IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 21 May 2022, accepted 18 August 2022, date of publication 26 August 2022, date of current version 2 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3201885

RESEARCH ARTICLE

A Blockchain Architecture for Trusted Sub-Ledger Operations and Financial Audit Using Decentralized Microservices

NOUSSAIR FIKRI, MOHAMED RIDA, NOURREDINE ABGHOUR, KHALID MOUSSAID, AMINA EL OMRI, AND MOUNIA MYARA

Department of Mathematics and Computer Science, Hassan II University of Casablanca, Casablanca 20000, Morocco

Corresponding author: Noussair Fikri (noussair.fikri-etu@etu.univh2c.ma)

ABSTRACT Blockchain has become an unavoidable future in enterprise finance, particularly enabling and securing cross-company transactions. By introducing a comparable notion of smart contract, the trusted sub-ledger operation (TSLO), this article will propose a complete architecture based on the Blockchain to solve the traceability and validity of accounting data by assets groupement. TSLO is a more flexible and adaptable method for asset management in the corporate accounting system and the enterprise resource planner. This method is built on a decentralized microservices tree (DMST) and is an extendable E-Bidding form of TEA (Triple Entry Accounting). Instead of using a multi-ledger architecture, the Hyperledger Fabric skeleton, limited to participant channels inside one entity or organization, our approach uses decentralized sub-ledgers with an implementation tree (DMST) for an assets-driven transactions. Furthermore, the government's audit and taxation procedures for financial groups are more accessible by combining Proof of Authority and Proof of Stake to assure the logic of More stake more reputation to preserve.

INDEX TERMS Blockchain, sub-ledger, accounting, hyper-ledger, decentralized, assets, authority, stake.

I. INTRODUCTION

Blockchain is a new generation of transaction-based technology that helps businesses enhance their processes. After decades of investing in in-house software, the financial sector finally realizes its promise [1]. This technique improves transaction reliability and, as a result, should have a wide range of applications in the banking industry, with a high return on investment. However, it is difficult to put this technology into practice: businesses cannot do it alone [2]. They must collaborate with their customers, suppliers, and competitors in new and more active ways. Banks and other financial organizations have long represented the role of ''trusted third party'' checking transaction legitimacy and correctness [3]. Thanks to blockchain technology, we are not totally reliant on a trusted third party. The promise of consent from all parties engaged in a transaction is at the heart of the Blockchain's potential. This is made possible by registering the transaction's provenance and ownership at every stage of its execution. It's because the Blockchain allows each step to be saved and authenticated. It might be used to safeguard and verify any transaction without requiring the involvement of a third party. While cybersecurity is a significant concern in the digital expansion of businesses, the Blockchain provides security for any transaction through its design and operation. Trade finance (international trade financing) is a natural fit for Blockchain since it involves a large number of stakeholders (banks from different countries, suppliers, buyers, warehouses, and so on) in a time-consuming and costly process (e.g., several verifications, issue of letters of credit) [4]. These various stakeholders might save documents directly on a single blockchain [5].

Smart contracts are another aspect of the Blockchain's development: these programs automatically carry out the provisions of a contract. There are a variety of uses, including insurance (for example, claim confirmation and automated payment of the insured) [6], [7], [8]. ERP performs at automating internal procedures. However, if the use case

The associate editor coordinating the review of this manuscript and approving it for publication was Chun-Hao Chen.

extends beyond the company's ''four walls'' and systems, the present platforms are unsuitable. Each company in a supply chain, for example, runs its own ERP, resulting in an information silo that impedes traceability and limits automation options for multi-stakeholder transactions. Even though Blockchain applications are still in their infancy, publishers and experts see it as a logical supplement to ERP, acting as a secure layer of immutable records for data shared throughout a diverse supply chain. With this increased level of integrity, ERP and Blockchain can work together to automate some time-consuming manual operations in a business-to-business workflow context, such as sending invoices or initiating payments [9], [10], [11].

Our research focuses on using Blockchain as a critical layer for a company's ERP. We investigate the means and the architecture that could be ideal for utilization. A company's information system is built around an essential module that enables it to track all its financial activities and submit them to an internal audit to distinguish an economic anomaly caused by poor management or to an external audit agency for tax audits. Theoretically, existing blockchain architectures are more dedicated to a token transaction with a shallow level of information and metadata, which does not allow better routing or grouping of transactions by functional type. Enterprise information systems (IS) (ERP – Accounting applications) have high precision in meta-data, making it challenging to graft the blockchain to state of the art. Conceptually, it is complicated to set up an IS Sub-ledgers/Blockchain merge. It is technically impossible to have leaked processing and organize inter-corporation or inter-sub-ledgers transactions. We chose an architecture with the same properties of Ethereum and smart contracts with a different rationale. Trusted Sub-Ledger Operation can wrap a specific implementation tree containing conditions for each asset family and deliver it in byte code format as a decentralized microservices tree. As a result, the Blockchain may conduct asset-driven transactions while considering the assets family's pre-defined requirements. It can change the way transactions are originated, processed, authorized, recorded, reported, and other recordkeeping activities.

Business models and processes changes may impact Back-office functions such as financial reporting and tax preparation. For example, PoS (Proof of Stake) and PoA (Proof of Authority) are used for transparency and authenticity of operations and less energy consumption. At the same time, an organization that takes the lead in signing transactions, in this case, the authority, can be assigned to an audit finance body. Adopting a microservice architecture benefits adaptability and integration with internal information systems, particularly modules or sub-ledgers with a strong accounting relationship. Our contribution focuses mainly on the resolution of the problems of grouping transactions by assets and adhering to them, this flexibility is not present on the current blockchain architectures. The problem we have targeted in this study is the asset traceability of inter-company transactions (Business To Business) and between companies and the government. The existing architectures could not satisfy this need to solve the problem simply because it requires implementations families specific to each asset group. We have already tried to implement this logic on The Archi Hyper Ledger Fabric (HLF) and ERC-20 (Ethereum). It has generated many implementations involving a lot. That is not very clean and impacts the performance of the blockchain because it comes down to the fact that we have several executions, one for each asset. To do this, we have remodeled the most refined grain of a blockchain structure, The ''Block'' unit, and set up a new architecture to manage the problem of asset groupings, and there is no concrete study aimed at this region. Most companies are moving towards a hyper-ledger fabric that presents tools facilitating nodes and consensus implementation. However, our approach makes this possible by creating implementation families where one inherits from the other to produce a group of implementations. And also, executions by asset families benefit the performance and organization of transactions during the interpretation. For example, suppose we have 1000 transactions involving 100 assets grouped by 10 families. In that case, the execution will be done by massive interpretation of 10 blocks of transactions instead of 1000 executions in the case of an HLF or ERC-20 implementation. Assets sometimes have common conditions and operations. The concept of microservices inheritance has allowed us to create fewer implementations by creating families to avoid the writing and execution of elements that already exist in the case of a particularity to be added or truncated in a specific implementation.

II. RELATED WORKS & CONTRIBUTION

A. OVERVIEW OF BLOCKCHAIN ARCHITECTURE

Blockchain is a decentralized technique for storing and sharing information that a single party does not control. Its architecture is similar to a public accessible distributed database. The technical concept of its operation is self-evident: Blockchain comprises a group or series of blocks linked together by cryptographic strings of characters called ''hash'', as seen in figure 1. The Genesis Block is the first block in the chain, and it allows us to start the chain by constructing and establishing the first hash [2], [3], [12].

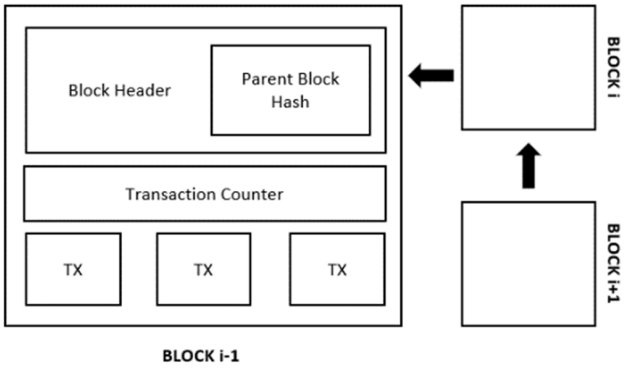

FIGURE 1. Blockchain structure.

A cryptographic hash is a string formed by running it through a hash function, transforming it into another string. The number of characters in the hash remains constant regardless of the length of the encrypted string, which is the first feature of these methods. However, the second aspect of this hash is equally appealing in the context of the Blockchain: even a single comma in a string can totally change the hash. As demonstrated in Figure 2, a block is made up of a header and a body [7], [13], [14]. Here are the attributes of the block header:

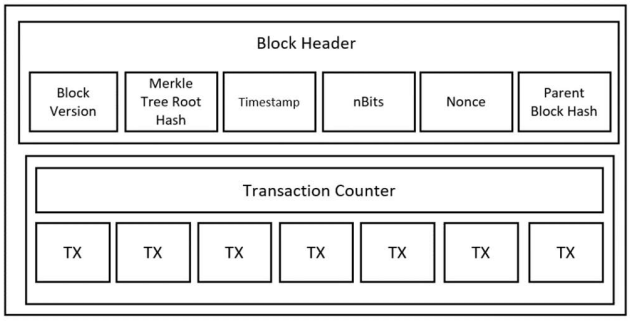

FIGURE 2. Block main components.

