IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 5, 2019, accepted December 9, 2019, date of publication December 23, 2019,
date of current version December 31, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2961404

Analysis of Data Management in Blockchain-Based Systems: From Architecture to Governance

HYE-YOUNG PAIK[1,2], XIWEI XU[1,2], H. M. N. DILUM BANDARA[1], SUNG UNE LEE[3], AND SIN KUANG LO[1,2]

[1]Data61, CSIRO, Sydney, NSW 2015, Australia
[2]School of Computer Science and Engineering, University of New South Wales, Sydney, NSW 2052, Australia
[3]School of Information Systems, Queensland University of Technology, Brisbane, QLD 4000, Australia

Corresponding author: H. M. N. Dilum Bandara (dilum.bandara@data61.csiro.au)

This work was supported in part by the Data61/UNSW collaborative research project "Data Oriented Architectures for Improved Data Management and Analytics on Blockchains."

ABSTRACT In a blockchain-based system, data and the consensus-based process of recording and updating them over distributed nodes are central to enabling the trustless multi-party transactions. Thus, properly understanding what and how the data are stored and manipulated ultimately determines the degree of utility, performance, and cost of a blockchain-based application. While blockchains enhance the quality of the data by providing a transparent, immutable, and consistent data store, the technology also brings new challenges from a data management perspective. In this paper, we analyse blockchains from the viewpoint of a developer to highlight important concepts and considerations when incorporating a blockchain into a larger software system as a data store. The work aims to increase the level of understanding of blockchain technology as a data store and to promote a methodical approach in applying it to large software systems. First, we identify the common architectural layers of a typical software system with data stores and conceptualise each layer in blockchain terms. Second, we examine the placement and flow of data in blockchain-based applications. Third, we explore data administration aspects for blockchains, especially as a distributed data store. Fourth, we discuss the analytics of blockchain data and trustable data analytics enabled by blockchain. Lastly, we examine the data governance issues in blockchains in terms of privacy and quality assurance.

INDEX TERMS Analytics, blockchain, databases, data governance, data handling, distributed data management, distributed databases, software architecture, transaction databases.

I. INTRODUCTION

The transformative potential of blockchain technology is often compared to that of the World Wide Web. In just a few years, besides the initial cryptocurrency applications, the foundations of blockchain technology are now being utilised for many other classes of applications, such as asset management, medical/health, finance, and insurance, to name a few. From the viewpoint of such applications, blockchain enhances the quality of the data through transparency, immutability, and consistency [1].

However, the exact nature of blockchains that gives these benefits also brings new challenges from a data management perspective. For example, in terms of the blockchain as a data store and a processing network, following open issues could be observed:

- The data models used in blockchains vary from *key-value* to document stores and are usually combined with "off-chain" data stores. Therefore, searching and retrieving heterogeneous data in blockchain-based systems takes hand-crafted and ad-hoc programming efforts, unlike the abstract and declarative query techniques in conventional databases. Considering the increasing demand for blockchain data analytics at scale, understanding how to efficiently access, integrate, and analyse data in this heterogeneous environment is essential.
- The volume of data that blockchain networks store and manage will only grow with time. However, many contemporary implementations show low throughput, low scalability, and high latency. Besides, to offset

The associate editor coordinating the review of this manuscript and approving it for publication was Mingjun Dai.

the high cost of building trust among transacting parties through consensus, as well as to discourage dormant data, fees are charged in public blockchains for both storing and manipulating data. Properly evaluating the on-chain/off-chain data architectural choices of a blockchain application can help resolve some of these issues.

- The data stored in blockchains are permanent and transparent to the whole network. This brings about a range of data governance issues such as privacy and quality assurance. While storing data in encrypted form is recommended, it could be subject to brute-force decryption attacks in the future (e.g., breakthroughs in quantum computing might render current encryption technologies ineffective) or lead to unintended privacy leakages. Therefore, it is imperative to carefully review these issues to help develop adequate frameworks for blockchain data governance to promote effective management and proper use of blockchain technology.

