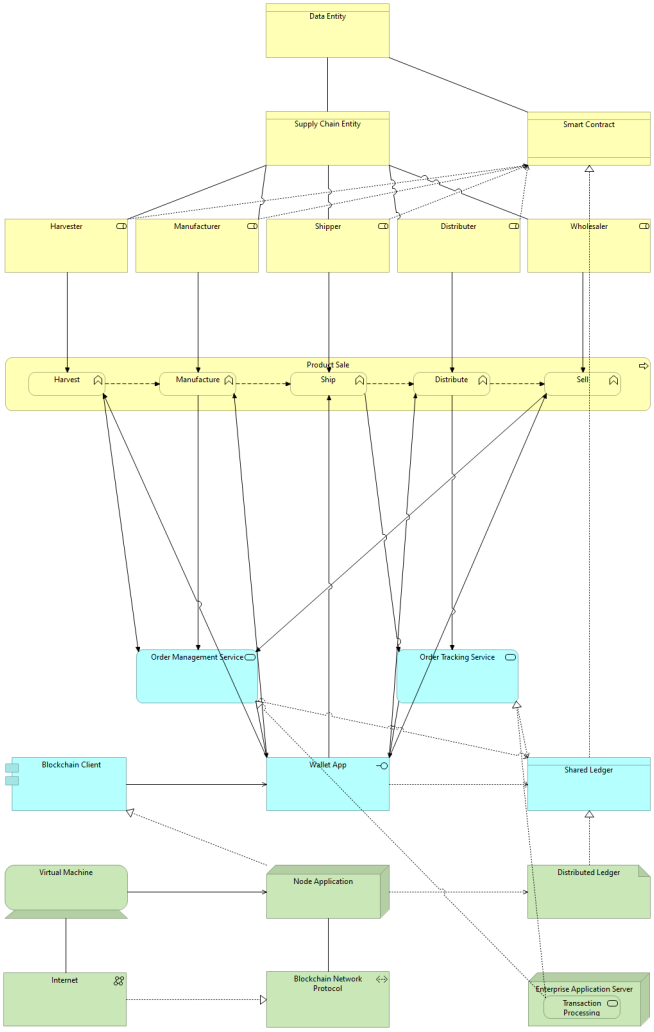

Fig. 3: Supply chain blockchain interoperability EA model in Archimate

European Interoperability Reference Architecture (EIRA) [18] should guide the design to ensure regulatory compliance.

Figure 3 depicts the three layers of the supply chain-specific EA created for the use case that was defined in Subsection II.B. The upper layer portrays the Business layer with the relations between the key business entities. Those are the business roles, functions, processes, and items that will serve as the main study of this paper. The middle layer represents the Application layer with the software applications that support the business layer. It includes components such as application services and blockchain interfaces. Finally, the bottom layer illustrates the Technology layer with the technological infrastructure that supports the application layer. Its elements are the infrastructure and networks that enable blockchain transactions.

2) Ontology Extraction and Assessment: After designing the Archimate model, the next step is to extract its model exchange file in XML and convert it into an RDF ontology. A

third-party tool called archimate2rdf designed specifically for converting Archimate models into RDF format handles this.

An RDF ontology is favored for managing data from multiple sources because unlike alternative ontology technologies, it accommodates various data types, ensuring seamless integration among diverse systems. Moreover, its semantic richness enables it to capture nuanced meanings and contextual relationships within the information.

Once the RDF ontology is ready, the assessment process using HermiT can proceed. HermiT is a reasoner that can analyze ontologies and infer additional knowledge based on the logical rules defined in the ontology. This process helps identify redundancies and missing information within the ontology.