- Block version: specifies which set of block validation criteria should be used.
- Merkle tree root hash: the sum of all transactions in the block's hash value.
- Timestamp: current time as seconds in the universal time since January 1, 1970.
- nBits: target threshold of a valid block hash.
- Nonce: a 4-byte field, which usually starts with 0 and increases for every hash calculation
- Parent block hash: a 256-bit hash value that points to the previous block.

The block's body is then made up of a transaction counter and transactions. The maximum number of transactions a block can contain is determined by the block size and the size of each operation. The Blockchain's cryptography is an asymmetric approach for verifying transaction authenticity. In contexts that are deemed to be slightly dependable, the asymmetric cryptographic type digital signature is used [15]. But in the case of an even more specific view on the assets concerned by the transactions, the classic block does not bear more interest on this part. Our contribution will propose a solution to this problem.

B. BLOCKCHAIN ARCHITECTURE OF ETHEREUM

Either a public-centric or a private-centric blockchain architecture exists. A permission-less blockchain network, often known as a public blockchain, allows anyone to join the network without requiring permission. The user can join as a simple node, a validating node, or a mining/block generating node. To attract more people to join, this form of network usually provides an incentive for users to participate in the consensus. A network participant's identity

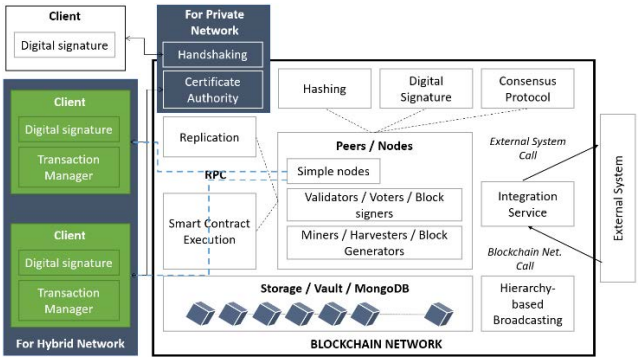

FIGURE 3. Single-ledger based architecture.

is pseudo-anonymous by employing a public key and a pseudo-name. The public nature of the transaction data raises the question of data privacy. A private blockchain, also known as permission Blockchain, is a network only accessible to those invited by an authentication authority. The network uses access-control rights for ledger queries and changes [15], [16].

1) SINGLE-LEDGER-BASED ARCHITECTURE FOR A PUBLIC NETWORK

Different designs have emerged due to the proliferation of blockchain platforms to meet the application requirements in an emerging collaborative environment. The Ethereum platform first offered this architecture in 2013. Peers represent network participants, as shown in Figure 3 (or nodes). There are three types of nodes: simple, complete, and miner. A client uses the RPC to connect to the Blockchain, and the integration service connects to an external system. Suppose the validation of a transaction is dependent on external data, such as the current weather, the price of a stock market, or the currency exchange rate, in that case, an external system is used [12], [15]. Our contribution will not be focused on a single-ledger architecture, but implicitly, it will form the basis of our structure.

2) SINGLE-LEDGER-BASED ARCHITECTURE FOR A PRIVATE NETWORK

The single-ledger-based architecture was established for a private network, by introducing building blocks to handle the concerns of privacy and access control in the public network design. Figure 3 shows the addition of a certificate authority and a handshaking mechanism. The certificate authority provides authentication and authorization for users to join the network. The access-control mechanism specifies how the ledger is queried and updates each participant's role. The handshaking method creates connections between nodes and confirms the legitimacy of the nodes participating in a transaction. Concerning the security and privatization part of the transaction, we keep the same logic for a private network through certificates of authorization.

3) SINGLE-LEDGER-BASED ARCHITECTURE FOR A HYBRID NETWORK

To support the development of applications of hybrid nature (private transactions in a public ledger), blockchain platform Quorum in 2016 introduced a constellation building block to the public architecture, as shown in Figure 3. Examples are real estate, social networking, retail industry, healthcare, and research. A constellation privately allows the submission of transactions by using encryption. It includes a transaction manager and an enclave. The transaction manager keeps the transaction data private and secure by broadcasting the hashed encrypted data to the network. The enclave performs the hashing and encryption/decryption operations.

4) MULTILEDGER-BASED ARCHITECTURE FOR A PRIVATE NETWORK

Hyperledger Fabric, a blockchain platform, introduced a multi-ledger-based design for private networks in 2016, as depicted in Figure 4. The goal is to make possible for a subgroup of participants to conduct confidential and private transactions within an organization or federation. The architecture divides the blockchain network into channels to facilitate private transactions between channel participants.

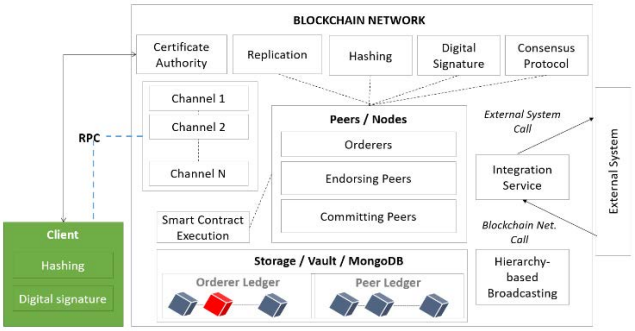

FIGURE 4. Multi-ledger based architecture.

The architecture uses a collection to undertake a private transaction between participants within a subgroup. Compared to building a group within a channel, creating a track within a channel is a CPU-intensive procedure that consumes a lot of energy. In this architecture, the validators are known as peers, while the miners are called orderers. Endorsing peers and committed peers are the two categories of peers. The global state and the ledger are the two components of the ledger in this design. The global state describes the ledger's present state [10], [17].

5) INTEROPERABILITY-BASED ARCHITECTURE

Many blockchain platforms have been developed due to the fast adoption of Blockchain by various application sectors. On the other hand, these platforms support various programming languages, smart contract types and structures, and communication protocols, making interoperability between blockchains problematic. The architecture for interoperability between public and private blockchain networks was

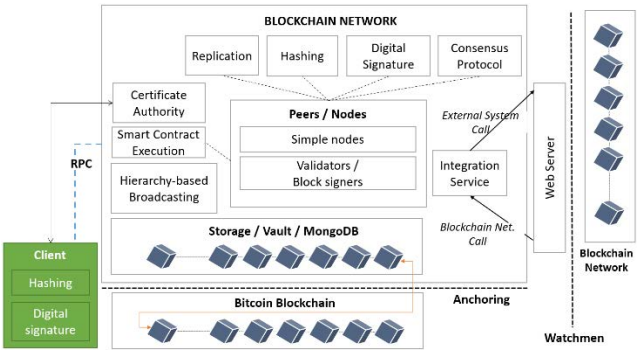

FIGURE 5. Interoperability based architecture.

presented in 2017 by Elements, as shown in Figure 5. The design can also be used to improve a blockchain's security by connecting it to another blockchain. For example, it is used by the platform Openchain to link its Blockchain (dubbed sidechain) to the Bitcoin blockchain (let us call it mainchain). When a new block is added to the sidechain, the cumulative hash for that block is determined by hashing the block hash with the cumulative hash of the previous block. The current cumulative hash is stored in the mainchain block [12], [18], [19].

C. SMART CONTRACTS

All complicated protocols and applications are built based on smart contracts. Smart contracts are small applications that are saved on a blockchain and operated in parallel by many validators. The term "smart contract" was coined by Jani [20]. Many agreements might be "hidden in the hardware and software with which we engage so that a breach of contract is costly for the offender", according to Szabo, who created the concept using the example of a vending machine. Buterin proposed a Blockchain-based decentralized smart contract framework to address any trust difficulties in the execution environment while providing for safe global states [21]. Figure 6 depicts the smart contract's working mechanism. Typically, smart contracts are connected to the Blockchain in computer codes (e.g., a Bitcoin transaction) and recorded in the Blockchain after being propagated by the P2P network and validated by the nodes after signing by all parties.

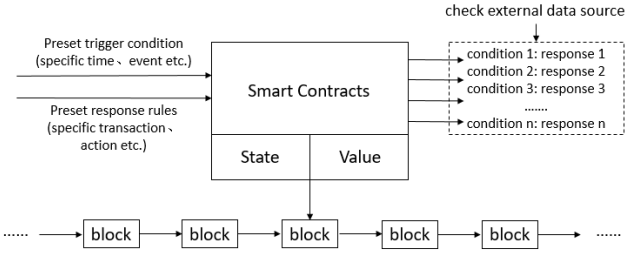

FIGURE 6. Smart contract structure.