Given these challenges, we believe there is a need to examine the use of a blockchain as a data store in the context of data management. A good understanding of blockchains in terms of how the data are stored and managed can help application developers and database administrators better design and manage a large software system where a blockchain and an auxiliary database may co-exist. It could also avoid sub-optimal designs, errors, and bugs due to unrealistic expectations on how blockchains behave.

Blockchain has been briefly compared with databases in other work regarding functionality and unique properties [2]–[5]. Our work is complementary to these efforts, where we further conceptualise the differences according to how the application developers would generally perceive the software system layers.

In this paper, we systematically examine blockchain technology through a database lens. We aim to enhance the understanding of blockchains as a data store with the objective of enhancing the utility and correct use of blockchains in large software systems. To achieve this, we identify and analyse data management issues that are critical in building and managing blockchain-based applications. We make the following contributions:

1) Propose a new interpretation of blockchain as an application's data store.
2) Identify and evaluate best practices in blockchain data architectures and operational issues.
3) Explore data administration aspects of blockchains.
4) Present practical insights into the emerging topic of blockchain data analytics and trustable data analytics using blockchain.
5) Examine current issues and future directions in the governance of blockchain data privacy and quality.

The remainder of the paper is organised as follows; Section II presents fundamental properties of blockchains that are relevant from a data and software system perspective. Section III presents blockchain terminology and interpretation of blockchain-based applications from a database viewpoint. How data could be integrated and stored in a blockchain are discussed in Section IV. Section V presents administrative aspects of data on a blockchain. Analytics of data stored on a blockchain, as well as how blockchain can help establish trust in data analytics are discussed in Section VI. Governance aspects of data privacy and quality are presented in Section VII and Section VIII presents concluding remarks.

II. BLOCKCHAIN PROPERTIES

Blockchains can provide a trustworthy and neutral data storage platform for a large software system that uses blockchain as a component. Trust and neutrality come from the following properties, which are resulting from the unique design of the ledger structure, the network, consensus protocol, and cryptographic mechanisms it uses:

- *Transparency* – Data stored on a blockchain are accessible to all participants within the blockchain network. Thus, the data on a public blockchain is visible to everyone on the Internet.
- *Immutability* – Due to the distributed consensus process, once data are appended to the blockchain, they cannot be changed or deleted. However, immutability might be probabilistic for blockchains using certain consensus protocols. All the transactions in the blockchain network are stored as immutable records. These immutable records become a public audit trail for regulatory purposes.
- *Consistency* – Distributed consensus and immutability ensure all committed data are visible to all future data manipulations establishing a single truth across the blockchain network.
- *Equal rights* – Due to disintermediation, every participant of the network has the same rights to manipulate and access the blockchain. With different consensus protocols, these rights may be weighted by the computation power or stake owned by the participant.
- *Availability* – Every participate within the blockchain network may host a full replica of the blockchain data. Hence, from the system perspective, the data are available as long as at least one node is in the blockchain network.

From the software architecture perspective, every design decision of a system is a trade-off among multiple properties. Likewise, *Confidentiality* and *Performance* are the two main criticisms arising from the design of blockchain. As there is no privileged user within the blockchain network, every participant can access all data on blockchain compromising confidentiality. Performance refers to the transaction processing rate and the latency of adding and confirming new records. The throughput is limited by the block size configuration and block generation rate. Further, the latency between the submit and commit of a transaction is affected by the consensus protocol, which is around 1 hour (10-minute block interval with

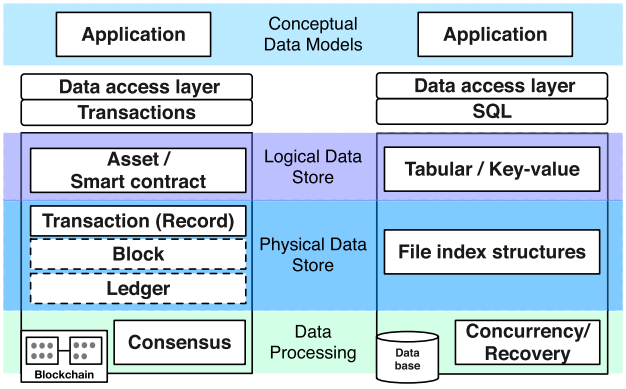


FIGURE 1. Application architectures: Blockchains vs. Database.

