Comparative Analysis of Permissioned Blockchains: Cosmos, Hyperledger Fabric, Quorum, and XRPL

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

4th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

5th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

6th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—Blockchain-based platforms are being increasingly adopted for building more transparent systems. However, their success depends on addressing some essential aspects during the planning and development phases. Notably, the permissioned blockchain model encompasses extra layers of customization to the network, requiring greater care to avoid pitfalls regarding performance bottlenecks, programmability limitations, and lack of interoperability with other systems. Considering the large number of solutions available today for developing permissioned blockchains, choosing the one that better fits the requirements of the target scenario is often a challenging task. Aiming to assist researchers, developers, and other professionals in the decision process, this work presents a comparative analysis of prominent open-source permissioned blockchain platforms, highlighting the main characteristics and differences that may drive this choice. Our work focuses mainly on indicating (un)favorable scenarios for each solution, an aspect that, to the best of our knowledge, has not been extensively explored in previous research. Namely, the blockchains hereby covered are Cosmos, Hyperledger Fabric, Quorum (Hyperledger Besu and GoQuorum), and XRP Ledger (XRPL). They are analyzed based on the characteristics of their consensus mechanisms, support for smart contracts, the capability of tokenizing tangible assets, interoperability with other systems, as well as their transaction throughput and latency.**

***Index Terms*—permissioned blockchain, comparative analysis, performance, interoperability, programmability**

I. Introduction

The National Institute of Standards and Technology (NIST) classifies blockchains into two groups [1]: (i) Non-permissioned (permissionless), a public and decentralized model where anyone can take part in the network and potentially publish a new block; and (ii) Permissioned, which encompasses private networks managed by a single organization, as well as consortiums managed by two or more organizations, having a limited number of identified participants subject to fine-grained access control. While many pioneer networks (e.g., Bitcoin [2]) follow the first approach, the permissioned model has been receiving much attention from industry and government entities due to its ability to bring more transparency to key sectors of the economy – e.g., for promoting the construction of Central Bank Digital Currencies (CBDCs) [3]. To address such needs, several blockchain frameworks supporting the permissioned model have been created in the last few years, becoming powerful allies in the process of building, implementing, and managing such systems.

To join a permissioned network, participants must be authorized by the organizations that manage it. The resulting systems are, thus, less prone to adversarial behavior that might sprout from anonymous users who could join and leave the network at any time. Consequently, permissioned solutions are usually able to adopt more efficient consensus mechanisms when compared with permissionless models. Still, there is a wide range of algorithms available, some requiring lower levels of trust among participants, at the expense of system performance in terms of throughput and latency. Thus, when designing a permissioned blockchain-based application, choosing an adequate consensus protocol is of utmost importance to reach an optimal balance between resilience to malicious behavior versus performance. Likewise, when deciding which platform to use, other aspects come into play. Examples include the programmability of intended features, including support for smart contracts and custom tokens, as well as the interoperability with other blockchains.

However, considering the abundant array of solutions, as well as the extra layers of complexity introduced by permissioned blockchains, choosing the one that better fits the requirements of the target scenario is often a challenging task. This work aims to address this concern, building a discussion that should assist researchers, developers, and other professionals during the process of deciding which framework to adopt in their projects. To achieve that, we establish and apply a set of criteria for the analysis of some of the most prominent open-source permissioned blockchains: Cosmos, Hyperledger Fabric, Quorum (Hyperledger besu and GoQuorum), and

XRP Ledger (XRPL). Besides comparing these technologies and considering their key features and benefits, we highlight (un)favorable scenarios for each solution, providing the reader with a clearer picture of when (not) to use each one of them.

The rest of this paper is organized as follows: Section II compiles the most relevant related works. Section III presents an overview of the compared blockchains. Section IV describes the characteristics used in the analysis of the blockchains. Section V, in turn, presents our results and comparisons. Finally, Section VI concludes the discussion and gives some directions for future work.

II. Related works

A literature review was conducted in order to identify works that aim to compare permissioned blockchains. To that end, we used Elsevier Scopus to obtain results from various databases (e.g., ACM, IEEE, and Springer), through the following search string applied to the title, abstract, and keywords of the works: *permissioned AND blockchain AND (comparison OR comparative OR survey)*. This search returned 164 documents, of which we analyzed the titles and abstracts. Subsequently, to account for potentially missing works in Scopus, we applied the same procedure for the first 100 results of Google Scholar, resulting in 13 works that were selected and read. Potentially interesting references cited in these documents were also analyzed. Finally, we selected 10 papers as our set of related works. A summary of them is represented in Table I, which shows the objective blockchain aspects that are analyzed on each, omitting characteristics that are presented subjectively or which are not properly explained. Also, some works are marked as experimental, meaning that the quantitative values they present (e.g., latency and throughput) were collected empirically instead of retrieved from other references.

Hamida et al. [4] show an overview of blockchain concepts, explaining aspects such as consensus algorithms, privacy, and scalability. Followingly, they present a table with the main characteristics of several blockchains. A comparative analysis is not provided.

Nadir [5] presents an overview of Hyperledger Fabric, Quorum, and Corda. Moreover, a comparative score, between 1 and 3, is given for each of them, for every analyzed metric. After summing these values, a final score is assigned to each framework. The actual values of throughput and latency are not explicitly presented.