A smart contract has several pre-defined states and transition rules, scenarios that trigger contract execution (for example, when a specific event occurs), answers in a particular scenario, and so on. The Blockchain keeps track of smart contracts in real-time and executes them when special trigger conditions are satisfied [16], [22]. It's on this part where our contribution will surface very explicitly. Smart Contracts are made to execute or carry out a single implementation. While on TSLO (Trusted Sub-Ledger Operations), it is possible to create legacy tree implementations that allow setting up execution derivatives and sharing the predefined tree structure on the P2P network of the blockchain. The usefulness of our contribution has a very considerable effect on the structuring of inter-corporate transactional standards. It will also build predefined instructions by asset family to carry out internal and also external (Government) audit processes.

D. CONSENSUS ALGORITHMS

1) PROOF OF WORK: COMPUTE-INTENSIVE BASED CONSENSUS

PoW was first proposed by Dwork and Naor in 1992 and utilized by Back A [12]. in 2002 to minimize the volume of spam emails by making sending several emails at the same time computationally difficult and time-consuming. In 2008, Lai and Chuen employed the PoW method in the Bitcoin blockchain network [23]. While generating legitimate blocks of transactions, blockchain mining nodes compete against one another. In addition to the requirement that the hash output is below a certain threshold, a mining node should hash the block contents using a counter, like in hash. When compared to the hash, this increases the computational complexity of mining. The nonce is the term used in Blockchain to describe the counter value. A mining node hashes the Merkle root hash value, the timestamp, the previous block hash, the block version, and the nonce value to calculate the block hash. The Bitcoin blockchain employs the SHA-256 hash algorithm [12]. The miner who gets the desired hash value adds the nonce to the block header and broadcasts it to the network.

All other miners must halt their mining operations and verify whether the proposed block is genuine. The ledger is updated with a valid block, and the miners begin mining the next block. The Bitcoin network, which uses PoW, can process 60 transactions per second. PoW adds security to the Blockchain by preventing miners from mining incorrect or malicious transactions. On the other hand, Miners can form mining pools to solve the PoW puzzle. Each miner in a pool employs their computing capacity. Then the mining reward is distributed among the miners according to their contribution to the mining reward. Suppose a mining pool controls more than half of the network's computational capacity. In that case, those miners may block proposed transactions from being validated, thereby stopping any transactions between users. In PoW, this is known as the 51 percent attack dilemma. Furthermore, as discussed by Conti *et al.*, PoW is vulnerable to security attacks such as routing, Sybil, eclipse, time jacking, and bribery [3].

King S. proposed prime number PoW in 2013 to channel the high energy consumption of PoW into dual-use. Prime number PoW is similar to PoW except for one variable, it is based on computing the Cunningham chain of prime numbers, which can be used to create an auto-recoverable auto-certifiable cryptosystem that enables a safe, robust, and recoverable file system [3], [24], [25]. Delayed Proof of Work (DPoW) seeks to apply PoW's compute-intensive security to other blockchain networks that use a more energy-efficient consensus algorithm. As a result, DPoW is a hybrid consensus mechanism that uses the mining power of a PoW blockchain to protect a blockchain network. A set of 64 notary nodes (chosen by the network's stakeholders) is responsible for creating a block in DPoW. Each notary node validates the transactions and builds a block in a round-robin method without engaging the compute-intensive and energy-hungry calculation of the mining proof. However, anytime a block is made in the PoW blockchain, the hash of the last created block in the DPoW blockchain is appended to the latter to ensure network security. Before being sent to the PoW blockchain network, the block hash in DPoW is signed by 33 (52%) of the notary nodes. The platform Komodo implements DPoW by using the Bitcoin blockchain's mining power [5], [26]. Furthermore, the mining competition for incentive money exacerbates the problem of energy use. As of June 22, 2019, the yearly energy usage was 67.937 TWh. With rising global warming, this rising energy use impacts the environment. The Bitcoin network's annual carbon dioxide emissions utilizing the PoW algorithm are as high as 22.9 million metric tons, nearly equaling the amount produced by countries like Sri Lanka and Jordan. In the Bitcoin, Litecoin, and Dogecoin networks, the PoW consensus is used [12], [14].

2) CAPABILITY-BASED CONSENSUS PROTOCOLS: PoS AND PoA

Proof of Stake (PoS) was first proposed in 2011 and implemented in 2012 by the cryptocurrency Peercoin (also known as PPcoin) [10], [12]. In PoS, the miners are known as forgers, and the mining process is referred to as forging. Each forger deposits a set amount of owned cryptocurrency coins in the network as a stake at the start of a forging round, which the protocol uses to pick the next forger in the network. PoS has two forger selection methods (1) coin-age selection based on the number of days the coins are staked. Coinage is computed by multiplying the total number of coins staked by a forger by the number of days the stake is held. (2) Random block selection based on calculating a hit value using the forger's private key; each forger encrypts the preceding block's hash with its private key to compute the hit value. Then, the encrypted value is hashed, and the first 8 bytes of the hashed result are translated to a hit value. In 2015, Proof of Authority, a reputation-based consensus process in which the miner's reputation is at stake rather than coins, was presented.

A validator performs the function of a miner in PoAuthority. In this technique, the validators (also known as authorities) are formally approved accounts whose identity is validated by a public notary system and is kept public on-chain for cross-checking. By being a validator, the authority must have a good reputation, which prevents them from engaging in unethical behavior. In a round-robin method, each validator will create a block. When a validator acts maliciously and suggests an invalid block, it has a bad reputation. The bitcoin trading platforms PoA network and Vechain both use PoAuthority. Proof of Reputation (PoR) is a version of PoAuthority in which a reputable institution serves as the validator instead of an authorized identity. Trading platforms Gochain and Menlo one are now using PoR. Because a fixed group of validators mining, the PoAuthority, and PoR algorithms make the blockchain network less decentralized, furthermore, they have not been thoroughly tested for performance and security issues [12], [27], [28].

E. DECENTRALIZED FINANCE DEFI AND SMART CONTRACTS

Decentralized finance (DeFi) is a Blockchain-based financial infrastructure that has recently grown in popularity. The term refers to a set of protocols established on open, unlicensed, and highly interoperable public smart contract systems, such as the Ethereum blockchain [6], [18]. Traditional financial services have been updated to be more open and transparent. The problem, in particular, does not rely on centralized organizations or intermediaries. Instead, decentralized applications and open protocols are used (DApps). The code ensures that agreements are followed, that transactions are safe and secure, and that allowed status changes are recorded on a public blockchain. As a result, this architecture can create an irreversible and highly interoperable financial system with unparalleled transparency, equal access rights, and little need for custodians, central clearinghouses, or escrow services because ''smart contracts'' can handle the majority of these functions.

DeFi uses a multi-layered architecture. Every layer has a distinct role. The layers are built on top of each other, resulting in an open and highly composable infrastructure that anyone can build on, rehash, or use. Here are the different layers of the DeFi stack [21], [6], for example:

- The settlement layer allows the network to securely store ownership data and guarantees that any state changes adhere to its set of rules.
- The asset layer: This category includes all assets issued on top of the settlement layer. This includes the native protocol asset as well as any other assets that have been issued on this Blockchain (usually referred to as tokens).
- The protocol layer: The standards cover specific use cases such as decentralized exchanges, debt markets, derivatives, and on-chain asset management. These standards are commonly implemented as a set of open-access smart contracts (or DeFi applications). As a result, these protocols are extremely compatible with one another.
- The application layer: Is where user-oriented programs that connect to specific protocols are generated. The smart contract interaction is usually abstracted via a web browser-based front interface, making the protocols easier to use.
- Aggregation layer: This is a layer that sits on top of the application layer. Aggregators provide user-centric solutions that integrate with a wide range of apps and protocols.

F. ERP SYSTEMS AND DECENTRALIZED ECOSYSTEM

The accounting process is gathering, identifying, classifying, summarizing, and documenting financial transactions in a company's books of accounts to create financial statements. As a result, the company's profits and financial status may be calculated at regular intervals. IFRS: International Financial Reporting Standards are defined to update earlier IAS standards and incorporate financial transparency in the European Union, as well as numerous Asian and South American countries, but not in the United States, which uses GAAS [29]. The International Financial Reporting Standards (IFRS) developed certain mandatory rules for dealing with corporate activities [30]:

- Statement of financial position
- Statement of comprehensive income
- Statement of changes equity
- Statement of cash flow
- Statement of profit and loss

In the form of a general ledger transaction system, a financial module of a company's information system contains a summary of its financial activity. Sub-ledgers are a sub-transaction system that records the details of business transactions within the general ledger. Sub-ledgers' primary goal is to follow financial activity at a finer level, and the most well-known of them are [31], and:

- Corporate debt.
- Accounts payable invoices.
- Sales orders.
- Purchase orders.
- Customer receivables.