HermiT can also check for the satisfiability and consistency of the ontology. Satisfiability determines if all the defined classes of the ontology have at least one instance, while consistency checks if there are no logical contradictions within the ontology. These checks are essential to ensure the ontology accurately represents the intended knowledge.

```
<NamedIndividual
      rdf:about="urn:uuid:id-6d7fc1370fcb48959daae518ea90b265">
   <rdf:type
         rdf:resource="http://bp4mc2.org/def/archimate#BusinessRole"/>
   <archimate:association
         rdf:resource="urn:uuid:id-67ccb819d7284462a8b9045f23e437d0"/>
   <archimate:triggering
         rdf:resource="urn:uuid:id-00f2f859356543cb8c0b8212e2731e1c"/>
   <archimate:writeAccess
         rdf:resource="urn:uuid:id-a5e7d160476c449ca342489c9b3f3d9f"/>
   <rdfs:label xml:lang="en">Manufacturer</rdfs:label>
</NamedIndividual>
```

Listing 1: Sample *Manufacturer* business role entity

```
<NamedIndividual
      rdf:about="urn:uuid:id-00f2f859356543cb8c0b8212e2731e1c">
   <rdf:type
         rdf:resource="http://bp4mc2.org/def/archimate#BusinessFunction"/>
   <archimate:flow
         rdf:resource="urn:uuid:id-94c418f274254d8a920235a645303a6e"/>
   <archimate:triggering
         rdf:resource="urn:uuid:id-01a27c33b42045f58a72188c63a03964"/>
   <archimate:writeAccess
         rdf:resource="urn:uuid:id-98c01165200a4f37872c7b85865a05d4"/>
   <rdfs:label xml:lang="en">Manufacture</rdfs:label>
</NamedIndividual>
```

Listing 2: Sample *Manufacture* business function entity

```
<NamedIndividual
      rdf:about="urn:uuid:id-eda65277dfba4fd987007920fc06210c">
   <rdf:type
         rdf:resource="http://bp4mc2.org/def/archimate#BusinessProcess"/>
   <archimate:composition
         rdf:resource="urn:uuid:id-00f2f859356543cb8c0b8212e2731e1c"/>
   <archimate:composition
         rdf:resource="urn:uuid:id-561dbe4948034f29b950c9ee130847ff"/>
   <archimate:composition
         rdf:resource="urn:uuid:id-94c418f274254d8a920235a645303a6e"/>
   <archimate:composition
         rdf:resource="urn:uuid:id-e939a980803d4261825d201a2d7d4d83"/>
   <archimate:composition
         rdf:resource="urn:uuid:id-f32e4603611141d899779f82e71be7ba"/>
   <rdfs:label xml:lang="en">Product Sale</rdfs:label>
</NamedIndividual>
```

Listing 3: Sample *Product Sale* business process entity

By using HermiT to assess the RDF ontology derived from the Archimate model, one gains valuable insights into the quality of the ontology. This analysis can help uncover potential issues or improvements in the design of the initial model, ensuring that the resulting ontology aligns with the intended architectural knowledge.

Listings 1, 2 and 3, display the format of sample business entities in the ontology.

B. Runtime Activities

At runtime, the integration process unfolds through a series of steps. First, the transaction context is defined, which consists of stating the target business role and the business process in which the transaction occurs. Afterwards, the source system sends the data to the shared Cacti BLP middleware. There, the data is mapped and converted to conform to the ontology. Finally, the BLP validates the data against the predefined constraints and rules. If the data is deemed valid, it is delivered to the target system and subsequently updated in the target blockchain.

1) Entity Mapping and Conversion: Mapping refers to retrieving the name of a relevant entity from the ontology. It can be either the same entity or a semantically similar one. Conversion refers to building the corresponding ontology role object based on the source role object by extracting all the available data. Both mapping and conversion processes involve querying the ontology in SPARQL by utilizing an open-source library called rdflib.js [19].

The first step is to create an rdflib.js store object that holds the ontology data. This store acts as a container for RDF triples, where each triple consists of a subject, predicate, and object. The store then reads the ontology file and populates itself with the existing triples in the ontology.

SPARQL, a standard query language for RDF data, is used to query the store. rdflib.js provides a convenient API to execute queries on the store. A SPARQL query can specify conditions to filter entities based on specific criteria, such as their properties or relationships with other entities. For example, one can query for all entities of a specific type or entities that possess a particular property value.

