An Architecture Proposal to Provide Interoperability Between DLT Platforms and Legacy Systems in the Financial Ecosystem

Fabricio Reis Furtado[1]
Alex Costa[2]
Courtnay Guimarães[3]

Abstract

Since the Bitcoin launch, the interest in Distributed Ledger Technologies has grown over the years and its potential was soon perceived for several applications. More recently, Smart Contracts and Decentralized Finance Protocols emergence made room for new business models, in order to provide more efficient settlement solutions and a better experience for people through Digital Assets. The Central Banks and Financial Regulators have been connected to this new context of the token economy, introducing their CBDC's initiatives and proposing regulatory frameworks and models to provide security in the crypto environment. The adoption of these new technologies in a financial environment brings several challenges, due to the incompatibility of standards used in the current centralized systems, demanding a great technical strategy to provide interoperability and efficiency for the operations. Considering this scenario, this paper proposes an architecture model to connect legacy systems to the DLT platforms, describing the necessary components and introducing a dedicated layer to abstract the network's characteristics, providing control, security and being agnostic and transparent to the current channels and financial services.

Palavras-chave: DLT*; architecture; interoperability; legacy systems.*

1 Grupo multidisciplinar de Criptoeconomia. Confederacação das Cooperativas do Sicredi – Porto Alegre, RS – Brasil. fabricio furtado@sicredi.com.br.
2 Grupo multidisciplinar de Criptoeconomia. Confederação das Cooperativas do Sicredi – Porto Alegre, RS – Brasil. alexcosta@sicredi.com.br
3 Grupo multidisciplinar de Criptoeconomia. Confederação das Cooperativas do Sicredi – Porto Alegre, RS – Brasil. c.guimaraes.jr@avanade.com

1 Introdução

Despite some concepts beyond Distributed Ledger Technology being older, this approach gained notoriety in 2009 when blockchain technology was born into Bitcoin to support transactions with the digital currency, ensuring security in a decentralized way (Nakamoto, 2008). Blockchain is composed of a continuously growing distributed database. It contains a list of permanent records, which are cryptographically signed and shared by all participants in a P2P network (Chuen, 2015). In a business view, blockchain is considered a network for exchanging transactions, values, or assets, without the help of intermediaries such as banks, card operators, or government institutions (Mougayar, 2016). Blockchain is also defined as a public and distributed ledger, which records all operations in a sequence of blocks that are added linearly and in chronological order (Swan, 2015). Unlike pure blockchain, DLT finds many uses across different industries, like eHealth, finance, supply chain monitoring, and the Internet of Things (IoT), providing an immutable and commonly verifiable ledger for larger-scale and highly complex systems (Gorbunova *et al.*, 2022).

In the financial industry, DLT has great potential to be the technology that will transform the market in several ways. Some directions are: servicing the existing and potential client base both at the retail and institutional levels; improvement of internal processes that remain slow, expensive, and error-prone; and, finally, by tokenizing assets, creating new financial products, and expanding the market (Gorbunova *et al.*, 2022). At the same time, around 100 central banks around the world are researching and experimenting with Central Bank Digital Currency (CBDC) (CBDC Tracker, 2023). CBDC, as a proposed application of blockchain and DLT, has attracted much interest within the central banking community for its potential to address long-standing challenges such as financial inclusion, payment efficiency, and both payment system operational and cyber resilience (Gorbunova *et al.*, 2022).

Even recognizing the importance of DLT in driving a new level of IT decentralization and the introduction of new services and applications, there are no consistent standards among different technologies and platforms, impacting the global mass adoption (Lima, 2018). Therefore, standards and interoperability are becoming important topics that need to be addressed for the DLT industry ecosystem to survive. A common set of standards and the focus on interoperability and scalability are among several industry blockchain requirements that are essential for the survival and mass adoption of DLT as an enabler of Web 3.0 and the decentralized Internet (Lima, 2018).