There is a growing body of information regarding blockchain applications in accounting, and triple-entry accounting is one of the newest concepts in the blockchain sector (TEA) [8], [19], [32], [33]. A significant component in a blockchain-based e-bidding to enable data replication over three independent points is a literature evaluation of such applications in the context of ERP and AIS; the third point must be valid on a peer-to-peer network (The blockchain network). Another method focuses on merging financial technology (FinTech), distributed ledger technology (DLT), and decentralized finance (DeFI) to improve the efficiency and security of present AIS and ERP System integration.

Many E-bidding strategies based on Blockchain have been proposed due to the development of blockchain technology. Hardwick *et al.* presented in 2018 to apply the notion of smart contract to government bidding [26], allowing for a fair, transparent, and independently auditable government bidding strategy. On the Ethereum blockchain, Galal presented a smart contract framework for a concisely verifiable sealed-bid auction in 2018 [28]. It also demonstrates how zk-SNARK can be used to create a Vickrey auction on top of the Ethereum network [32]. In 2019, Manimaran *et al.* published a blockchain-based E-bidding system [33]. There is no need for a third party in this model. All bidding transactions will be handled using smart contracts, and the system will ensure that the bidding process' integrity is maintained. E.O. Blass *et al.* presented a system in 2020 to safely perform a variety of sealed-bid auctions using building blocks, which can enhance efficiency and achieve low interactivity between participants to support blockchains or other scenarios where several rounds are time-consuming. X.C. Li proposed a blockchain-based credible e-bidding system (BCES) in 2020 to handle operational compliance, multi-party coordination, and cybersecurity issues in the process of bidding data file distribution, verification, and backtracking [33]. A. Sarfaraz *et al.* suggested a blockchain-based framework for an open-bid auction system in 2021, in which multiple cryptographic primitives are used to consider privacy and security limitations. It replaces the original chain structure with a tree structure to integrate the blockchain framework. Elliptic curve cryptography (ECC) and a dynamic cryptographic accumulator encryption algorithm improve the auctioneer's and the bidder's security. I. Omar *et al.* proposed a solution based on the Ethereum blockchain in 2021, which uses Ethereum smart contracts, decentralized storage systems, and trusted Oracle to capture interactions between auctioneers and bidders to ensure data integrity and transparency, eliminating intermediaries.

G. DECENTRALIZED MICROSERVICES

Web services' principal purpose is to transmit data in an easy-to-understand manner. Web services come in two flavors: REST and SOAP. The most common type of web service is the REST technique. The HTTP protocol is used to consume REST (Representational state transfer), which manipulates services using the four methods POST, GET, PUT, and DELETE [27], [34]. SOAP (Simple Object Access Technology) is a protocol that enables decentralized and distributed XML communication between peers [35]. Input, output, and service parameters are all part of the web service setup. A SOAP service may be consumed using WSDL (Web Service Description Language), while a REST service can be consumed using WADL (Web Service Description Language) (Web Application Description Language). In Figure 7, the two types of online service usage are depicted:

RPC employs the client-server model. The RPC translates a message sent by the asking server (also known as the client) to another server. After receiving the request, the server responds to the client. A client can also ask for a function in a specific format and receive a response in the same format via RPC. Regardless, the URL contains the mechanism for making an RPC API request. RPC supports remote procedure calls in both local and distributed environments. When working with protocol buffers (ProtoBuf), the first step is to generate a proto file that defines the structure of the data you want to serialize, as illustrated in Figure 8. A proto file is simply a plain text file with the ending a.proto. The protocol buffer's data is organized as messages, each of which is a brief, logical record of information that includes a field sequence of name-value pairs [36]. Using Protoc and a special gRPC plugin, gRPC creates code from your proto file, including gRPC client and server code, as well as regular protocol buffer code for populating, serializing, and retrieving your message types [36].

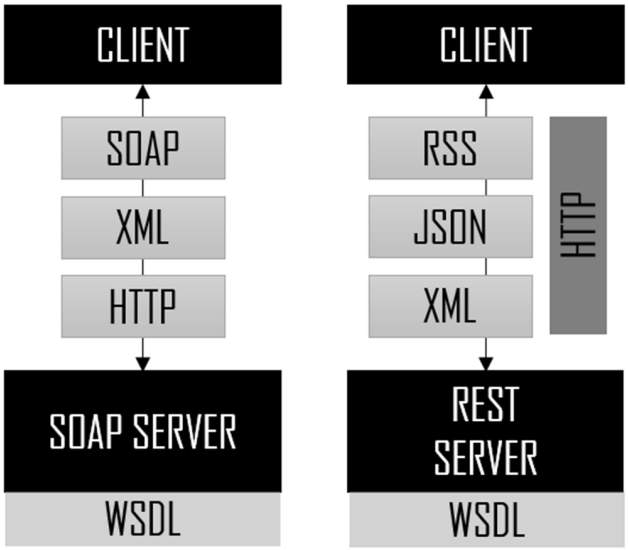

FIGURE 7. REST vs SOAP web services.

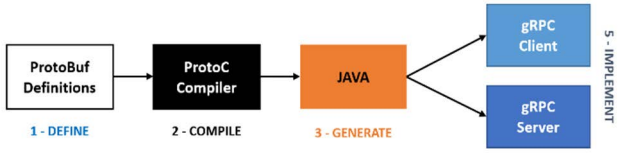

FIGURE 8. GRPC workflow.

An approach proposes a BLockchain-ENabled Decentralized Microservices Architecture for Smart Public Safety (BlendMAS) [25] based on microservices architecture and blockchain technology. A microservices-based security mechanism is introduced within a protected blockchain network to defend data access management in an SPS system.

Security services are decoupled and deployed as distinct containerized microservices that are generated with a smart contract on edge and fog computing nodes. Accord-

ing to a detailed experimental investigation, the suggested BlendMAS could enable distributed IoT-based SPS systems with decentralized, scalable, secure data exchange and access control.

III. METHOD

Our method attempts to create a decentralized architecture of microservices based on the Blockchain concept to simplify the financial transaction auditing process. The P2P network of the Blockchain verifies the activities sent from one organization to another under this design. Each transaction is logged on the appropriate sub-ledger in the company's IS before being transferred to the shared ledger as trusted transactions.

A. BLOCK OF SUB-LEDGER OPERATIONS

Our app's specific block, the TSLO Block (Trusted Sub-Ledger Operations), is made up of the same elements as a traditional block, except for one feature: the grouping of operations or transactions, which changes the block's structure, as shown in Figure 9, with the following main elements:

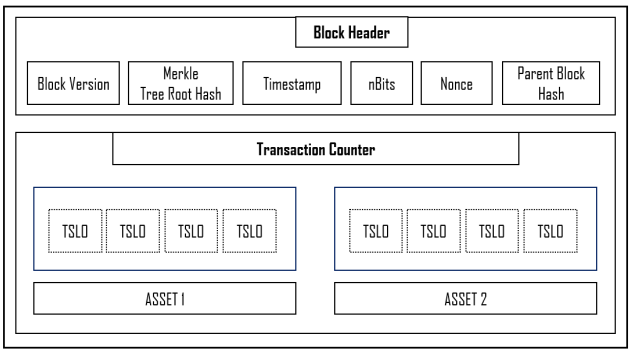

FIGURE 9. TSLO block.

- Block version
- Merkle tree root hash: the hash value of all the transactions in the block. The tree may have a new layer with the root node, the asset node.
- Timestamp
- nBits
- Nonce: There is no need for the nonce on the first stage, only in the case of implementing the PoW mechanism.
- Parent block hash

B. DECENTRALIZED SUB-LEDGERS

To put the concept of decentralized sub-ledgers in effect, we needed to create a grouping of transactions generated by each corporate's sub-ledger. Every company, for example, has a portfolio of products and services. Each asset purchase or sale is represented by a transaction issued by SL (sub-ledger) and labeled SLO, as illustrated in the diagram (Sub-Ledger Operation Not Verified). They are sent to the Blockchain to be confirmed and tagged TSLO (Trusted Sub-Ledger Operation) before being redirected to Decentralized Sub-ledgers before being aggregated by SL.

The main Blockchain contains blocks, including transactions from all sub-ledgers. Those who are verified are placed in blocks ranging from 1 to N (TSLOs), while those who are subject to the Blockchain's P2P network are placed in block N + 1 (SLO) while waiting to be verified by the consensus mechanism. We've shown verified transactions in white and unverified transactions in gray, as well as blocks, in figure 10.