2) Transaction Validation: The transaction validation process consists of two steps. The first is data validation, and the second is state validation. Successful completion of both steps ensures a valid transaction. If either step is unsuccessful, the transaction cannot occur, and the remaining process should abort.

Data validation refers to checking whether the properties of the object are valid. Specifically, it ensures that the values are within the expected boundaries stipulated in the EA model. This step is depicted in Algorithm 1. This algorithm validates data in the ontology role input object by checking specific fields containing the word *Format*. For each of these fields, it ensures a corresponding non-format field exists. If not, it moves to the next field. The code validates the format field as

a regular expression; if invalid, it returns false. It then checks if the non-format field matches the specified pattern. If not, it also returns false. If all checks pass, it returns true.

Algorithm 1 Data Validation Algorithm

Input: ontologyRoleObj: Object
Output: isValid: Boolean

```
Function validateData(ontologyRoleObj)
    fieldsToCompare ← getFieldsContainingWord(ontologyRoleObj, "Format")
    for i ← 0 to length of fieldsToCompare do
        formatFieldName ← fieldsToCompare[i] fieldName ← remove "Format" from formatFieldName
        if fieldName not in ontologyRoleObj then
            continue
        end
        fieldValue ← ontologyRoleObj[fieldName] formatValue ← ontologyRoleObj[formatFieldName]
        if formatValue is not a regular expression then
            return false
        end
        if fieldValue does not match the format specified in formatFieldName then
            return false
        end
    end
    return true
end
```

State validation refers to assuring that the transaction is legal. In other words, a transition from or to specific business functions can occur in the current business process according to the EA model. This step is depicted in Algorithm 2. This algorithm performs a validation check on the input parameters *sourceRole*, *targetRole*, and *process*. It first ensures that *sourceRole* is not falsy; if it is, the method returns false. Then, it attempts to fetch a list of functions related to the specified *process* using an RDF query and returns false if no functions are found. The method then iterates through the list of functions, checking if any of them are triggered by the given *sourceRole* or *targetRole*. The method returns true if *sourceFunction* is not null and either *targetFunction* is not null and the *areAdjacent* method confirms the adjacency of *sourceFunction* and *targetFunction*, or if *targetRole* is falsy.

IV. Evaluation

It is essential to obtain a comprehensive understanding of the advantages and limitations of a novel approach. Conducting both accuracy and performance analyses allows for a rigorous assessment of its capabilities across various dimensions.

Accuracy analysis scrutinizes the reliability of the system irrespective of its operational load. This examination serves the purpose of discerning the inherent capabilities of the system under controlled conditions. It is also pivotal in ensuring that the system effectively fulfills its intended objectives and operates correctly, laying the foundation for a thorough evaluation.

Algorithm 2 State Validation Algorithm

Input: sourceRole: String, targetRole: String, process: String
Output: isValid: Boolean

```
Function validateState(sourceRole, targetRole, process):
    if sourceRole is empty then
        return false
    end
    processObj ← RDFQuerier.fetchDetailsByLabel(process)
    if processObj is null then
        return false
    end
    foreach funcUri in processObj.get('archimate#composition') do
        funcName ← RDFQuerier.fetchLabelByUri(funcUri)
        func ← find function in this.businessFunctions where func.entity equals funcName
        if func.triggeredBy equals sourceRole then
            sourceFunction ← func
        end
        if func.triggeredBy equals targetRole then
            targetFunction ← func
        end
        add func to funcs
    end
    return (sourceFunction ≠ null) ∧ ((targetFunction ≠ null ∧ areAdjacent(sourceFunction, targetFunction, funcs)) ∨ ¬targetRole)
end
```

Conversely, performance analysis explores the system's behavior under diverse load conditions, simulating real-world scenarios of varying demand and stress levels. This data enables the assessment of scalability, resilience, and the system's ability to perform consistently in a production environment.

Integrating accuracy and performance analyses is crucial for a holistic evaluation. This method minimizes the risk of oversight and mitigates potential biases. Furthermore, it supports the principle of continuous improvement. As the system evolves, periodic reevaluation through these lenses can assist in addressing emerging challenges.