In this context, this study proposes an architecture model to connect financial legacy systems to DLTs, to become the DLT platforms more suitable to support the businesses. The proposed model acts in an intermediate layer, which is coupled to the DLT's middleware. It is responsible for providing services and features for the channels and products in a transparent way. This layer is centralized and was developed to be compatible with the existent DLT solutions, making possible interactions with legacy systems easy. We summarize the contributions of the proposed model as follows:

- Development of an architecture model to connect legacy systems to DLT's;
- Focus on transparency and abstraction for current financial services, providing a standard layer to perform operations, independent of the DLT Platform or network;
- Enable integration speed and facility to maintain the services;

- Improvement of system management, monitoring, and availability, through a partial decentralized approach;

- Introduction a new approach as Hybrid Finance, connecting Decentralized Finance and Centralized Finance in the same model;

- Presents use cases in the Financial Industry mixing centralized and decentralized solutions.

This study is organized into 4 sections. In this introductory section, we made an initial approach to the topic and presented the research goals. Section 2 shows the method applied in this study to search the related works, listing the criteria for selecting them and highlighting the identified gaps. Section 3 presents the model, contemplating the system architecture and each detail of its components. Finally, section 4 provides the final considerations, presenting contributions and suggestions for future work.

2 Methodology

This section presents the research methodology applied in this study. In order to identify the main references to base this proposal, the search keywords were defined, delimiting the research to some major topics: DLT architecture, interoperability, legacy systems. This was the starting point for starting the research, however, as suggested in the study protocol developed by Petticrew and Roberts (2006), we identified and combined new terms to obtain the expected result. Petticrew and Roberts (2006) recommend the use of synonyms, acronyms, abbreviations and alternate spellings to be combined using Boolean operators, thereby helping to enrich the search results. The evolution of the configuration of the search keywords is presented in Table 1. The base keywords are derived from the search query. They represent the general scope of the search. Afterward, as previously mentioned and recommended by the method, some synonyms and related terms were identified, thus defining an expanded and more complete set of keywords. We have used the expanded keyword set in the first round of queries in the literature databases.

From the studies obtained in the first round were extracted other specific terms, which are considered as a new set of keywords that are used to broaden the scope of the research. Thus, these new terms were added to the expanded keyword set to generate a third set, identified as final keywords. The Table 2 presents the search strings used in each step. This set of keywords represents the full scope of the query in the literature databases. The result of this consultation was considered for further analysis in this study, as described below.

Table 1 – Keywords evolution

Step	Keywords
1	Base Keywords
2	Base Keywords + Expanded Keywords
3	Base Keywords + Expanded Keywords + Specifc Keywords

Table 2 – Set of keywords and their combinations

Step	Keywords
1	$DLT \wedge (Architecture \vee LegacySystems)$
2	$(DLT \vee Blockchain \vee Cryptocurrencies \vee SmartContracts) \wedge ((Architecture \vee ArchitecturalAssessment \vee ArchitecturalAnalysis) \vee (Eficciency \vee Interoperability \vee Scalability))$
3	$(Blockchain \vee DistributedDatabase \vee Cryptocurrencies \vee SmartContracts \vee Bitcoin \vee Ethereum \vee HyperLedger) \wedge ((Architecture \vee ArchitecturalAssessment \vee ArchitecturalAnalysis) \vee (Performance \vee Eficciency \vee Interoperability) \vee (Composability \vee DeFi \vee DigitalAssets \vee Tokenization))$

2.1 Article selection

Once we have obtained the articles through web mechanisms, those that were not relevant to this study were removed. A significant number of articles were not useful for this study, the main reason for this is that a keyword may have different meanings or it is used in studies that are not connected to our main purpose. The criteria for inclusion and exclusion of articles should be based on the research goal. They can be found in the literature, as in Kitchenham and Charters (2007) and Biolchini *et al.* (2005), or be defined by the researchers themselves. The following are the inclusion and exclusion criteria defined for this work:

1. Merge and Removal of Duplicate Articles: Studies of individual databases were grouped and duplicate files were deleted;

2. Title and Summary Review: The title and summary of all studies selected in step [1] have been revised; those that do not address technical aspects related to architecture and interoperability in DLT have been removed;

3. Full-Text Analysis: We have read the full text of the articles selected in step [2]. All of them have been analyzed, and only those that do not present contributions or proposals related to interoperability between DLT and Legacy Systems have been removed.