Monrat et al. [6] deploy four blockchains and benchmark their latency and throughput. Likewise, Sedlmeir et al. [7] present their results based on the same metrics. The latter create and use an open-source framework meant to benchmark blockchains.

Dabbagh et al. [8] compile papers comparing the latency and throughput of blockchains. The environment and network setups used in the benchmarks are presented in tables, alongside the results. An interesting contribution is that they highlight experimental inconsistencies in some of the projects (e.g., overlooking the relationship between efficiency and the number of nodes, or neglecting the impact of consensus on the performance results), which can also be observed in some of our related works.

Polge et al. [9] briefly explain the analyzed blockchains and present a table comparing their characteristics. They also quantify the analyzed aspects on a scale from 0 to 5, but the actual values for scalability, latency, and throughput are not

TABLE I
Summary of related works that compare permissioned blockchains

Year	Permissioned blockchains	Presented aspects	Experimental performance evaluation	Reference
2017	ChainCore, Corda, Fabric, Quorum, OpenChain, Monax	Consensus, Privacy mechanisms, Smart contracts, Throughput	No	[4]
2019	Corda, Fabric, Quorum	Consensus, Privacy mechanisms, Smart contracts Throughput, Tokens	No	[5]
2020	Corda (4.3, 4.4, 4.5), PoA Ethereum (Parity), Fabric, Quorum	Consensus, Latency, Smart contracts, Throughput, Tokens	Yes	[6]
2021	PoA Ethereum (Geth 1.9.8 and Parity 2.5.1), Fabric 1.4.4, Indy 1.12, Quorum 2.3, Sawtooth 1.2	Consensus, Latency, Smart contracts, Throughput	Yes	[7]
2021	PoA Ethereum (Parity and Geth), Fabric (0.6, 1.0, 1.4.1), Libra, Quorum	Consensus, Latency, Throughput	No	[8]
2021	Corda, Fabric, Multichain, Quorum	Adoption, Consensus, Latency, Privacy mechanisms, Smart contracts, Throughput, Tokens	No	[9]
2021	BigchainDB, BlockchainDB, Blockchain Relational Database, ChainifyDB, Fabric, FalconDB	Consensus, Database type, Replication model, Smart contracts, Transaction processing, Throughput	No	[10]
2021	Fabric, Sawtooth	Consensus, Throughput, Transaction processing	Yes	[11]
2023	Corda, PoA Ethereum, Fabric, Quorum	Consensus, Latency, Throughput	Yes	[12]
2023	Fabric 2.2, Quorum 21.1 (GoQuorum and Besu), Sawtooth 1.2	Consensus, Latency, Privacy mechanisms, Throughput	Yes	[13]

presented. Moreover, an in-depth analytical comparison is not provided.

Fekete et al. [10] present a thorough overview of six permissioned blockchains and of LedgerDB, a centralized ledger database. The description of the technologies as well as the comparative analysis focus mainly on the transaction flow on each blockchain, providing an unusual, yet important, view of the evaluated solutions. Using a similar approach, Thomaz et al. [11] benchmark Hyperledger Sawtooth and Hyperledger Fabric. For the former solution, scenarios with either parallel or sequential transactions are proposed and compared. For Hyperledger Fabric, scenarios with non-conflicting and conflicting transactions are tested, showing that conflicting transactions might be a problem on blockchains that execute transactions before ordering them.

Gupta et al. [12] compare the general scalability of Corda versions 4.3 and 4.5. The same is done for Ethereum using Proof-of-Authority (PoA) and Proof-of-Work (PoW). It concludes that PoA Fabric has better scalability, albeit quantitative data and an in-depth comparative analysis are not provided.

Finally, Copocasale et al. [13] define several blockchain aspects and relate them to Hyperledger Fabric, Hyperledger Sawtooth, and Quorum, based on information from the literature. Moreover, they set up similar networks using each of the frameworks and compare the throughput of each. Their results are then contrasted with related works.

A. Research gap

By analyzing Table I, it becomes clear that certain platforms have been predominant throughout the years. Specifically, Hyperledger Fabric is present in all of the related works, Quorum appears in 80% of them, Corda in 50%, and PoA Ethereum in 40%. Although this is a good indication of their maturity and evolution, it also implies a certain bias to this kind of work. In real-world applications, there are other strongly established and widely used solutions, and thus they should be benchmarked alongside the frameworks presented in the related works.

It is also important to note that the motivation of most of the works is not to provide useful information for those who want to decide which blockchain to use. As a consequence, none of them provide all of the following: a) A comprehensible set of metrics (i.e., all or most of the objective aspects from the related works, as shown in Table I); b) An analytical comparison of the results; c) Favorable and unfavorable scenarios for each framework; d) Proper definition of the presented metrics and aspects; e) Actual values for the quantitative metrics; and f) The version of the analyzed frameworks. These pieces of information are essential to properly aid the reader who seeks guidance during the process of deciding which framework to use in their project.

This work comprises all of the aforementioned items not only for some of the most commonly analyzed frameworks in this kind of paper, but also for other popular choices. Therefore, it considers the relevant and extensively studied models from previous research, as well as the most important criteria for evaluating them. As a result, an updated and embracing collection of information, discussions, and use cases is presented, specifically for projects needing help deciding which technology to adopt.