A. Experimental Setup

The experimental infrastructure consisted of three test ledgers (one for each supply chain entity) interoperating with Cacti. The three test ledgers were Besu [20], Quorum [21], and Fabric [22]. These ledgers simulated different blockchain networks and enabled the deployment and interaction with smart contracts. Each component ran on a separate Docker [23] container on a single machine equipped with an AMD Ryzen 7 3750H processor and 16GB of RAM.

The deployment of smart contracts was a critical aspect of the experimental setup, as each of them handled different supply chain aspects. The *BambooHarvestRepository.sol* and *BookshelfRepository.sol* contracts were written in Solidity and the former managed bamboo harvest records, while the latter handled bookshelf records. The *Shipment.ts* script written in Go represented a smart contract for managing shipments. The bamboo harvest and bookshelf smart contracts were deployed in the Besu and Quorum blockchains respectively while the shipment smart contract was deployed on the Fabric ledger.

The deployment process involved creating test accounts, generating contract addresses, and storing contract artifacts in the keychain plugin to manage cryptographic keys securely [24]. It also included specifying the target organizations, channel, and policy that determines which members of each organization can approve a transaction involving a specific smart contract [25].

The purpose of the above setup was to provide a self-contained and local environment that allowed for assessing the effectiveness of the proposed solution by evaluating the POC that puts it into practice. Thus, it provided the necessary functionality for the experiments without requiring a complex and time-consuming process.

B. Accuracy Analysis

The chosen metric for the accuracy analysis was the error rate of the POC when validating transactions (Tx) among blockchain systems. Therefore, the approach followed was to conduct experiments with valid and invalid transactions to ascertain if the solution correctly identified each case.

Each validation process scrutinizes two main elements. Those are the transaction context and the object that initiates the transaction. If either of them is invalid, the transaction should not complete.

Specifically, a context is valid when its target role and business process entities are present in the ontology. Similarly, an object is in a legal state when its source role entity exists in the ontology and its fields conform to the specified format.

In the first experiment, both the context and the object were valid. Here the transaction was deemed successful as expected because all steps passed.

The second experiment had an invalid context and valid object. This time, despite the mapping step executing successfully, the validation step failed because of its invalid context.

The third experiment involved a context that was valid and an invalid object. Here, the mapping step was unsuccessful, so the validation failed as it did not even get a chance to execute.

In the fourth experiment, both the context and the object were invalid. Again, the mapping failed, so the transaction validation was deemed unsuccessful.

The experiment results are summarized in Table I.

C. Performance Analysis

The metric used for the performance analysis was the processing time to complete a transaction among two separate blockchain systems in the POC.

Experiment#	Context	Object	Valid Tx	Expected
1	Valid	Valid	✓	Yes
2	Invalid	Valid	×	Yes
3	Valid	Invalid	×	Yes
4	Invalid	Invalid	×	Yes

TABLE I: Experiment results

Given that the proposed solution is supported by the infrastructure of Hyperledger Cacti, it is worth noting that the evaluation of the performance of the former is framed to that of the latter.

Experiments were conducted for three distinct groups of ”low,” ”normal,” and ”high” load profiles corresponding to a rate of 10, 50, and 100 simultaneous transactions. These three load profiles are the most common in an enterprise ecosystem. Furthermore, worker threads permitted the parallel execution of these transaction load groups.

To extract objective and unbiased results, each transaction load group execution was repeated 100 times and their mean time was calculated. While performing the first transactions, the systems may not have reached their processing potential, so the first iterations can be considered a warm-up phase.

Figure 4 depicts the impact on processing time as the rate of simultaneous transactions increases before and after integrating the proposed solution. One may interpret the graph as the processing overhead imposed on the participating systems when leveraging the approach presented in this work.