C. DECENTRALIZED MICROSERVICES TREE FOR TSLO

Smart contracts are the equivalent of trusted sub-ledger operations but with more flexibility in implementation which can be represented by a set of instructions or a complete algorithm that handle a specific case for an asset. In our case the implementations are written in plain Java code by taking into consideration an inheritance logic and abstraction of Classes and interfaces in JAVA. The main advantages in using implementations tree, is the benefits of reusable code, if an assets family has already a processing code for a specific scenario and the users have to add a new implementation for a new asset that belong to the existing family, we can just assign this asset to the current family. The TSLO are transactions generated by a dynamic microservices tree where each layer inherits its parent's properties, allowing for imbricated implementations based on asset categories, families, and hierarchy levels, providing a better strategy and alternative.

Each asset, or group of assets, operates based on the implementation surcharge, taking into account the base implementation, which is made up of the following events:

- allSubledgers() : To return all sub-ledgers with their details
- subledgerName(id): To return SubLedger name or functionnal notation inside information syste
- subledgerById(id) : To return SubLedger and its details by its I
- allAssets() : Return all Assets details (ID, Name . . .) for all subledgers
- assetsBySubLedger(id) : Return all Assets (Id, Name, Quantity . . .) for given Subledger
- assetById(id) : Return Asset details by its ID
- assetBalance(id) : Return Asset quantity by its ID
- subledgerOperation(sourceSubledgerId, corpId, targetSubledgerId, assetId, quantity) : transfers a certain quantity of asset from a sub-ledger to a sub-ledger of a specific corporation.
- subledgerAuthorized(sourceSubledgerId, assetId): returns the quantity of asset authorized to be transferred from the source sub-ledger.

Figure 11 assumed three different families' assets and three transactions for each. The implementations previously mentioned are contained in the Abstract TLSO Service tree and in node 1 of services, which includes two implementations specific to the Asset 1 and Asset 2 families, and the node N (In the same depth of node 1), which has

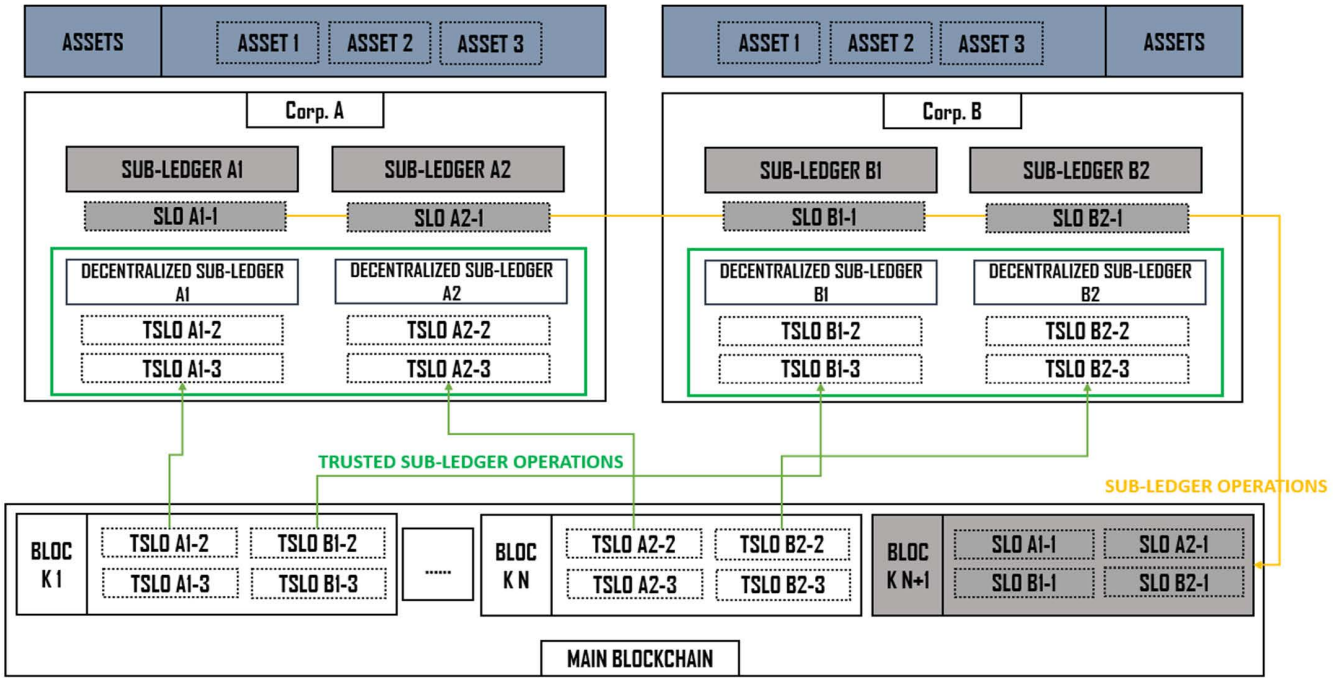

FIGURE 10. Decentralized sub-ledgers.

one implementation dedicated to the Asset 3 family. The deployment of decentralized microservices corresponds to the submission of transactions that are conditional on specific implementations known as SLO (sub-ledger operation) (Not yet trustworthy) to a consensus mechanism and then becomes TSLO (Trusted Sub-Ledger Operation). An arborescence of decentralized microservices characterizes each blockchain node's asset management and transaction processing.

D. MIXED POS AND POA AS CONSENSUS

The PoS (Proof Of Stake) algorithm aims to reach a distributed consensus by assigning validators based on the following criteria: The stake's size of the stake; age, and Randomization. The stacking amount qualifies a node execution. This does not necessitate large investments in equipment or energy. If you do not have enough stake in contributing, you can join a pool of investors. And the stacking is becoming much more dispersed. It allows for increased participation, and a higher number of nodes does not imply a higher return, as in the mining industry. Stacking allows for secure sharding. Ethereum's shard chains will simultaneously create many blocks, increasing transaction debit. The network sharing in a proof-of-concept system would reduce the amount of power required to compromise a portion of the network.

The authority proof grants the right of validation to the node with the best reputation, and the block generation is done via the Round Robin approach. This approach provides some security, regulatory transparency, and significant capacity benefits, albeit at the expense of a low but not insignificant level of decentralization. The combination of limiting the validator's status to a small number of people who have successfully completed a verification process and established economic and reputational incentives (validation agents have something to lose) creates an inherent level of trust among the participants. Because there is no competition among validators to create blocks, the transaction fee may be increased (faster block times), yet energy consumption and computational complexity are significantly reduced (in comparison to the proof of work). The reduction of computing and energy requirements lowers the operating costs of validation agents. Combined with the increase in debt, this reduces transaction costs and makes them more predictable than those on our approach. We used a combination of two consensus approaches, evidence of stake and authority, in our approach. As shown in Figure 12, the verification of SLO (Sub-Ledger Operation) transactions is divided into four steps:

- **Step 1:** The decentralized microservices tree has added a new transaction (SLO) of the sub-grand livre to the P2P blockchain network.
- **Step 2:** The composed algorithm (Proof Of Stake + Proof Of Authority) assigns a primary validator with the most significant interest among the nodes with authority.
- **Step 3:** The chosen validation node adds the new SLO transaction to the current block (Prochain block to verify).
- **Step 4:** The verified block is added to the main Blockchain after being validated by the consensus on the P2P network.

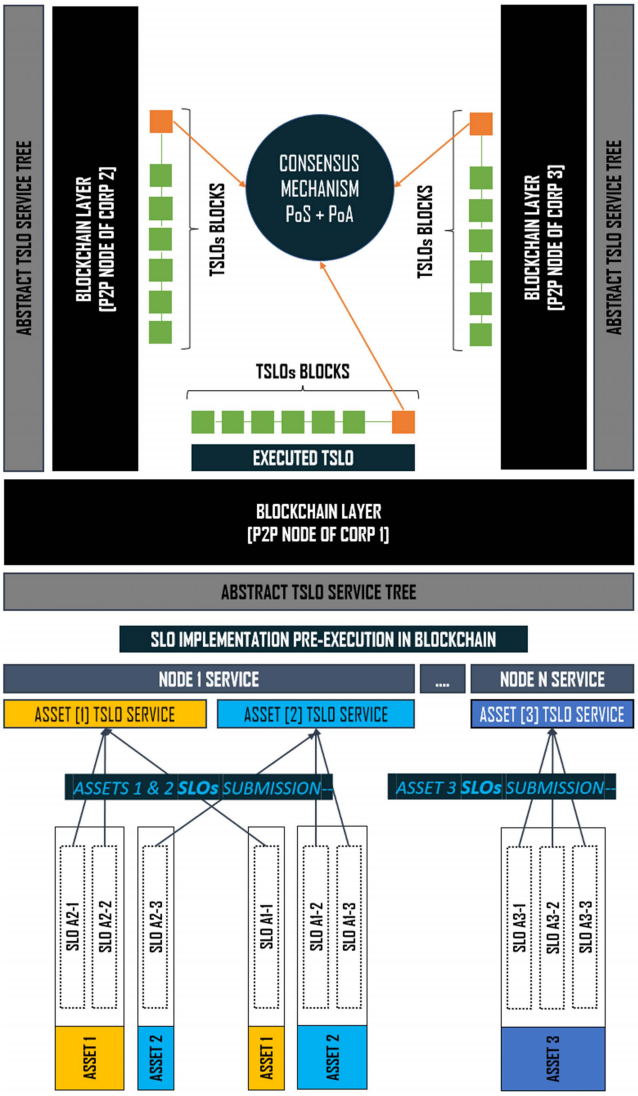

FIGURE 11. Decentralized microservices tree.