III. Analyzed blockchains

In this paper we analyze five open-source permissioned blockchains (referred to in this work as *frameworks*): Hyperledger Fabric and Quorum (both Besu and GoQuorum) since they are the most present in the related works; and, aiming to benchmark popular alternatives, XRPL and Cosmos, whose cryptocurrencies figure as the 5th and 20th highest market capitalizations, at the time of writing [14]. Also, their Github activity stats surpass those of any other top 20 permissioned blockchains, with 12,000 and 13,000 commits, as well as 4,400 and 5,600 stars, respectively [15], [16].

A. Cosmos

Cosmos calls itself the "Internet of blockchains" because it aims to interconnect multiple networks. It is a comprehensive ecosystem with protocols, an SDK (Software Development Kit), tokens, wallets, applications, and services. It promotes a network of independent blockchains that are interconnected by the IBC (Inter-Blockchain Communication), allowing communication and token transaction between them without the need for exchanges. The SDK enables the development of a blockchain with custom configuration and connection to other networks through IBC. Binance Coin (BNB), for instance, is a token developed using the Cosmos ecosystem. When building a new network, Tendermint Core is used by default, a consensus engine that replicates the developed application on different machines and implements Tendermint, a consensus algorithm based on Proof-of-Stake (PoS). Furthermore, through the concept of modules (i.e., Go programs) , it is possible to customize the blockchain, its logic, rules, and properties, including the consensus mechanism.

The IBC is an interoperability protocol which is also considered a module. It can be added to the developed network, allowing its connection to the rest of the Cosmos network or only part of it, as desired. Furthermore, any network that uses a fast-finality consensus (i.e., does not depend on the blockchain's length to probabilistically assume a past block as part of the actual chaining) can connect to Cosmos via IBC [17]. Otherwise, to connect probabilistic-finality chains (e.g., networks using Proof-of-Work (PoW)), cross-chain bridges need to be employed through smart contracts that exchange assets between Cosmos and these networks.

B. Hyperledger Fabric

Hyperledger Fabric has a modular blockchain architecture that enables the connection of distinct components, such as consensus mechanisms and identity services, allowing the creation of highly customizable private and consortium networks. The nodes are maintained by one or more organizations, which can be independent networks representing and managed by anything from individuals to big corporations. When several

organizations exist, a consortium network is formed. The communication between them is established through channels, which are subnetworks where authenticated and authorized nodes can privately communicate. In that manner, it is even possible to create more than one consortium, by setting up several channels between specific organizations.

Regarding the consensus mechanism, Hyperledger Fabric currently only supports Raft, but it is going to introduce SmartBFT on its next major release (version 3.0). It is important to mention that Raft is a crash fault tolerance (CFT) algorithm, and thus, it is not tolerant to Byzantine faults (i.e., not resistant to malicious nodes), therefore it should be avoided when the participants of the network are not trusted. In case an alternative protocol is desired, Hyperledger Fabric allows the development and use of custom consensus mechanisms, even though this process might be laborious.

C. Quorum

Quorum is an Ethereum-based distributed ledger protocol that enables the creation of permissioned blockchain networks with support for private transactions [18]. Originally developed and maintained by JPMorgan Chase, it has two main implementations: Hyperledger Besu and ConsenSys GoQuorum. The main difference between them is that the former is capable of connecting to the Ethereum mainnet, while the second is not. Additionally, Besu counts with consensus algorithms based on Proof-of-Authority (PoA), PoW, and PoS, whilst GoQuorum supports PoA-based protocols and Raft. In both implementations, it is possible to perform access control of validator nodes and accounts, and also configure who can propose transactions to the network.

Quorum is built on top of the Go implementation of Ethereum (geth), and consists of two main services: the Quorum Client and the Privacy Manager [19]. The first runs the consensus algorithm and executes the Ethereum Peer-to-Peer (P2P) protocol, only accepting connections from participants of the permissioned network. The Privacy Manager, on the other hand, enables smart contract operations and private transactions, by encrypting the payload of these messages. In this manner, unauthorized participants are not capable of reading encrypted transactions not meant for them, whilst authorized peers can execute these transactions and update their databases accordingly.

D. XRP Ledger

XRP Ledger (XRPL) is Ripple's blockchain. It employs a voting-based consensus mechanism created by them, in which each validator needs to maintain its own list of trusted nodes. As time passes, the nodes propose sets of transactions to form the blocks. At a given moment, if the network does not agree on the next block, the validators adapt their proposals by adding or removing transactions according to the transactions that are most or least present in the proposals of their trusted nodes. This process may repeat for several rounds, until 80% of the trusted nodes agree on the proposed set of transactions. This whole process takes place within seconds, allowing up to 20 blocks to be created per minute.

XRPL encompasses characteristics of permissioned and permissionless networks at once [20]. Whilst a mainnet exists, where any computer can join as a validator, it is also possible to develop sidechains. These are permissioned or permissionless independent networks containing their own tokens and transaction rules. They may as well utilize custom consensus algorithms. Moreover, sidechains are capable of moving assets back and forth with the mainnet, allowing their tokens to be traded through the Decentralized Exchange (DEX) on the mainnet.

IV. Evaluation criteria

Use cases for permissioned blockchains are broad, and applications in the financial, logistics, mobility, and energy sectors are already a reality [4], [21]. The technology is applied in asset management, supply chain tracking, and process automation, for instance. This diversity of applications has led to the development of increasingly specialized platforms with characteristics. As a result, many use cases can be addressed by more than one framework, making the process of comparing and choosing the blockchain to be adopted a complex one. Therefore, for this work, it is important to define comparison criteria relevant to most of the use cases to which permissioned models are applied. This set of criteria comprehends the key aspects presented in the related works (Table I), focusing on performance, interoperability, and programmability. However, some of them, such as privacy mechanisms, are not included here due to the space constraints of the paper.