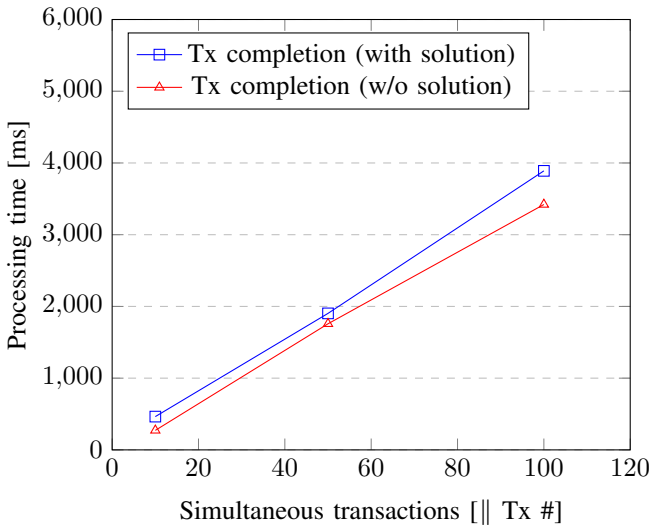

Fig. 4: Overhead of proposed solution

D. Result Discussion

The experiments show that the POC leveraging the proposed solution identifies valid and invalid transactions among systems with 100% accuracy. The benefits of this outcome are immense, as the interoperation process becomes transparent and efficient.

The increase in the processing overhead while utilizing the proposed solution indicates possible scalability issues. Nev-

ertheless, the overhead is considered negligible for enterprise standards until a certain point. Thus, it is suitable for private ecosystem usage where simultaneous transactions typically do not exceed this threshold, and data quality is of foremost concern.

In other words, it is acceptable to sacrifice system performance for reliability. As a result, the capabilities of the individual blockchain systems get enhanced by reducing duplication of effort, improving data integrity and consistency, and enabling new decentralized applications and services to be built on top of them securely.

Another observation is the low cost and ease of extensibility of the solution. While the design and consequent change management of modifying the EA model involve resources, the possibility of automation of the remaining processes implies that spending does not increase for those.

The applicability of the proposed solution is ample. Every network that involves complex transactions among participating systems would gain from this solution. Thus, this approach could potentially impact many industries that actively leverage blockchains and, perhaps, encourage others to migrate their current infrastructure to DLT.

E. Considerations

A fundamental constraint of the solution lies in the need to establish a common ontology. This requires consensus and collaboration between the blockchain systems involved. The ontology must define the concepts, relationships, and properties relevant to the specific domain in which the systems operate. However, reaching an agreement on this common ontology may present challenges, as different systems may have varying perspectives and terminology.

Another significant constraint is the availability of transaction context. The solution assumes that information regarding the target entity and the process in which the transaction takes place is accessible. This contextual knowledge is crucial for accurately validating and ensuring the integrity of transactions. However, obtaining and maintaining this contextual information may introduce complexity and overhead, particularly in decentralized and dynamic blockchain ecosystems.

Data conversion and mapping are essential steps for achieving interoperability. The solution assumes that a clear mapping can be established between the concepts and properties of the source blockchain system and the common ontology. This mapping process is critical for seamless data exchange between the systems. However, mapping complex and heterogeneous data structures can be non-trivial, requiring careful analysis and consideration of data semantics and structural differences.

Successful data conversion leads to the subsequent step of data validation. The solution assumes that the converted data can be validated against the constraints and rules defined within the common ontology. This validation ensures that the data adheres to the ontology's specifications and maintains its integrity throughout the interoperability process. Nonetheless, validating data against complex ontologies can be computationally intensive, particularly when dealing with large-scale and dynamic systems.

The POC introduces additional assumptions to simplify the implementation. It assumes that all participating entities utilize the same backend and frontend instances, reducing complexity for demonstration purposes. However, in real-world scenarios, entities would typically have their own independently developed backend and frontend instances, necessitating additional considerations for achieving interoperability.

Furthermore, the POC assumes trivial mapping and conversion steps due to the common backend model shared by all entities involved. This assumption alleviates the ambiguity that may arise when models are defined separately. In real-world scenarios, where different entities may have distinct models, the mapping and conversion processes would require careful handling of potential conflicts and inconsistencies.