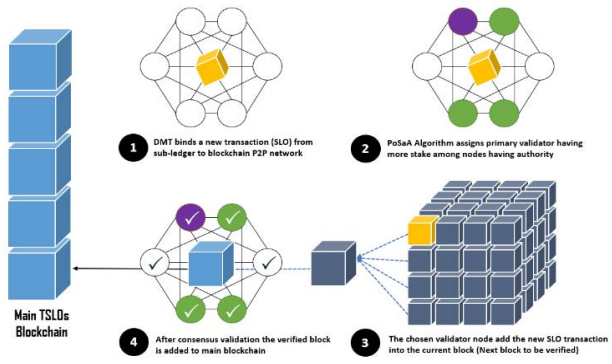

FIGURE 12. PoS + PoA consensus.

E. EXECUTION MECHANIC

In main Blockchain, we have Z corporations and Y assets family. For each corporation C, we have X_Z assets, where:

$$\mathbf{TS} = \{t_0, \ldots.., X_n\} \mid n \in [0, N]$$

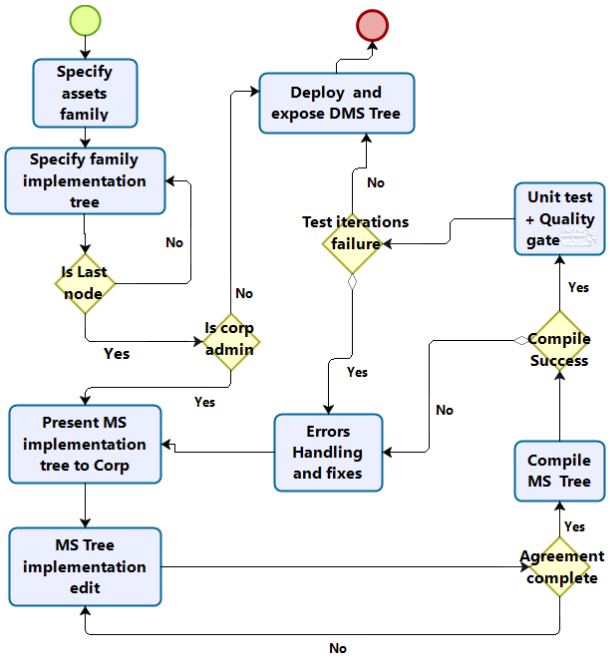

FIGURE 13. DMS implementations tree development workflow.

where N is the number of all transactions, and TS is the transactions set triggered by Information systems of all corporations having their own node in the main block, then we have:

$$\mathbf{TFS} = \{t_0, \ldots.., X_k\} \mid n \in [0, K_f] \mid K_f \, \mathbf{C} \, N$$

where K_F is the number of transactions in the family (f). Then for each family F:

$$F_{parent} = \{F_{Child\,0}, \ldots.., F_{Child\,j}\} \mid j \in [0, J] \mid J \, \mathbf{C} \, Y$$

where J is the number of family nodes that are decentralized and available across all blockchain nodes, we also have

$$F_j = \{A_0, \ldots.., A_i\} \mid i \in I_f \mid [0, I_f] \, \mathbf{C} \, X_z$$

where A_i is the asset at index **i** into family **j**. And for each node of F_{Parent} We have:

$$E_F = \{E_{FC0}, \ldots.., E_{FCl}\} \mid l \in [0, L]$$

where E_F is the execution of family F service and E_{FCl} is the execution of a family child. E_{FCl} implements all properties and execution of E_F. And **L** is the number of executions in its family.

The main implementation of Microservices Tree can be seen as follow:

$$\mathbf{T} = \{E_{FC\,0}, \ldots.., F_{FC\,m}\} \mid m \in [0, M] \mid M \, \mathbf{C} \, Y$$

Figure 13 shows the workflow of family assets and implementations tree deployment

It is assumed that there are a 3 corporations, C1, C2 and C3 environment with the N1, N2 and N3 nodes on the p2p network. To do our tests, we simulated the same behavior on

TABLE 1. Comparison between Hyperledger, ethereum ERC-20, and DMS tree (our approach).

Features	Hyperledger	Ethereum	DMS Tree
Purpose	*The preferred platform for B2B businesses*	*A platform for B2C businesses and generalized applications*	*A platform for B2B & B2B businesses and generalized applications*
Confidentiality	*Confidential transactions*	*Transparent*	*Confidential transactions / Transparent to Government*
Mode of Peer Participation	*Private and Permissioned Network*	*Public/Private and Permissionless Network*	*Private and Permissioned Network*
Consensus Mechanism	*Pluggable Consensus Algorithm: No mining is required*	*PoW Algorithm: Consensus is reached by mining*	*PoS + PoA as Hybrid Consensus Algorithm*
Programming Language	*Chaincode written in Golang*	*Smart Contracts are written in Solidity*	*Implementation Tree is written in Java*
Cryptocurrency	*No built-in cryptocurrency*	*Built-in cryptocurrency called Ether*	*No built-in cryptocurrency*
Development Cycles for current Scenarios	*6*	*6*	*3*
	Neutral	**Deprecated**	**Recommended**

the 3 architectures: HyperLedger – ERC-20 and DMS TSLO. The behaviour in question is as follows:

$$\mathbf{T1(C1 => C2) - [Family\ 1/Asset\ 1]}$$
$$\mathbf{T2(C2 => C1) - [Family\ 1/Asset\ 2]}$$
$$\mathbf{T3(C2 => C3) - [Family\ 2/Asset\ 3]}$$
$$\mathbf{T4(C3 => C2) - [Family\ 2/Asset\ 4]}$$
$$\mathbf{T5(C1 => C3) - [Family\ 2/Asset\ 5]}$$
$$\mathbf{T6(C3 => C1) - [Family\ 3/Asset\ 6]}$$

With Tx is a transaction between Ca and Cb of Asset Y having implementation Z of family Z. Table 4 describes the main differences between the 3 candidates and the verdict for each one:

F. MAIN ARCHITECTURE: TRUSTED SUB-LEDGER OPERATIONS AND DECENTRALIZED MICROSERVICES

In figure 14 we've assumed the existence of an ecosystem involving three businesses and their information systems. Each IS having three or more databases, either SQL or NoSQL, that act as data sources and targets for internal microservices representing each sub-ledger. In our schema, we have two microservices for each of our three ISs, which could be account payable and account receivable sub-ledgers, respectively. Our operations will start from their SLs. Microservices 1 and 2 will be in charge of storing transactions on the database and then transmitting them to the DMST (Decentralized Microservices Tree) via gRPC messages. The DMST creates DSLs, decentralized sub-ledgers containing SLOs (Sub-Ledger Operations), which are not verified because they have not yet been submitted to the PoSaA to become TSLOs (Trusted Sub-Ledger Operation). The SLO encapsulates the conditions (by asset family or asset, more specifically) implemented on the microservices tree. Decentralized Sub-Ledgers are displayed as a pool of elements at the start of the leading Blockchain. We believe that every business needs at least one information system to concentrate its activities through modules (Purchase – Sale – Stock, and so on). Each module is linked to a sub-ledger that communicates with decentralized microservices using the gRPC protocol. Decentralized microservices are instances with two primary parts: the blockchain part, which contains the replication of the shared chain on a peer-to-peer network, and the part code, which is in charge of the impact code of our accounting concept: TRUSTED SUB-LEDGER OPERATION. The operation is sent from the sub-ledger to the decentralized microservice to insert the effect code and then to p2p for validation and blockchain chaining. Finally, the operation is retransmitted to the general ledger as a verified operation after it has been validated.