6-block confirmations) on Bitcoin [6] and around 3-minutes (15-second block interval with 12-block confirmation) on Ethereum [7].

III. BLOCKCHAIN ARCHITECTURE AS A DATA STORE

In Fig. 1, we define a conceptual architecture of a software system, detailing a blockchain as its data store layer [5]. On the right, we show a conventional database to highlight our interpretation on how a blockchain data store can be explained from the conventional view of a database-backed application architecture.

Broadly, three different types of applications utilise blockchain technology at its core, namely, currency (e.g., Bitcoin and micropayments), contracts (e.g., escrow and automated insurance process based on agreed terms), and asset management (e.g., land registry and digital coupons). Just like in a conventional database-backed application, the conceptual data model underpinning a blockchain-based application needs to be mapped to the logical and physical levels of the data store to persist.

In the following, we present our view of a blockchain data store as four layers, namely logical data store, physical data store, data access, and data processing layer. It is noted that while the layers are stacked in the order seen in Fig. 1, we present the layers in a different order for the sake of simplifying the discussion.

A. LOGICAL DATA LAYER

From a database developer's viewpoint, this level concretises a conceptual model of the data to a materialised form such as relational tables so that the application can interact with the data store (e.g., issuing queries to a database). It is a well-defined area of programming in the conventional database applications, and most programming constructs support this layer in a standardised way (e.g., SQL over relational tables with JDBC in Java).

Here, we examine how this concept applies to the blockchain environment. The issue of accessing the data store via queries is discussed separately. This section focuses on what we consider the "logical models" of blockchain-based applications. Primarily, two constructs are visible to the database developer at this layer; assets and smart contracts.

1) ASSETS

Assets include both digital assets like cryptocurrency and digitalised traditional assets such as stocks or titles, of which blockchains track the ownership. They are also referred to as *states* in many systems, as the primary concept of an asset is to track any piece of information beyond ownership (e.g., attributes and reputation of a physical object). Blockchains represent assets in two ways:

- *UTXO (Unspent Transaction Output)* is an asset represented as an output of a transaction and bound to an account. An UTXO can be spent once as an input in a new transaction. The balance of an account is calculated as the sum of all UTXOs (i.e., transactions with unspent outputs) associated with the account. Bitcoin, R3 Corda,[1] and QTUM[2] use a UTXO-based model.
- *Account-balance model* maintains a separate entry for the asset balance of each account. The balances of all accounts are traced as the *global state* of a blockchain network. Ethereum, Hyperledger,[3] ESO,[4] NEO,[5] Ripple,[6] and Stellar[7] are based on this model.

The UTXO model enables parallel transactions and better privacy as they are stateless. However, they can be fragmented, which increases computational, storage, and programming complexity. Alternatively, the account-balance model provides an abstract view of an account, bulk transactions, as well as reduced computational, storage, and programming complexity as they are stateful. As might be expected, the statefulness of this model limits concurrent transactions and privacy.

2) SMART CONTRACTS

Another primary construct at this layer is a smart contract. A *smart contract* is a set of executable instructions that are activated in response to a message. When executing, these instructions may change the assets and generate new messages. In first-generation blockchains, like Bitcoin, a simplified form of smart contracts can be embedded into a transaction as an executable script. In second-generation blockchains, like Ethereum, smart contracts facilitate storing and manipulating data on the blockchain. Compared to stored procedures in databases, smart contracts ensure that data they embed can only be manipulated by calling the approved functions. Therefore, smart contracts can be considered as "data with rules".

Remarks: An *account* (aka., address) is a unique reference (i.e., key) to an asset or a smart contract, e.g., owner of

[1] http://www.corda.net
[2] https://qtum.org/en
[3] https://www.hyperledger.org
[4] https://eos.io
[5] https://neo.org
[6] https://ripple.com
[7] https://www.stellar.org

an UTXO in Bitcoin or balance of an Ethereum account. Therefore, in blockchain, the logical data store layer can be abstracted as a *key-value* store that keeps track of accounts and their assets or smart contracts. This is similar to how a NoSQL database stores its data at this level. Depending on the blockchain platform, the *key* is an account and the *value* can be of anything from a simple data structure, object, to a JSON/XML document representing an asset or a smart contract. Thus, we can state that blockchains have a schema-less key-value or document store at its logical layer.