The solution assumes validation of the business process defined in the EA. It expects a sequential execution of functions triggered by the source and target entities, with the latter function succeeding the former. However, ensuring adherence to the defined business process introduces challenges, as it requires coordination and synchronization between the participating systems.

Additionally, the solution assumes fixed transactions among three clearly-defined business entities for demonstration purposes. Therefore, it is limited in scope and does not apply directly to other use cases. To extend the solution's applicability, code generation techniques would be necessary to adapt it to different scenarios and enable broader interoperability [26].

Understanding that some of these aspects are inherent to the approach followed in this solution is crucial for realizing its potential benefits and drawbacks. However, others are imposed by the limited time to develop the POC, and additional resources can trivially overcome them.

V. Literature Review

It is worth mentioning that there are currently no solutions tackling the enhancement of semantic BI other than the one presented in this paper. Still, there has been an attempt to introduce an EA reference model specific for blockchain systems, although with no demonstrated practical use so far [27]. On the other hand, transaction validation which is also a focal part of the work presented in this paper has been explored by various tools, namely Bungee [28]. Furthermore, ontologies have been extensively utilized for the semantic integration of supply chain processes [29].

Thorough research into the state-of-the-art and technical literature has led to the compilation of some empirical observations about the field of interoperability in blockchain systems. The following subsections categorize them into paradigms, trends, and barriers to achieving interoperability.

A. Paradigms

There are several approaches to designing interoperability solutions for blockchain systems. The most prominent ones

are implementing interoperability as a cross-cutting component [30], as a background layer [31], or by following an interoperability framework [32].

The first approach is to have integrated public service governance. Achieving this requires holistic management of interoperability activities across administrative levels and sectors, setting processes to select relevant standards and specifications, evaluating them, and monitoring their implementation and compliance. Also, using a structured, transparent, objective, and a common approach to assessing and selecting standards and specifications and consulting relevant catalogs of standards, specifications, and guidelines when procuring and developing information and communication technology (ICT) solutions is of high importance. Lastly, one should actively participate in standardization work according to needs to ensure that all requirements go through.

The second strategy has to do with interoperability governance. Achieving this requires instituting the necessary governance structure, that is, the decision-making process, requirements, change management, and recovery plans, and establishing interoperability in all layers, complemented by operational agreements and change management procedures.

For the third approach, one has to ask the 4W and H questions. Firstly, "what" data objects or assets does the solution need to handle, and "who" controls the cross-chain transaction process and thus accounts for trust establishment. The "where" refers to the source and target ledgers. The "when" aims to understand the processes executing at design time or runtime. Finally, the last question is "how" do the underlying DLTs implement the cross-chain transaction and the testing approach for said implementation.

B. Trends

Presently, there are unique conditions to implement a robust solution for interoperability. The technology is mature enough to deal with many different implementations, incompatible abstractions, and the heterogeneity of stacks. Also, experimentation creates opportunities for gradually adopting and expanding network boundaries [33].

Consequently, researchers have shifted their focus to topics such as the ability to move an asset from chain A to chain B and the atomic exchange of assets from chain A to another chain B. Also interesting are the cases of asset encumbrance to lock and unlock an asset in chain A depending on an event of chain B. Another area is that of general cross-chain contracts to be able to use data from chain A in operations conducted in chain B [34].

When all these techniques become refined, it will finally be possible to think about data silo removal for the seamless flow of value among participants and network scale and growth while preserving trust and security. Furthermore, orchestrating complex cross-network business functionality will become a reality, leading to a significant increase in market size, liquidity, and efficiency of blockchain applications.

C. Barriers

Despite promising breakthroughs from projects in this field, there are still some open issues to overcome. The gap between theory and practice, including the lack of standardization, and network discoverability, are among the most challenging. Additionally, one must not forget the ever-important issue of privacy and security that the extensive literature repeatedly mentions as a tradeoff of interoperability and the eventual need for complex governance tactics [35].

Especially concerning to our research is the topic of bidirectional communication across permissioned ledgers, that as described in [36], is being hampered by the lack of semantic interoperability.