IV. DISCUSSIONS

The main problems that prompted us to develop this approach are whether smart contracts are completely compatible with the exchange of business transactions in any sort of assets? And could Trusted Sub-Ledgers Operation (Our solution) overcome Ethereum / Smart Contract in reconsidering financial audit? The following statements are direct answers:

- A smart contract comprises a set of pre-defined states and transition rules, as well as scenarios that cause the agreement to be executed. Those encapsulations are exclusive to one token and are deployed in EVM (Ethereum Virtual Machine) (For example, ERC-20). On the other hand, a trusted Sub-Ledger Operation can encapsulate a specific implementation tree comprising conditions for each asset family and deploy it as a byte code-based decentralized microservices tree. As a result, the Blockchain may conduct asset-driven transactions while considering the assets family's pre-defined requirements.
- Blockchain technology's Trusted Sub-Ledger Operation can change all recordkeeping procedures, such as how originated transactions are processed, authorized, recorded, and reported. In addition, changes in business models and processes may impact Back-office functions such as financial reporting and tax preparation. As new blockchain-based methodologies and procedures arise, CPA (Certified Public Accountant) auditors' roles and skillsets may alter. Methods for gathering ade-

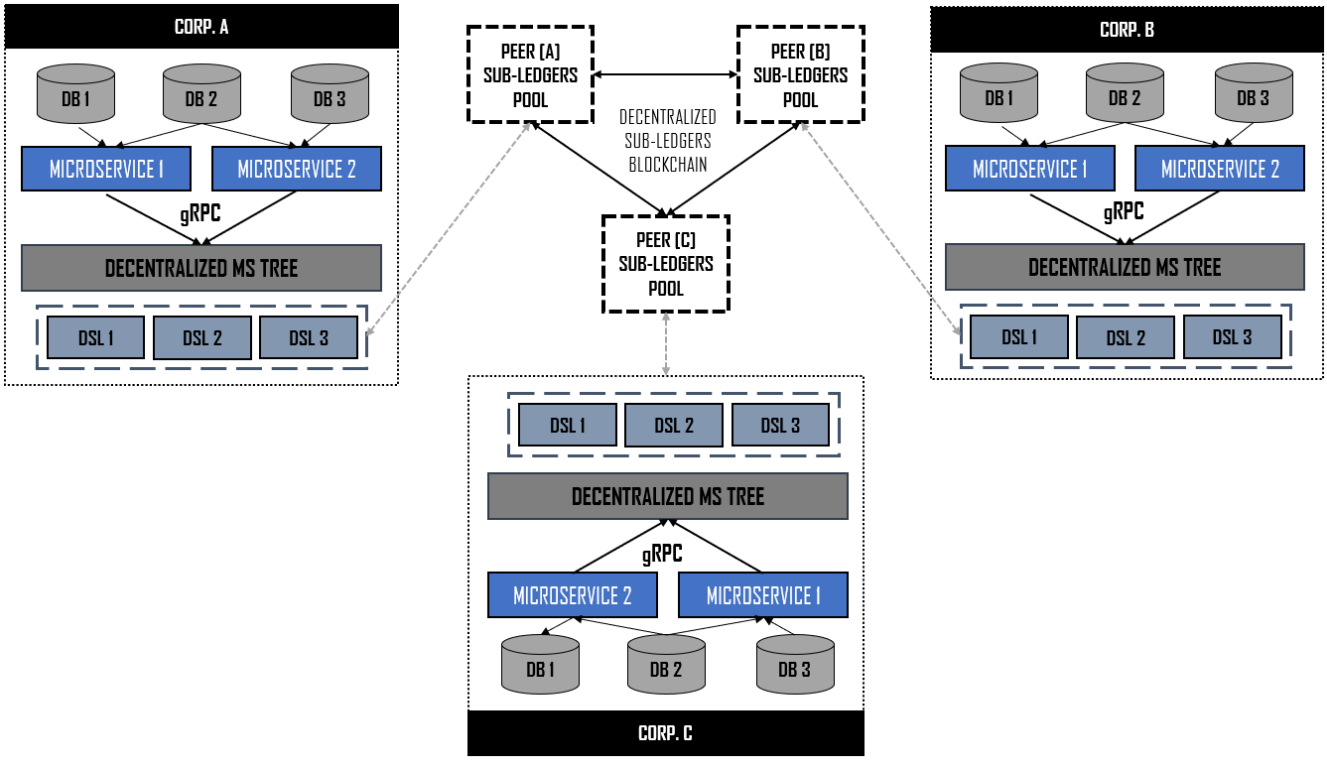

FIGURE 14. Main architecture.

quate, relevant audit evidence, for example, will need to consider both traditional stand-alone general ledgers and blockchain ledgers. Furthermore, better standardization and openness in reporting and accounting may allow for more efficient data extraction and analysis.

- The use of two mechanisms, proof of stake and proof of authority, ensures block validation by relying on peer-to-peer nodes with the most stake and authority to provide the constraint: With a higher stake, there is a greater need to protect one's reputation. In this case, the authority might be attributed to a financial auditing organization as well as designing an entity that takes the lead for transaction signatures.
- Using a microservices architecture has advantages in adaptability and interoperability with internal information systems and direct interaction with modules or sub-ledgers with solid accounting ties.
- The Hyper Ledger Fabric is a multi-Ledger architecture based on the principle of private and confidential transactions between sub-groups of an organization represented by participant channels (Clients groups). The Blockchain is divided into two ledgers, hence the term ''multi-ledgers'', the first of which is the order ledger, where miners operate, and the second is the peer ledger, where peer validation occurs. This approach does not fully address our issue because it is not enough to have internal multi-ledgers with smart contract execution for each Token (in our case, type ERC-20), but rather to have decentralized sub-ledgers for a group of companies. In our approach, this logic is represented by DSL Pool, which groups the sub-ledgers of each company. The ledgers of information systems are books that include inter-society transactions organized by assets or asset families and categorize them according to a set of rules unique to each family. Therefore, applying the logic of contract smarts will result in a large number of implementations as well as doubles. Our approach solves the problem using a decentralized microservices tree (DMST). The main goal is to divide implementations into asset families while maintaining connectivity between internal microservices and decentralized sub-ledgers.

V. CONCLUSION & FUTURE WORKS

The auditing profession must accept and ''lean in'' to the advantages and challenges that widespread blockchain adoption will offer. Advances in blockchain technology present opportunities for CPA auditors and assurance providers to grow, learn and exploit their proven ability to adapt to the needs of a rapidly changing corporate world. We have established a complete architecture that offers a solution to the challenges mentioned above, and we have illuminated the solution's major components in this paper. To be able to pose the final chain, we investigated numerous blockchain architectures, and blockchain integration approaches to ERP-type

information systems. We've clarified the general and individual components that make up our blockchain architecture. We are sure that integrating our technique is a first step toward integrating blockchain technology into the financial auditing industry and that we can now move on to studying the developing limits of this union. We have already implemented a Java API with pivotal microservices, which can be easily decentralized. Our approach's next stage and vision will cover some enhancements to the current architecture then propose a concrete stable version for individual entities or corporations according to governmental standards and policies. We have many other projections concerning the implementation of this architecture by setting up a platform under a middleware that will be grafted to any information system, which gives us a new challenge of adaptability, and which constitutes work of theoretical and technical improvements of our approach.

REFERENCES

[1] D. Mohanty, "The world of blockchains," in *Ethereum for Architects and Developers: With Case Studies and Code Samples in Solidity*, D. Mohanty, Ed. Berkeley, CA, USA: Apress, 2018, pp. 1–36, doi: 10.1007/978-1-4842-4075-5_1.

[2] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *Int. J. Web Grid Serv.*, vol. 14, no. 4, p. 352, 2018, doi: 10.1504/IJWGS.2018.095647.

[3] A. P. Joshi, M. Han, and Y. Wang, "A survey on security and privacy issues of blockchain technology," *Math. Found. Comput.*, vol. 1, no. 2, pp. 121–147, May 2018, doi: 10.3934/mfc.2018007.

[4] G. Prause, "Smart contracts for smart supply chains," *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 2501–2506, 2019, doi: 10.1016/j.ifacol.2019.11.582.

[5] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *Proc. IEEE Int. Congr. Big Data (BigData Congress)*, Jun. 2017, pp. 557–564, doi: 10.1109/BigDataCongress.2017.85.

[6] I. A. Omar, H. R. Hasan, R. Jayaraman, K. Salah, and M. Omar, "Implementing decentralized auctions using blockchain smart contracts," *Technol. Forecasting Social Change*, vol. 168, Jul. 2021, Art. no. 120786, doi: 10.1016/j.techfore.2021.120786.

[7] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016, doi: 10.1109/ACCESS.2016.2566339.

[8] Y.-H. Chen, S.-H. Chen, and I.-C. Lin, "Blockchain based smart contract for bidding system," in *Proc. IEEE Int. Conf. Appl. Syst. Invention (ICASI)*, Apr. 2018, pp. 208–211, doi: 10.1109/ICASI.2018.8394569.

[9] F. I. Lessambo, "Overview of financial statements," in *Financial Statements: Analysis and Reporting*, F. I. Lessambo, Ed. Cham, Switzerland: Springer, 2018, pp. 3–22, doi: 10.1007/978-3-319-99984-5_1.

[10] P. Thakkar, S. Nathan, and B. Viswanathan, "Performance benchmarking and optimizing hyperledger fabric blockchain platform," in *Proc. Int. Symp. Model., Anal. Simul. Comput. Telecommun. Syst. (MASCOTS)*, Sep. 2018, pp. 264–276, doi: 10.1109/MASCOTS.2018.00034.