While key-value or document stores have emerged as the preferred data store for highly-scalable applications, some level of explicit modelling at this level is needed to manage the data in many applications correctly. In fact, the lack of appropriate tools to model data with rules have been identified as a gap in developing and engineering blockchain-based applications [8]. So far, conventional modelling languages have been used to model blockchain-based applications. For example, UML class diagrams are adopted to model smart contracts as the contract languages tend to follow the object-oriented paradigm [9], [10], e.g., Solidity in Ethereum and JavaScript in Hyperledger. Sequence diagrams are used to model the system behaviour, where smart contracts are explicitly modelled as roles. In terms of extending the existing modelling languages for blockchain, a model-driven engineering tool Lorikeet [11] extends BPMN (Business Process Model and Notation) to model smart contracts as a data store, as well as the business process itself as a set of smart contracts. Although these early works are valuable starting points, it is essential to extend these works to capture unique aspects of blockchains.

The complexity of supported data types and the extent of their manipulation are limited by the design of the smart contract language. For example, *tokens* are assets that are embedded within a smart contract. In Ethereum and Hyperledger, a user-defined schema can also be enforced. While a JSON object or CSV file added as an asset could be emulated as a set of tables within a smart contract to overcome schema-less nature of blockchains, it is important that smart contracts are not over-engineered such that their cost-efficiency and security are lost.

B. PHYSICAL DATA LAYER

A conventional view of a physical data store would involve understanding different index structures (e.g., B-tree and Hash table) which are highly optimised for searching and retrieving data items. In this section, we examine in what physical forms the blockchain data are represented and their implications on reading and writing.

As shown in Fig. 1, we see the data at this level in three tiers, namely transactions, blocks, and the ledger. A selected set of valid transactions form a *block*, while a set of blocks that satisfy the consensus protocol is included in the *ledger*. The term *transaction* in blockchain can mean different things depending on the context. It could refer to the operation that manipulates the data stored on a blockchain, as well as the data structure that stores the parameters used by the operation. To differentiate the two usages, here we use the term *transaction record* to mean the data structure. We reserve the term transaction itself for Sections III-C and III-D where we discuss data access and processing operations.

1) TRANSACTION RECORDS

A transaction record holds both the parameters and results of ''blockchain data operations'' performed on the assets and smart contracts (i.e., the logical constructs from the previous layer). Typical operations are creating new accounts, exchanging assets, or creating/executing smart contracts.

An essential characteristic of a transaction record is, once chosen to be included in a block, the record is permanently stored in the block achieving immutability. Most blockchains also persistently store the failed transactions. This is because of blockchain's roots to the financial sector, where every data record in blockchain is a financial transaction requiring utmost transparency. As the blockchain transaction records are immutable, the only way to correct any mistakes is to issue a reverse transaction. Each transaction record has a unique identifier and stored as a key-value pair within a block.

2) BLOCK

Every block contains a list of transaction records (may be empty if blocks are built periodically). Therefore, the exact content and structure of a block are affected by that of transaction records it contains and implementation of the blockchain. For example, the transaction records in a Bitcoin or Hyperledger block are structured in a Merkle tree, whereas a Trie is used in an Ethereum block [7]. Moreover, a block can maintain other data structures, such as the global state. For example, an Ethereum block keeps track of all account-balance pairs of assets and smart contracts using another Tries. In Hyperledger, a key-value store (e.g., LevelDB or CouchDB) is used to keep track of the global state.

The block size is a configurable parameter and is subject to a trade-off between speed of block data replication, inter-block generation time, and transaction throughput [4]. The block size could be specified in several ways. For example, Bitcoin specifies a limit on data (in MB) while Ethereum specifies a limit on computation (as gas limit) per block.