In this section, the selected criteria are explained and justified. To this end, we highlight important considerations that projects should be cognizant of when choosing which framework to adopt. Specific scenarios that are (un)favorable for each framework are then presented in Section V, through the lens of these criteria.

A. Consensus mechanism

Consensus mechanisms are executed by the participants of the network, so they can agree on the data that is added to the chain. In permissioned blockchains, even though the nodes are not necessarily trusted by one another, at least they are known and authorized to participate in the system. Consequently, heavier algorithms based on competition and reward (e.g., PoW and PoS), usually are not necessary. Instead, Byzantine Fault Tolerant (BFT) mechanisms can be employed when the participants do not trust each other, and Crash Fault Tolerant (CFT) protocols are recommended when the network nodes trust one another. Popular examples are Proof-of-Authority (PoA), which is BFT, and Raft, which is CFT.

Byzantine faults are caused by the erratic behavior of participating nodes, occurring as a consequence of a compromised node contributing to ambiguous responses or deceiving other agents [22]. In general, BFT algorithms allow the network to continue operating correctly even if some of its nodes exhibit arbitrary or malicious behavior [23]. The computational cost

of this model (i.e., the number of nodes required to maintain the network functioning properly) is generally given by *3f + 1*, where *f* is the number of nodes with failures or malicious intent. CFT mechanisms, on the other hand, can maintain the correct operation of the network in case of failures of some of its nodes, but, unlike BFT, they do not offer protection against malicious behavior. While this characteristic may be a disadvantage in some scenarios, its lower computational cost (i.e., *2f + 1*) can be advantageous in others. Thus, the choice of the most appropriate mechanism should be directly related to the business model.

B. Performance (throughput and latency)

Transaction throughput is defined as the number of transactions consolidated in a certain period, and transaction latency is the elapsed time from the moment the system receives a transaction proposal until it becomes consolidated [13]. Both metrics are heavily influenced by the consensus algorithm used by the network. Since permissioned blockchains usually employ lightweight mechanisms, these values tend to be high. However, the implementation of the consensus can impact the actual performance of the system. For this reason, it is important to compare the throughput and the latency of each framework in numbers. These metrics are especially important for projects seeking large volumes and high speed of transactions.

C. Smart contracts

Smart contracts are self-executing contracts that contain the terms of an agreement between parties. They are written in the form of code that allows transactions to be automatically carried out between untrusted entities, without the need for a central authority. Support for smart contracts in permissioned networks provides flexibility to the system by enabling the development of distributed applications with well-defined behavior.

Unlike ”traditional” software, built with general-purpose programming languages (e.g., Java and Python), most smart contracts are written in less mature programming languages, created specifically for developing this kind of program. Consequently, many challenges are posed: a lack of general-purpose libraries, a deficiency of debugging features, and limited standardization [24]. Thus, when choosing which framework to adopt in a project, the maturity of the supported languages needs to be assessed and its weaknesses must be known upfront.

Moreover, when working with a framework that supports smart contracts in multiple programming languages, knowing which one is more efficient is also advisable. This can be particularly relevant when seeking high throughput and low latency of the transactions, given that more performant languages are more prone to provide that [13] [25].

Finally, the prior knowledge of the programmers involved in the project should also be taken into account. Some languages have steep learning curves, so if time is a key factor, prioritizing frameworks that use a language that is mastered by the developers is highly recommended.

D. Tokens

Tokenization is defined as the process of translating a physical asset into a digital representation of it, known as a token [26]. In this work, we analyze if the studied frameworks support the creation of these assets in the form of Non-Fungible Tokens (NFTs). The capability of minting Fungible Tokens (FTs), such as cryptocurrencies, is also assessed. This feature is desirable in most projects, since it enables the establishment of an economy for the application. Even if that is not the case, the creation of custom tokens allows further adaption of the system to the business logic, enabling the definition of data structures that are stored in the blockchain.

E. Interoperability mechanisms

Interoperability is defined as the capability of transferring assets among distinct blockchains [27]. In the context of this work, interoperability mechanisms are those enabling the exchange of tokens among blockchains that are created by distinct frameworks or belong to distinct ecosystems. This capability might be desirable, for instance, in scenarios where it is important to incorporate tokens from other blockchains (permissioned or not). Moreover, sending tokens to popular networks like Ethereum might facilitate exchanging tokens for fiat money.

V. Comparative analysis

The information collected to support our comparative analysis was obtained through the study of the selected framework, involving the reading of official documentation and related work.

A. General aspects

The general aspects of the compared frameworks are compiled in Table II.

In the context of permissioned networks, the consensus mechanism is executed by a restricted set of nodes. When they are trusted, it is possible to employ CFT consensus, which tends to require fewer computational resources, as discussed in Section IV-A. Otherwise, when adversarial behavior is more prone to happen, BFT algorithms are more suitable. It is noteworthy that the consensus mechanisms used by the analyzed blockchains are mostly BFT, with GoQuorum and Hyperledger Fabric being the only ones that, by default, offer the possibility of using a CFT mechanism (i.e., Raft). Also, Besu supports PoS and PoW-based algorithms, which are rarely used in permissioned blockchains. Ripple, on the other hand, only offers a BFT mechanism called XRP Ledger Consensus Protocol. This mechanism is based on voting and on lists of trusted nodes, maintained by the validators, as explained in Section III-D Finally, Cosmos adopts Tendermint by default, a BFT mechanism based on two rounds of voting, and capable of tolerating up to 1/3 of the network acting maliciously [28].