[11] G.-W. Bock, E. Flores, D. Latumahina, H. Cheng, V. T. Lam, C. Stephanie, R. Soeharto, and Y. J. Kang, "Integrating ERP systems in a decentralized company: A case study," *J. Inf. Technol. Case Appl. Res.*, vol. 11, no. 1, pp. 47–64, Dec. 2021. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/15228053.2009.10856153

[12] L. Ismail and H. Materwala, "A review of blockchain architecture and consensus protocols: Use cases, challenges, and solutions," *Symmetry*, vol. 11, no. 10, p. 1198, Oct. 2019, doi: 10.3390/sym11101198.

[13] N. Chaudhry and M. Yousaf, "Consensus algorithms in blockchain: Comparative analysis, challenges and opportunities," in *Proc. 12th Int. Conf. Open Source Syst. Technol. (ICOSST)*, Dec. 2018, pp. 54–63, doi: 10.1109/ICOSST.2018.8632190.

[14] N. R. Potlapally, S. Ravi, A. Raghunathan, and N. K. Jha, "Analyzing the energy consumption of security protocols," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, New York, NY, USA, Aug. 2003, pp. 30–35, doi: 10.1145/871506.871518.

[15] D. Mohanty, "Ethereum architecture," in *Ethereum for Architects and Developers: With Case Studies and Code Samples in Solidity*, D. Mohanty, Ed. Berkeley, CA, USA: Apress, 2018, pp. 37–54, doi: 10.1007/978-1-4842-4075-5_2.

[16] D. Mohanty, "Ethereum use cases," in *Ethereum for Architects and Developers: With Case Studies and Code Samples in Solidity*, D. Mohanty, Ed. Berkeley, CA, USA: Apress, 2018, pp. 203–243, doi: 10.1007/978-1-4842-4075-5_9.

[17] E. Androulaki, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, New York, NY, USA, Apr. 2018, pp. 1–15, doi: 10.1145/3190508.3190538.

[18] F. Schär, "Decentralized finance: On blockchain- and smart contract-based financial markets," Social Sci. Res. Netw., Rochester, NY, USA, Tech. Rep., Mar. 2020, doi: 10.2139/ssrn.3571335.

[19] S. M. Sarkintudu, H. H. Ibrahim, and A. B. Abdwahab, "Taxonomy development of blockchain platforms: Information systems perspectives," in *Proc. AIP Conf. Proc.*, Sep. 2018, Art. no. 020130, doi: 10.1063/1.5055532.

[20] S. Jani, "Smart contracts: Building blocks for digital transformation," Tech. Rep., 2020, doi: 10.13140/RG.2.2.33316.83847.

[21] V. Buterin, "A next generation smart contract & decentralized application platform," Tech. Rep., pp. 1–36.

[22] D. Mohanty, "Deploying smart contracts," in *Ethereum for Architects and Developers: With Case Studies and Code Samples in Solidity*, D. Mohanty, Ed. Berkeley, CA, USA: Apress, 2018, pp. 105–138, doi: 10.1007/978-1-4842-4075-5_4.

[23] R. Lai and D. L. K. Chuen, "Blockchain—From public to private," in *Handbook of Blockchain, Digital Finance, and Inclusion*, vol. 2, D. L. K. Chuen and R. Deng, Eds. New York, NY, USA: Academic Press, 2018, ch. 7, pp. 145–177, doi: 10.1016/B978-0-12-812282-2.00007-3.

[24] J. A. T. Casallas, J. M. Cueva-Lovelle, and J. I. R. Molano, "Smart contracts with blockchain in the public sector," *Int. J. Interact. Multimedia Artif. Intell.*, vol. 6, no. 3, p. 63, 2020, doi: 10.9781/ijimai.2020.07.005.

[25] R. Xu, S. Y. Nikouei, Y. Chen, E. Blasch, and A. Aved, "BlendMAS: A blockchain-enabled decentralized microservices architecture for smart public safety," in *Proc. IEEE Int. Conf. Blockchain (Blockchain)*, Jul. 2019, pp. 564–571, doi: 10.1109/Blockchain.2019.00082.

[26] F. S. Hardwick, R. N. Akram, and K. Markantonakis, "Fair and transparent blockchain based tendering framework—A step towards open governance," in *Proc. 17th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun./ 12th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Aug. 2018, pp. 1342–1347, doi: 10.1109/TrustCom/BigDataSE.2018.00185.

[27] E. D. Nitto, L. Florio, and D. A. Tamburri, "Autonomic decentralized microservices: The Gru approach and its evaluation," in *Microservices: Science and Engineering*, A. Bucchiarone, N. Dragoni, S. Dustdar, P. Lago, M. Mazzara, V. Rivera, and A. Sadovykh, Eds. Cham, Switzerland: Springer, 2020, pp. 209–248, doi: 10.1007/978-3-030-31646-4_9.

[28] H. Galal, "Succinctly verifiable sealed-bid auction smart contract," in *Proc. Int. Workshop Data Privacy Manage.*, Barcelona, Spain, Sep. 2018, pp. 3–19, doi: 10.1007/978-3-030-00305-0_1.

[29] F. I. Lessambo, "IFRS and GAAP," in *Financial Statements: Analysis and Reporting*, F. I. Lessambo, Ed. Cham, Switzerland: Springer, 2018, pp. 299–324, doi: 10.1007/978-3-319-99984-5_22.

[30] D. Bensadon and N. Praquin, *IFRS in a Global World: International and Critical Perspectives on Accounting*. Springer, 2016. Accessed: Dec. 24, 2021. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02284403

[31] D. Mohanty, "Advanced programming in oraclize and IPFS, and best practices," in *Ethereum for Architects and Developers: With Case Studies and Code Samples in Solidity*, D. Mohanty, Ed. Berkeley, CA, USA: Apress, 2018, pp. 151–179, doi: 10.1007/978-1-4842-4075-5_6.

[32] M. Farnaghi and A. Mansourian, "Blockchain, an enabling technology for transparent and accountable decentralized public participatory GIS," *Cities*, vol. 105, Oct. 2020, Art. no. 102850, doi: 10.1016/j.cities.2020.102850.

[33] A. Sarfaraz, R. K. Chakrabortty, and D. L. Essam, "A tree structure-based improved blockchain framework for a secure online bidding system," *Comput. Secur.*, vol. 102, Mar. 2021, Art. no. 102147, doi: 10.1016/j.cose.2020.102147.

[34] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman, "Analysis of the HTTPS certificate ecosystem," in *Proc. Internet Meas. Conf.*, New York, NY, USA, Oct. 2013, pp. 291–304, doi: 10.1145/2504730.2504755.

[35] S. Malik and D.-H. Kim, "A comparison of RESTful vs. SOAP web services in actuator networks," in *Proc. 9th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2017, pp. 753–755, doi: 10.1109/ICUFN.2017.7993893.
[36] X. Wang, H. Zhao, and J. Zhu, "GRPC: A communication cooperation mechanism in distributed systems," *ACM SIGOPS Operating Syst. Rev.*, vol. 27, no. 3, pp. 75–86, Jul. 1993, doi: 10.1145/155870.155881.

NOUSSAIR FIKRI was born in Casablanca, Morocco, in 1992. He received the master's degree in computer science and networks engineering from the Faculty of Science, Hassan II University of Casablanca. He is currently pursuing the Ph.D. degree. He is currently working at Leyton, as a Software Architect. His research interests include big data, machine learning, and blockchain applied to financial markets and financial auditing.

MOHAMED RIDA received the Ph.D. degree from University Hassan II Mohammedia, in 2005. He is currently a Professor of computer science at the Faculty of Sciences, Hassan II University of Casablanca, Morocco, where he is a member of the LIMSAD Laboratories. His research interests include transport, geographic information systems, and big data. His Ph.D. thesis subject was "Virtual container terminal: Design and development of an object platform for the simulation of the operations of a container terminal."

NOURREDINE ABGHOUR received the Ph.D. degree from the National Polytechnic Institute of Toulouse, France, in 2004. He is currently an Associate Professor with the Faculty of Science, Hassan II University, Morocco. His research interest includes security in distributed computing systems.

KHALID MOUSSAID received the B.Sc. degree in applied mathematics, the master's degree in computer science, and the Ph.D. degree in oriented object database. He is currently the Director of Computer Science, Modeling Systems and Decision Support Laboratory, Hassan II University of Casablanca. His research interests include optimization, algorithmic, and especially in the field of big data and cloud computing.

AMINA EL OMRI is currently a Professor of Higher Education in computer science at the Faculty of Sciences, Hassan II University of Casablanca, Morocco. She has participated with a lot of research papers in workshops and conferences, and published several journal articles. Her main research interests include concern algorithms, optimization, transport, and logistic problems.

MOUNIA MYARA is currently a Professor and a Researcher with the Department of Computer Sciences, Université Hassan II de Casablanca.