3 Architecture Model

3.1 Article selection

Recent studies have shown that, by 2025, blockchain applications will represent 10% of the global gross domestic product (Lima, 2018). Especially in the financial ecosystem, DLT has the potential to provide appropriate structure for CBDC initiatives and deliver decentralized finance services to the centralized and traditional financial system. From now on, there will be a strong need for end-to-end interoperability standards, allowing cross-chain interoperability among different enterprise-grade and public DLT systems, with various platforms interacting with each other to make the development of applications and use cases much easier (Lima, 2018).

This context brings a new paradigm for the industry in which all IT processes, applications, and network structure must be redesigned and rethought to address the decentralization and disintermediation requirements, moving from a traditional cloud-based centralized approach toward a decentralized and distributed Web 3.0 architecture (Lima, 2018). The work (Lima, 2018) classified DLT Standards and introduced the Vertical-Industry-Specific Standards that focus on building customized industry solutions derived from the generic framework standards and adding customized modules of enabling technologies to create a business-driven, use-case-specific standard for the vertical industry addressed.

In our case, the proposed model in this paper aims to suggest an architecture standard for the financial systems environment, providing new features and connections among the financial legacy systems to the public and private DLT platforms and networks. This approach also presents a new concept that we call Hybrid Finance (HiFi), where in the same ecosystem is possible to provide decentralized and centralized finance together, integrating all the systems and platforms in the same topology.

3.2 Architecture proposal

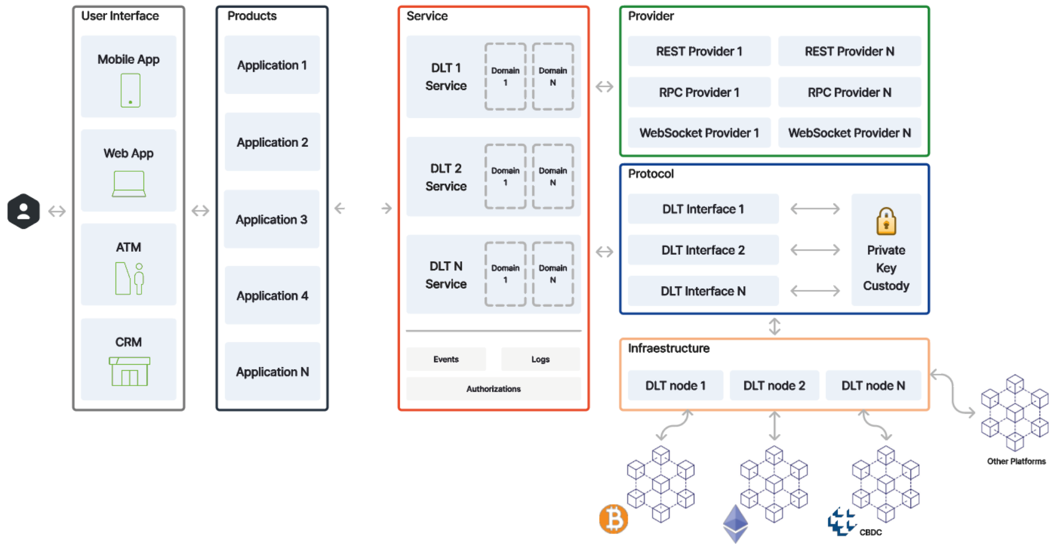
This architecture model suggests an intermediate layer to manage the communication between legacy systems and the DLT Platforms. It is a centralized component designed to be transparently coupled to the DLT platforms and provide a standard abstraction interface for financial products and services, allowing the only way to connect and perform transactions to more different DLT networks and platforms. The system architecture consists of defining the software components, their properties, and their relationships with internal and external elements. The drafting of the architecture design aims to facilitate communication between stakeholders, document the initial decisions about the model at a high level, and allow the reuse of the techniques and standards adopted in future projects.

The related works show that the introduction of a centralized component in the general architecture does not interfere with the main features provided by the DLT and has already been used in other studies and applications (Furtado *et al.*, 2020). This kind of approach also does not violate the DLT principles while maintaining the safety and decentralization characteristics in the conduct of business operations (Furtado *et a*l., 2020).