TABLE II
GENERAL ASPECTS OF THE ANALYZED PERMISSIONED BLOCKCHAINS

Blockchain	Consensus mechanism	Smart contracts	Custom tokens (FTs and NFTs)	Interoperability mechanisms
Cosmos	Tendermint (BFT)	Agoric / Javascript, Ethermint / Solidity, Wasm / Rust	Yes	IBFT, Ethereum,ERC tokens
Hyperledger Fabric	RAFT (CFT), SmartBFT (only Fabric 3.0)	Go, Java, Node.js	Yes	ERC tokens
Hyperledger Besu	PoA-based (QBFT, IBFT 2.0, Clique), PoS, PoW-based (Ethash)	Solidity, Vyper	Yes	Ethereum, ERC tokens
GoQuorum	PoA-based (QBFT, IBT, Clique) Raft	Solidity, Vyper	Yes	ERC tokens
XRPL	Voting-based BFT algorithm	Partially supported	Yes	————

The support for smart contracts constitutes the second comparison criterion in Table II. Cosmos, Fabric, and Quorum (both Besu and GoQuorum) employ smart contracts, making them versatile frameworks that apply to a wide variety of use cases beyond asset tokenization, such as in healthcare, IoT, and Supply Chain [29]. Ripple, on the other hand, does not offer native support for this feature. Instead, it counts with escrows, which allow conditional payments and are less versatile than actual smart contracts [30]. Also, there is the possibility to use hooks to add logic before and after transactions, but only in the testnet [31]. Alternatively, to use smart contract functionalities in the mainnet, it is necessary to use third-party solutions like Evernode and Xahau, which are sidechains built on XRPL [32], [33].

Quorum being a fork of Ethereum inherits many of its features, such as smart contracts in Solidity and Vyper. In contrast, Fabric and Cosmos support general-purpose languages, such as Go and JavaScript, which might be preferable, depending on the skill set of the programmers of a given project. In any case, it is important to be aware that programming languages do not perform equally. In a research project by Capocasale et al., Fabric transactions performed with Go smart contracts achieved better throughput than in Java, reaching from 15% to 235% higher TPS, depending on the experiment [13]. Thus, although Fabric supports three possible programming languages for smart contracts, Go should be the first choice if performance is a priority.

Another relevant criterion in choosing the most suitable blockchain model is the support for creating tokens, as well as providing resources for their transaction and management. In general, any platform that supports smart contracts can also support tokenization. All of the analyzed frameworks support the creation of both FTs and NFTs. In Quorum, it is easier to build applications using widely implemented open-source Ethereum guidelines, such as its Token Standards (e.g., ERC-20, ERC-721, and ERC-1155). Although in a more limited manner, Cosmos and Fabric also provide open-source templates of ERC-based smart contracts.

This standardization is an important step toward integrating blockchains (e.g., networks that adopt ERC-20 tokens). In this regard, Besu supports direct connection with public Ethereum networks, whilst in GoQuorum, Fabric, and XRPL, non-native cross-chain solutions would be necessary to that end. On the other hand, Cosmos presents outstanding interoperability features, allowing the interconnection of independent fast-finality blockchains (not only those in the context of Cosmos) through its protocol, Inter-Blockchain Communication (IBC). The framework also counts with the Gravity Bridge, which allows ERC-20 tokens to be transferred between Cosmos and Ethereum networks. Moreover, Cosmos uses the concept of Application-specific Blockchains, which are blockchains with their own rules and applications, linked and built with modules developed through the Cosmos SDK.

Finally, for some projects, it might be useful to know that Hyperledger Fabric provides outstanding flexibility for the participants of the networks, providing them with the choice of which type(s) of nodes they want to assume. After a transaction is submitted to the network, it is executed by the endorsing nodes, then the ordering nodes run the consensus mechanism to order the transactions and generate the blocks, which, finally, are validated by the peer nodes. It is relevant to highlight that this processing flow is quite uncommon, since the transactions are executed before being ordered (execute-order (XO) architecture), instead of being ordered before being executed (order-execute (OX) architecture). Consequently, the transactions can be executed in parallel, and, depending on the endorsing policy of the network, the minority of the nodes need to actually execute them, thus saving processing and enhancing latency and throughput. On the other hand, due to the XO architecture, Hyperledger Fabric is incapable of handling conflicting transactions (i.e., transactions that depend on the result of a previous one, such as those that have the same recipient address) [11]. In this scenario, the transaction fails and the client needs to resubmit it. As a result, the participants of the network need to spend computational resources again to process the request. For this reason, Hyperledger Fabric should be avoided in projects in which conflicting transactions are likely to be requested simultaneously.

B. Performance

The performance of the analyzed frameworks is presented in Fig. 1, which displays transaction latency in seconds, and transaction throughput in transactions per second (TPS). The values in the chart correspond to the highest values found (be it in the literature or in the official documentation of the frameworks) for Cosmos [34], [35], Hyperledger Fabric [36], Hyperledger Besu [37], GoQuorum [38], [39], and XRPL [40].