VI. Conclusion

There are a plethora of solutions related to achieving interoperability in blockchain systems. Nevertheless, most of them do not focus on the semantic layer of interoperability. This paper explored this gap by implementing a solution that enhances interoperation among systems owned by members of a supply chain consortium, as those usually entail operations among different business domains, demonstrating the potential of this mechanism. The novel approach leverages EA modeling to provide a common ground for systems to achieve interoperability in the semantic layer. It consists of an ontologically guided business logic plugin that validates cross-chain transactions. The outcome of the evaluation is satisfactory, proving the added value of this approach. Although the POC is limited to a network of permissioned ledgers owned by a business consortium, the paradigm can be extended for a set of blockchain systems of indefinite size. Thus, this paper supports that EA techniques can aid in solving open issues in BI and, in this way, encourage further research in this area. As mentioned before, this work needs to be generalized for ease of extensibility to other use cases. For instance, applying this mechanism to enhance interoperability amongst public chains in fields other than supply chain management would be noteworthy. Another interesting future research path is to extend the Hermes fault-tolerant middleware and Hephaestus, solutions described in the preliminary section of this document. The goal would be to increase their robustness by leveraging the transaction validation capabilities offered by the proposed solution.

References

[1] M. E. Peck, "Blockchains: How they work and why they'll change the world," *IEEE Spectrum*, vol. 54, no. 10, pp. 26–35, 2017.

[2] S. Underwood, "Blockchain beyond Bitcoin," *Communications of the ACM*, vol. 59, no. 11, pp. 15–17, 2016.

[3] V. Kalogirou and Y. Charalabidis, "The european union landscape on interoperability standardisation: status of european and national interoperability frameworks," in *Enterprise Interoperability VIII: Smart Services and Business Impact of Enterprise Interoperability*. Springer, 2019, pp. 359–368.

[4] R. Belchior, A. Vasconcelos, S. Guerreiro, and M. Correia, "A survey on blockchain interoperability: Past, present, and future trends," *ACM Computing Surveys (CSUR)*, vol. 54, no. 8, pp. 1–41, 2021.

[5] T. Hardjono, A. Lipton, and A. Pentland, "Toward an interoperability architecture for blockchain autonomous systems," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1298–1309, 2019.

[6] G. Antunes, A. Caetano, M. Bakhshandeh, R. Mayer, and J. Borbinha, "Using ontologies for enterprise architecture model alignment," in *Proceedings of the 4th Workshop on Business and IT Alignment (BITA 2013). Poznan, Poland*, 2013.

[7] H. Montgomery, H. Borne-Pons, J. Hamilton, M. Bowman, P. Somogyvari, S. Fujimoto, T. Takeuchi, T. Kuhrt, and R. Belchior, "Hyperledger cactus whitepaper," 2020.

[8] R. Belchior, A. Vasconcelos, M. Correia, and T. Hardjono, "Hermes: Fault-tolerant middleware for blockchain interoperability," *Future Generation Computer Systems*, vol. 129, pp. 236–251, 2022.

[9] C. Dannen, *Introducing Ethereum and solidity*. Springer, 2017, vol. 1.

[10] K. Sekniqi, D. Laine, S. Buttolph, and E. Sirer, "Avalanche platform," 2020.

[11] J. Kwon and E. Buchman, "Cosmos whitepaper," *A Netw. Distrib. Ledgers*, p. 27, 2019.

[12] R. G. Brown, "The corda platform: An introduction," *Retrieved*, vol. 27, p. 2018, 2018.

[13] R. Belchior, L. Riley, T. Hardjono, A. Vasconcelos, and M. Correia, "Do you need a distributed ledger technology interoperability solution?" *Distributed Ledger Technologies: Research and Practice*, vol. 2, no. 1, pp. 1–37, 2023.

[14] S. A. White, "Introduction to bpmn," *Ibm Cooperation*, vol. 2, no. 0, p. 0, 2004.

[15] A. Josey, M. Lankhorst, I. Band, H. Jonkers, and D. Quartel, "An introduction to the archimate® 3.0 specification," *White Paper from The Open Group*, 2016.