In Figure 1 the detailed architecture of the solution is presented, which contemplates the internal structure of each component and the connection between them. The communication among the elements occurs in a TCP/IP network using the REST API's and Data Distributed Streaming Components, thus maintaining compatibility and facilitating the coupling of the new layer. As shown in Figure 2, the introduced layer to

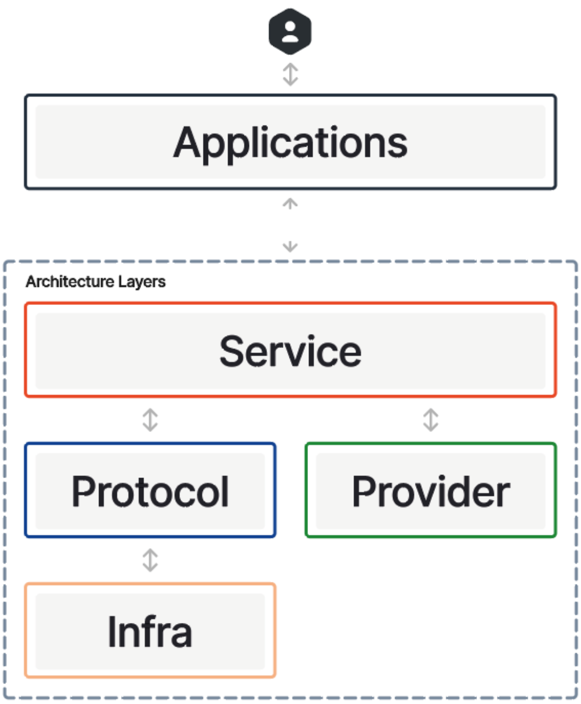
intermediate the communication with the DLT platforms has 4 sub-layers, called Service, Provider, Protocol, and Infrastructure.

Figure 1 - Detailed Solution Architecture



We designed the model so that the system continues to present itself to the external user as a single system and acts internally in the DLT network in a manner compatible with the topology and the available protocols. The solution contributes to interoperability, being easily adapted to new topologies, and easily customized for the interconnection of different DLT solutions. The following sections detail each sublayer of the proposed solution.

Figure 2 - Sub-layers and their relationship



3.2.1 DLT Service Layer

The Service layer offers for the products and services a communication interface with DLTs that abstracts the most of the complexities inherent to the interaction with this type of technology. One of the main objectives of this layer is to abstract the interaction com- plexity of products/services with DLTs. Through a setup, operations in DLT can be per-formed in a standardized way (including between different technologies, when possible),allowing interaction through a single communication interface. An example of a mes-sage format to be used for interaction with this layer, through a distributed data streaming component:

```
{
    "transactionId": "c5e2aafc-8376-4b0d-9248-ca5b3e8d8f36",
    "chain": "ETHEREUM_MAINNET",
    "action": "getBalance",
    "args": {
        "from": "0x1f9090aaE28b8a3dCeaDf281B0F12828e676c326"
    }
}
```

Attributes

- **chain:** represents the DLT that the product intends to execute the query/transaction. The Service layer will only allow connections to DLTs previously registered approved by the responsible team.
- **action:** represents the action to be performed on the DLT.
- **args:** represents the parameters needed to perform a certain action in the DLT. Each action will require zero or N parameters.

Features

- Traceability: The Service layer will also be responsible for logging messages that trace operations performed on the DLT. Thus, even though the product/service must record logs to track the execution of its operations, the Service layer will also maintain a second layer of logs. Each action provided by the Service layer may require zero or N attributes to be used for recording these actions. The product/service will need to pass these parameters, according to the needs of each action, in the context attribute, as shown below:

```
{
     "transactionId": "c5e2aafc-8376-4b0d-9248-ca5b3e8d8f36",
    "chain": "ETHEREUM_MAINNET",
    "action": "getBalance",
    "args": {
        "from": "0x1f9090aaE28b8a3dCeaDf281B0F12828e676c326"
    },
    "context": {
        "invoker": "SICREDI_X_MOBILE_WALLET",
        "walletId": "8bf63be3-7a20-4295-8a7f-527ccfd21b6f",
         "customerId": "05ab425a-a1f4-4f56-a1cb-af351464d40b"
    }
}
```

- Authorization Management: Certain DLT operations may require authorization to perform. For example: if we are moving balances from member wallets, a daily analysis of the allowed transfer limit may be necessary. This way, the Service layer will be able to publish an authorization request to execute a certain action. This request will be sent to the products/services through Data Streaming threads, and the response will need to be provided by them through the same channel. Whenever the action in DLT requires a product/service authorization, the Service layer will need to receive an attribute that represents the time limit for receiving the response. Actions that do not return authorization within the parameterized (set) time limit will be automatically canceled.