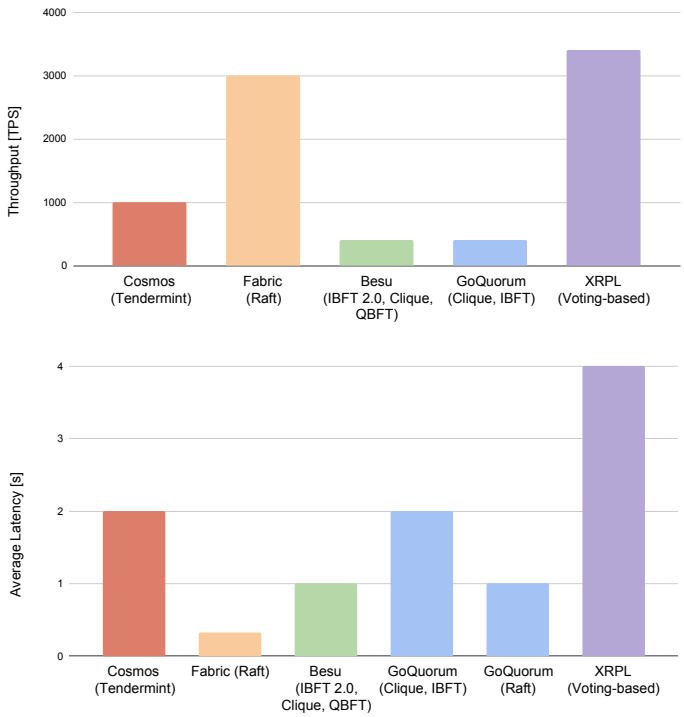

Fig. 1. Performance metrics for several combinations of frameworks and consensus mechanisms.

The results presented were acquired using estimates derived from the references. Nevertheless, these estimates are capable of conveying the expected behavior of each framework. This data represents the potential of each solution under optimal scenarios, i.e., few nodes in the network, non-adversarial behavior, no hardware bottlenecks, successful write transactions of small payloads, and no conflicting transactions. The version of the software used in these benchmarks were: Cosmos Gaia 7.0.3, Hyperledger Fabric 2.5, Besu v22.4, GoQuorum v2.6.0-updated_ibft, and XRPL Rippled 1.12.0. It is important to note that the throughput and latency values for some combinations of frameworks and consensus were not found.

In general, by executing a lightweight consensus mechanism on a limited set of nodes, permissioned blockchains can achieve fast convergence. This is particularly true for CFT algorithms. Thus, as expected, Hyperledger's implementation of Raft presents lower latency (0.33s) than the BFT algorithms, which, nonetheless, exhibit low latency.

Regarding throughput, every BFT algorithm implemented in Quorum reaches 400 TPS, whilst Cosmos' Tendermint can process up to 1,000 TPS. Raft's throughput (3,000 TPS) is higher than the others, except for XRPL's BFT mechanism, which, according to Ripple's official website, can achieve up to 3,400 TPS. However, it is important to note that the experimental procedure of this benchmark is not presented, and we could not find works in the literature that would provide us with this information.

C. Overview

To provide a more visual representation of the discussion, a radar chart is presented in Fig. 2. Thus, for each analyzed aspect, a score between 0 a 3 was assigned for each framework.

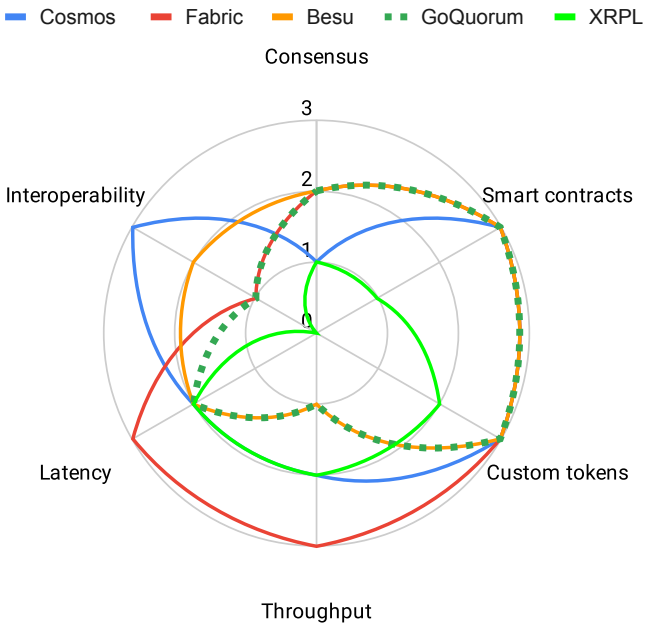

Fig. 2. Quantified comparison of the analyzed aspects.

The evaluation criteria are:

- **Consensus** - 1 point for each: CFT, BFT, others.
- **Smart contracts** - 1.5 points for each (since the last two criteria are mutually exclusive): Variety of languages, general-purpose languages, Ethereum-based
- **Custom tokens**: 1 point for each: FTs, NFTs, ERC tokens
- **Throughput**: Fabric has the maximum throughput proved experimentally (3). XRPL claims even higher values, but does not describe the process of benchmarking (2). Cosmos has intermediary throughput (2). Quorum has the lowest throughput (1).
- **Latency**: Fabric has remarkably low latency (3). The other frameworks have low latency (2).
- **Interoperability**: 1 point for each: IBC, native connection to Ethereum, ERC tokens.