[16] S. Decker, S. Melnik, F. Van Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, and I. Horrocks, "The semantic web: The roles of xml and rdf," *IEEE Internet computing*, vol. 4, no. 5, pp. 63–73, 2000.

[17] R. Bansal and S. Chawla, "An approach for semantic information retrieval from ontology in computer science domain," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 4, no. 2, pp. 58–65, 2014.

[18] E. Commission, "Eira support series - how eira supports interoperability v1.0.0," 2015.

[19] R. Taelman, J. Van Herwegen, M. Vander Sande, and R. Verborgh, "Comunica: a modular sparql query engine for the web," in *The Semantic Web–ISWC 2018: 17th International Semantic Web Conference, Monterey, CA, USA, October 8–12, 2018, Proceedings, Part II 17*. Springer, 2018, pp. 239–255.

[20] C. Fan, C. Lin, H. Khazaei, and P. Musilek, "Performance analysis of hyperledger besu in private blockchain," in *2022 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*. IEEE, 2022, pp. 64–73.

[21] C. Hounsom, "Quorum whitepaper v0.2," 2018.

[22] 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.

[23] I. Docker, "Docker," *línea].[Junio de 2017]. Disponible en: https://www. docker. com/what-docker*, 2020.

[24] Y. Hu, Y. Xiong, W. Huang, and X. Bao, "Keychain: blockchain-based key distribution," in *2018 4th International Conference on Big Data Computing and Communications (BIGCOM)*. IEEE, 2018, pp. 126–131.

[25] W. Zhang, Y. Yuan, Y. Hu, K. Nandakumar, A. Chopra, S. Sim, and A. De Caro, "Blockchain-based distributed compliance in multinational corporations' cross-border intercompany transactions: A new model for distributed compliance across subsidiaries in different jurisdictions," in *Advances in Information and Communication Networks: Proceedings of the 2018 Future of Information and Communication Conference (FICC), Vol. 2*. Springer, 2019, pp. 304–320.

[26] E. Syriani, L. Luhunu, and H. Sahraoui, "Systematic mapping study of template-based code generation," *Computer Languages, Systems & Structures*, vol. 52, pp. 43–62, 2018.

[27] T. Erdenebold, J. J. Rho, and Y. M. Hwang, "Blockchain reference model and use case for supply chains within enterprise architecture," *Journal of Information Technology and Architecture*, vol. 16, no. 1, pp. 1–10, 2019.

[28] R. Belchior, L. Torres, J. Pfannschmid, A. Vasconcelos, and M. Correia, "Can we share the same perspective? blockchain interoperability with views," 2022.

[29] N. Petersen, "Towards semantic integration of supply chain and production data," Ph.D. dissertation, Universitäts-und Landesbibliothek Bonn, 2020.

[30] G. Wood, "Polkadot: Vision for a heterogeneous multi-chain framework," *White paper*, vol. 21, no. 2327, p. 4662, 2016.

[31] Z. Chen, Y. Zhuo, Z.-B. Duan, and H. Kai, "Inter-blockchain communication," *DEStech Transactions on Computer Science and Engineering*, 2017.

[32] A. Kouroubali and D. G. Katehakis, "The new european interoperability framework as a facilitator of digital transformation for citizen empowerment," *Journal of biomedical informatics*, vol. 94, p. 103166, 2019.

[33] R. Pandit, "Blockchain interoperability: Challenges, ongoing efforts, and potential solutions," 2021.

[34] V. Buterin, "Chain interoperability," *R3 Research Paper*, vol. 9, 2016.

[35] E. Abebe, Y. Hu, A. Irvin, D. Karunamoorthy, V. Pandit, V. Ramakrishna, and J. Yu, "Verifiable observation of permissioned ledgers," in *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2021, pp. 1–9.

[36] S. Nazarov, P. Shukla, A. Erwin, and A. Rajput, "Bridging the governance gap: Interoperability for blockchain and legacy systems," in *World Economic Forum whitepaper*, 2020.