  Message to be sent to the Service layer that requires an authorized action:

  Upon receiving a message in this format, the Service layer will record the request in an internal database and send the following message to the products/services (original payload + dltAuthorizationId):


```
{
    "transactionId": "c5e2aafc-8376-4b0d-9248-ca5b3e8d8f36",
    "chain": "ETHEREUM_MAINNET",
    "action": "sendBalance",
    "args": {
        "from": "0x1f9090aaE28b8a3dCeaDf281B0F12828e676c326",
        "to": "0xddf4c5025d1a5742cf12f74eec246d4432c295e4",
        "value": 5
    },
    "authorizationTimeout": "2024-01-05 00:00:00.0000",
    "authorizationContext": {
        ... atributos extras que o produto/serviço possa precisar
        para validar e autorizar a transação
    }
}
```


  After evaluating the authorization request, the product should return a message in the format:


```
{
    "dltAuthorizationId": "a78ef277-d577-45cb-ae1e-a0ddd2b3b885",
    "transactionId": "c5e2aafc-8376-4b0d-9248-ca5b3e8d8f36",
    "chain": "ETHEREUM_MAINNET",
    "action": "sendBalance",
    "args": {
        "from": "0x1f9090aaE28b8a3dCeaDf281B0F12828e676c326",
        "to": "0xddf4c5025d1a5742cf12f74eec246d4432c295e4",
        "value": 5
    },
    "authorizationTimeout": "2024-01-05 00:00:00.0000",
    "authorizationContext": {
        ... atributos extras que o produto/serviço possa precisar
        para validar e autorizar a transação
    }
}
```

Once that authorization was received, if the 'action' attribute is marked as ACCEPTED and 'authorization Timeout' has not been reached, the query/transaction will be sent to the DLT.

```
{
    "dltAuthorizationId": "a78ef277-d577-45cb-ae1e-a0ddd2b3b885",
    "action": "ACCEPTED|UNACCEPTED"
}
```

- Event Tracking: The vast majority of transactions executed in DLTs are asynchronous, as many of them need to be mined, confirmed or validated by entities belonging to the network. In this way, actions such as creating a Smart Contract, transferring balances between wallets and updating data in a Smart Contract always need to be confirmed. Considering that it is not possible to define a minimum and maximum time to wait for these confirmations, each product/service that wants to transact data in DLTs would need to build its own logic to monitor these confirmations, creating monitoring processes for each scenario. In the same way, events occurring in DLTs can be a fundamental part of business logic implemented by products/services, such as tracking the first deposit of a given wallet, tracking asset movements in real time or tracking the redemption of funds linked to a Smart Contract. Seeking to avoid the repeated implementation of these controls in each product/service, as well as to make the process of monitoring actions performed in DLTs more efficient and secure, the Service layer will also offer tools for personalized monitoring of events.

  Monitoring of the following events in DLTs will be offered for any product/service:

  - Deposit identification in a wallet;
  - Redemption identification in a wallet;
  - Confirmation identification of a transfer of balances from one wallet to another;
  - Identification of confirmation of creation of a Smart Contract;
  - Identification of confirmation of alteration of a Smart Contract.