Fig. 2 shows that the analyzed aspects of XRPL are not capable of competing with the aspects of the other frameworks. Thus, this solution is recommended mainly when integrating with Ripple's environment is a requirement.

Regarding Quorum, most aspects are similar in both implementations, but Besu is more flexible in terms of interoperability, given that it enables integration with public Ethereum networks. Thus, the only scenario where GoQuorum would be preferable would be if a CFT algorithm (Raft) is desired, given that this protocol is not supported in Besu.

In general terms, Hyperledger Fabric achieved the best scores for every aspect, apart from interoperability. So if this criterion is not essential, Fabric should be suitable for most scenarios. Otherwise, if integrating with other blockchains (beyond Ethereum) is a priority, then Cosmos is highly recommended.

VI. Considerations & Future works

Permissioned blockchains are widely employed in academic and enterprise projects due to their flexibility and capability of allowing the set of organizations that manage the network to define participants, permissions, and rules that fit their business logic. Several frameworks exist to aid developers in building this sort of system, however, their features and characteristics are heterogeneous, and, thus, the decision of which solution to adopt is not simple. Simplifying this process, our work compiles the most important aspects that should be accounted for during this process, presenting them for five of the most prominent frameworks in related works and in the market. For each of them, scores are assigned concerning the analyzed aspects.

The results show that XRP Ledger still lacks important features such as native smart contracts and a variety of consensus algorithms. Conversely, Quorum is capable of providing a wide range of consensus mechanisms, from CFT protocols, suitable for scenarios where the network nodes trust each other, to PoW-based algorithms, appropriate when there is no trust at all among the participants. On the other hand, Cosmos stands out for its interoperability mechanisms, which allow the exchange of assets with other fast-finality chains, as well as Ethereum. Finally, Hyperledger Fabric is the most resourceful of the analyzed solutions, presenting the highest scores for every aspect (consensus, smart contracts, custom tokens, latency, and throughput), except for interoperability.

Although Hyperledger Fabric could fit most scenarios where integrating with other chains is not a requirement, some considerations must be highlighted. First of all, although it supports smart contracts in multiple programming languages, using Go typically results in better transaction throughput and latency. Moreover, due to its execute-order processing architecture, Hyperledger Fabric should be avoided in projects where simultaneous conflicting transactions are likely to take place, given that they might fail.

In future work, we are going to add other criteria to the analysis of the frameworks, such as the supported privacy and security mechanisms (e.g., private transactions and Hardware Security Modules), the employed database type (e.g., relational and non-relational), and the available auxiliary tools (e.g., benchmarking tools and blockchain explorers). Moreover, we intend to measure both the latency and the throughput of Cosmos and XRPL, which, to the best of our knowledge, no work in the literature has performed. Finally, after the release of Hyperledger 3.0, we are going to analyze its performance when running SmartBFT, and compare the results to the Raft implementation.

References

[1] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Nistir 8202 - blockchain technology overview," National Institute of Standards & Technology - NIST, Tech. Rep., 2018.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2009. [Online]. Available: http://www.bitcoin.org/bitcoin.pdf

[3] T. Zhang and Z. Huang, "Blockchain and central bank digital currency," *ICT Express*, vol. 8, no. 2, pp. 264–270, 2022.

[4] E. B. Hamida, K. L. Brousmiche, H. Levard, and E. Thea, "Blockchain for enterprise: overview, opportunities and challenges," in *The 13th Int. Conf. on Wireless and Mobile Communications (ICWMC'17)*, 2017.

[5] R. M. Nadir, "Comparative study of permissioned blockchain solutions for enterprises," in *Int. Conf. on Innovative Computing (ICIC)*. IEEE, 2019, pp. 1–6.

[6] A. Monrat, O. Schelén, and K. Andersson, "Performance evaluation of permissioned blockchain platforms," in *IEEE Asia-Pacific Conf. on Computer Science and Data Engineering*. IEEE, 2020, pp. 1–8.

[7] J. Sedlmeir, P. Ross, A. Luckow, J. Lockl, D. Miehle, and G. Fridgen, "The dlps: a new framework for benchmarking blockchains," in *Hawaii Int. Conf. on System Sciences*, 2021.

[8] M. Dabbagh, K.-K. R. Choo, A. Beheshti, M. Tahir, and N. S. Safa, "A survey of empirical performance evaluation of permissioned blockchain platforms: Challenges and opportunities," *Computers & security*, vol. 100, p. 102078, 2021.

[9] J. Polge, J. Robert, and Y. Le Traon, "Permissioned blockchain frameworks in the industry: A comparison," *Ict Express*, vol. 7, no. 2, pp. 229–233, 2021.

[10] D. L. Fekete and A. Kiss, "A survey of ledger technology-based databases," *Future Internet*, vol. 13, no. 8, p. 197, 2021.

[11] G. A. Thomaz, G. F. Camilo, L. A. C. de Souza, and O. C. M. Duarte, "Architecture and performance comparison of permissioned blockchains platforms for smart contracts," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 1–6.

[12] M. K. Gupta, R. K. Dwivedi, A. Sharma, M. Farooq *et al.*, "Performance evaluation of blockchain platforms," in *Int. Conf. on IoT, Communication and Automation Technology (ICICAT)*. IEEE, 2023, pp. 1–6.