  As needed, other events will be implemented and offered to those interested in using them. To request the tracking of an event, the product/service must publish a message in the Data Distributed Streaming Component as follows:

```
{
    "transactionId": "67ae13e0-add7-43a9-9630-97c61d8478dd",
    "chain": "ETHEREUM_MAINNET",
    "event": "walletDeposit",
    "args": {
        "from": "0x66DE98A9CB6C5502869d4eb10EC7D69E7336eA9a"
    },
    "until": "2025-01-15 00:00:00.0000",
    "notificationRule": "first|all"
}
```

Attributes

- **chain:** represents the DLT that the product/service intends to monitor.
- **event:** represents the event to be monitored. There will be a pre-defined list of events available, with the possibility of implementing new events if necessary.
- **args:** represents the parameters needed to run the follow-up. Each event will require zero or N parameters.
- **until:** represents the time limit for monitoring the event. Upon reaching the date in this field, the event tracking will be automatically removed.
- **notificationRule:** represents the rule for notification of this event. If the product/service wants to be notified only the first time this event occurs, it must be parameterized with ‘first’. If you want to be notified whenever this event occurs, until the date parameterized in the ‘until’ attribute, it must be parameterized with ’all’.

Once registered, the event monitoring system will track the DLTs automatically, and it is not necessary for the product/service to perform actions. When identified in the DLT, the event tracking system will notify the product/service via Data Distributed Streaming Component message below:

```
{
    "transactionId": "67ae13e0-add7-43a9-9630-97c61d8478dd",
    "datetime": "2023-03-18 14:16:27.4874",
    "chain": "ETHEREUM_MAINNET",
    "event": "walletDeposit",
    "args": {
        "from": "0x66DE98A9CB6C5502869d4eb10EC7D69E7336eA9a"
    }
}
```

3.2.2 DLT Provider Layer

The Provider layer aggregates partners that allow Sicredi to carry out (to perform) operations in DLTs without the need to install or build internal software or hardware compo nents. Examples of providers are:

- APIs that allow checking balances or carrying out (performing) transactions in DLTs;
- Systems that provide reports for auditing crypto movements;
- Exchanges that allow the purchase, sale or exchange of assets;
- Suppliers that allow access to self-managed DLT nodes by cloud platforms;
- Crypto as a service Hubs and API’s.

It will be considered a component of the Provider layer any resource/provider/software/service that can be accessed by the Service layer purely by cloud.

3.2.3 DLT Protocol Layer

The purpose of this layer is to concentrate all the software components that will need to be developed or used to connect the Service layer with the used DLT nodes. Each DLT provides different ways to connect and exchange data by software. Some provide REST APIs, others allow communication via WebSocket, some via RPC, as well as others require the construction of components developed in specific programming languages, using libraries provided especially for the respective DLT. All data necessary for connecting to the DLT will be provided by the Service layer, such as:

- node IP
- connection port
- Connection type (REST, GraphQL, RPC, WebSocket etc.)
- Authentication (if required)

Due to the ability to connect directly to DLT nodes, components of this layer can only be accessed by components of the Service layer. It is up to the Service layer to ensure that all transactions sent to this layer have been authorized and recorded in the audit logs, since the only objective of this layer is to convert generic actions into low-level commands that respect the particularities, technologies and restrictions of each DLT.

3.2.4 DLT infrastructure Layer

The infrastructure layer includes all DLT nodes, both public and private. A DLT node comprises one or more instances (Cloud or on-premises) that can only be accessed by Protocol layer components. DLT, in its essence, consists of a decentralized P2P architecture, which contains a series of interconnected computers, where each acts as a node in the network. In general, all nodes are considered hierarchically equal, having the same privileges and the same influence on the network. Thus, they can act as both clients and servers, sharing the same data and resources. To perform the operations, the nodes use mechanisms of consensus and trust in direct communication between them, without the intermediary of third parties. The nodes in the network exchange messages containing transactions and addresses of other nodes. The database is shared by all nodes in the P2P network, so when a new transaction occurs, the information in this message is propagated between all of them (Furtado et al., 2019).