[13] V. Capocasale, D. Gotta, and G. Perboli, "Comparative analysis of permissioned blockchain frameworks for industrial applications," *Blockchain: Research and Applications*, vol. 4, no. 1, p. 100113, 2023.

[14] CoinMarketCap. (2023) Today's cryptocurrency prices by market cap. [Online]. Available: https://coinmarketcap.com

[15] Ripple. (2023) Xrpl repository. [Online]. Available: https://github.com/XRPLF/rippled

[16] Cosmos. (2023) Cosmos sdk repository. [Online]. Available: https://github.com/cosmos/cosmos-sdk

[17] Cosmos, "What is cosmos?" https://v1.cosmos.network/intro, 2022.

[18] M. Mazzoni, A. Corradi, and V. Di Nicola, "Performance evaluation of permissioned blockchains for financial applications: The consensys quorum case study," *Blockchain: Research and Applications*, vol. 3, no. 1, p. 100026, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S209672092100021X

[19] E. Shevchenko and R. Lunsford. (2023) Blockchain disruption in finance: JPMorgan chase's success story and the transfer of Quorum to ConsenSys. [Online]. Available: https://edeconomy.com/wp-content/uploads/2023/08/Blockchain-Disruption-in-Finance-JPMorgan-Chases-Success.pdf

[20] K. Christodoulou, E. Iosif, A. Inglezakis, and M. Themistocleous, "Consensus crash testing: Exploring ripple's decentralization degree in adversarial environments," *Future Internet*, vol. 12, no. 3, p. 53, 2020.

[21] M. Needham, "Global spending on blockchain solutions forecast to be nearly $19 billion in 2024, according to new idc spending guide," 2021. [Online]. Available: https://www.idc.com/getdoc.jsp?containerId=prUS47617821

[22] B. Lashkari and P. Musilek, "A comprehensive review of blockchain consensus mechanisms," *IEEE Access*, vol. 9, pp. 43 620–43 652, 2021.

[23] G. S. Veronese, M. Correia, A. N. Bessani, L. C. Lung, and P. Verissimo, "Efficient byzantine fault-tolerance," *IEEE Transactions on Computers*, vol. 62, no. 1, pp. 16–30, 2011.
[24] W. Zou, D. Lo, P. S. Kochhar, X.-B. D. Le, X. Xia, Y. Feng, Z. Chen, and B. Xu, "Smart contract development: Challenges and opportunities," *IEEE Transactions on Software Engineering*, vol. 47, no. 10, pp. 2084–2106, 2019.
[25] J. H. F. Battisti, G. P. Koslovski, M. A. Pillon, C. C. Miers, and N. M. Gonzalez, "Analysis of an Ethereum private blockchain network hosted by virtual machines against internal DoS attacks," in *Advanced Information Networking and Applications*, ser. Lecture Notes in Networks and Systems. Cham: Springer International Publishing, 2022, pp. 479–490.
[26] P. Freni, E. Ferro, and R. Moncada, "Tokenization and blockchain tokens classification: a morphological framework," in *IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2020, pp. 1–6.
[27] P. Lafourcade and M. Lombard-Platet, "About blockchain interoperability," *Information Processing Letters*, vol. 161, p. 105976, 2020.
[28] E. Buchman, "Tendermint: Byzantine fault tolerance in the age of blockchains," Ph.D. dissertation, University of Guelph, 2016.
[29] E. Leka, B. Selimi, and L. Lamani, "Systematic literature review of blockchain applications: Smart contracts," in *2019 Int. Conf. on Information Technologies (InfoTech)*. IEEE, 2019, pp. 1–3.
[30] Ripple. (2023) Use an escrow as a smart contract. [Online]. Available: https://xrpl.org/use-an-escrow-as-a-smart-contract.html
[31] ——. (2023) Xrpl hooks documentation. [Online]. Available: https://xrpl-hooks.readme.io/docs
[32] Geveo. (2023) Evernode. [Online]. Available: https://www.geveo.com/evernode/
[33] onXRP. (2023) Xahau: The 1st smart contracts on the xrp ledger ecosystem. [Online]. Available: https://onxrp.com/xahau/
[34] J. O. Chervinski, D. Kreutz, X. Xu, and J. Yu, "Analyzing the performance of the inter-blockchain communication protocol," *arXiv preprint arXiv:2303.10844*, 2023.
[35] J. Kwon and E. Buchman, "Cosmos whitepaper," *A Netw. Distrib. Ledgers*, vol. 27, 2019.
[36] Hyperledger Foundation. (2023) Benchmarking hyperledger fabric 2.5 performance. [Online]. Available: https://www.hyperledger.org/blog/2023/02/16/benchmarking-hyperledger-fabric-2-5-performance
[37] C. Fan, C. Lin, H. Khazaei, and P. Musilek, "Performance analysis of hyperledger besu in private blockchain," in *Int. Conf. on decentralized applications and infrastructures (DAPPS)*. IEEE, 2022, pp. 64–73.
[38] S. De Angelis, "Assessing security and performance of blockchain systems and consensus protocols: taxonomies, methodologies and benchmarking procedures," Ph.D. dissertation, U. Southampton, 2022.
[39] M. Mazzoni, A. Corradi, and V. Di Nicola, "Performance evaluation of permissioned blockchains for financial applications: The consensys quorum case study," *Blockchain: Research and applications*, vol. 3, no. 1, p. 100026, 2022.
[40] Ripple. (2023) Xrp ledger. [Online]. Available: https://ripple.com/xrp/