4 Conclusion

It is noticeable that the DLT has potential to transform the traditional finance industry with its characteristics such as decentralization, persistence and audibility. The architecture model is proposed as an alternative to existing solutions, providing the stacking of services and transparently coupling to the DLT platforms, allowing them and client applications that integrate with them to remain unchanged in their configurations. The model introduces the concept of Hybrid Finance, presenting a management layer to intermediate transactions with DLT platforms and enabling different services and features at the application layer. The model is coupled in a transparent way to the DLT platforms, without generating friction for the end-users, code changes or configuration in the native applications. The solution also does not interfere with the distributed consensus functions or access permissions, maintaining the security and decentralization characteristics of the DLT networks. In the context, where the proposed model serve as a bridge to connect Centralized and Decentralized Finance, we can highlight some use cases to make tangible its applicability:

- Hybrid Payments: this consists of making a payment using a traditional instrument like a card or an instant payment application, withdrawing funds from a crypto wallet linked to a DLT platform. This approach requires interaction with legacy system services and DLT platforms simultaneously.
- Transfers: sending money from Checking or Saving Accounts to some DLT Platform, performing the appropriate conversions, and applying fees.
- Unique UX: to provide a single channel for users to manage the WEB3 accounts and traditional accounts, allowing them to move assets between them.

The framework allows the model to evolve and integrate with other solutions. In this way, as future work, we highlight some modules to be included in the solution:

- Caches: provide resources to keep queries in cache, avoiding overload of operations on the nodes.
- Homologated Smart Contracts Repositories: provide homologated smart contracts for quick use, offering pre-tested versions of code to deploy on DLTs.
- Transaction Vulnerability Scanners: scan transactions for vulnerabilities before they are sent to the DLT.
- Blacklists: provide blacklists that indicate nodes, wallets, and smart contracts classified as fraudulent or blocked for transaction

In addition, some possible future directions have also been proposed, exploring opportunities and use cases that have not yet been addressed in existing studies. DLT-based applications are emerging in the finance environment and other industries, and these new directions can inspire researchers to conduct more detailed investigations in the future in order to improve current gaps and consequently expand technology usage. Finally and more broadly, the obtained results aim to provide more understanding and confidence to increase adhesion to DLT platforms by financial institutions. In academia, it is expected that this paper content can support and encourage researchers to perform new studies in order to seek new solutions to the identified challenges.

References

BIOLCHINI, J.; NATALI, A. C. C.; MIAN, P. G.; TRAVASSOS, G H. **Systematic review in software engineering**. Rio de Janeiro: PESC, 2005. (Technical report).

CBDC Tracker. **CBDC tracker**. 2023. Available in: https://cbdctracker.org/.

CHUEN, D. Lee Kuo. **Handbook of Digital Currency**. Amsterdam: Elsevier, 2015.

FURTADO, Fabricio Reis; SILVA, Josué ; CAPPELARI, Márcio; CASTILHOS, Cláudio; RODRIGUES, Vinicius; COSTA, Cristiano André da; RIGHI, Rodrigo.Towards characterizing architecture and performance in blockchain: A survey. **International Journal of Blockchains and Cryptocurrencies**, v. 1, n. 21, 2019.

FURTADO Fabricio Reis; RIGHI, Rodrigo, COSTA, Cristiano André da; ROEHRS, Alex; SINGH, Madhusudan. L7sp: a layer seven service provider for private blockchain systems. **International Journal of Blockchains and Cryptocurrencies**, v. 1, n. 3, p. 236-272, 2020.

GORBUNOVA Maria; MASEK, Pavel; KOMAROV Mikhail; OMETOV, Aleksandr. Distributed ledger technology: State-of-the-art and current challenges. **Computer Science and Information Systems**, v. 19, 01, p. 65-85, 2022.

KITCHENHAM, B.; CHARTERS, S. **Guidelines for performing systematic literature reviews in software engineering**. UK: Keele University, 2007. (EBSE Technical Report).

LIMA, Claudio. Developing open and interoperable dltblockchain standards [standards]. **Computer**, v. 51, n. 11, p. 106-111, 2018.

MOUGAYAR, William. **The Business Blockchain:** promise, practice, and application of the next internet technology. New Jersey: John Wiley & Sons, 2016.

NAKAMOTO, Satoshi. Bitcoin: A Peer-to-Peer Electronic Cash System. **Bitcoin.Org**, p. 1-9, 2008.

PETTICREW Mark; ROBERTS, Helen. **Systematic reviews in the social sciences**. New Jersey: Blackwell Publishing, 2006.

SWAN, Melanie. **Blockchain Blueprint for a New Economy**. Sebastopol, CA: O'Reilly Media, Inc., 2015.