3) LEDGER

Blockchain is a single global list (chain) of blocks, where each block is ''chained'' back to the previous block through the inclusion of the hash of a representation of the previous block's data. Bitcoin, Ethereum, and Hyperledger are well-known examples of such a chain of blocks. Alternatively, Hashgraph[8] uses a Directed Acyclic Graph (DAG) of blocks. The ledger of IOTA[9] is a DAG of individual transactions rather than blocks.

[8]https://www.hedera.com
[9]https://www.iota.org

Remarks: The physical forms of blockchain data is inter-connected in the three tiers explained above. As pointed out, the internal organisation and data structures of these tiers depend on the implementation of a particular blockchain platform. Regardless of the differences in block implementations, data storage models in blockchains are rather limited and optimised for storage, rather than for searching and indexing, unlike the conventional counterparts. This is because the data storage is implemented to guarantee the unique properties of blockchain, reduce data storage and transmission costs, and to support financial transactions.

Most blockchain ledgers such as Bitcoin and Ethereum are fully replicated where assets, transactions, smart contracts, and blocks are duplicated on every node in the blockchain network. Whereas Hyperledger replicates only to all the nodes in a *channel*, which is a logical subset of nodes in the blockchain network that is allowed to access each other's data. Such high-level of replication enhances immutability as any change to data on a small fraction of nodes cannot affect the data on other nodes without going through the consensus process. However, contrary to distributed databases, such replication does not increase transaction throughput nor reduce latency. This is because of the consensus process that attempts to enhance the consistency of data by preferably electing one node as the miner to build the next block, and then replicating it to all other nodes. Alternatively, blockchains such as R3 Corda and BigchainDB[10] shard the ledger (stored as a database) across a set of nodes to provide better throughput and latency characteristics. A similar effect is expected in Ethereum 2.0 when it implements sharding.

C. DATA ACCESS LAYER

In this section, we examine the API-level access to the data store. As depicted in Fig. 1, between the application and the data store, the conventional data access mechanism typically wraps around SQL (Structured Query Language) statements to issue data read/write operations, and the practice of managing the CRUD (Create, Read, Update, and Delete) operators is well established.

1) CREATE AND UPDATE

In the CRUD-centric view of data access, transactions support only the create and update operators. For example, a transaction can change the ownership of a title or debit cryptocurrency from one account and credit to another. A transaction may also be used to deploy and initiate the execution of a smart contract. Some blockchains further distinguish transactions used to manipulate accounts and assets from smart contracts, e.g., Ethereum refers to them as transactions and messages.

2) DELETE

None of the blockchain solutions explicitly support the delete operator to ensure immutability. However, a transaction could be used to set an asset to a null value or change a state to an unusable state. Similarly, assets created or embedded in a smart contract can be distorted by calling the appropriate smart contract function via a transaction. While this could emulate the behaviour of a delete operator, all changes are recorded on the blockchain.

3) READ

Compared to databases, reading blockchain data is not straightforward. For example, as blockchain transactions use a receipt-based transient synchronous communication, they do not directly return results or indicate whether the transaction is executed. While smart contract data can be queried within a smart contract function, such functions also do not return a result due to the same reason. Similarly, we cannot issue read requests to a blockchain. Instead, we have to passively access first-class data elements (assets, accounts, transactions, smart contracts, and blocks) using unique identifiers (IDs). A tool used for such querying is referred to as the *blockchain explorer*. A blockchain explorer connects to one or more nodes that store recently generated blocks or full ledger through an application called the blockchain *client*. Blockchain client sequentially goes through the ledger, starting from the most recent block, looking for the given asset, account, transaction, or smart contract ID. Therefore, explicit querying is required even to check whether a transaction is accepted, rejected, included, or confirmed.

Remarks: Providing more efficient data access to the application layer is a critical component of blockchain-based systems, and there are on-going efforts in this area. For example, to support faster and more complicated queries, many blockchain explorers, such as Etherscan,[11] copy the blockchain data to a centralised indexing server. Hyperledger Fabric maintains a purpose-built index to provide a fast ID and time-based querying of first-class data elements.

Ethereum Query Language (EQL) is an SQL-like query language which aims to provide a general-purpose query/answer implementation for blockchain data. It allows queries to quickly extract information scattered through several records in the blockchain using collections of blocks, types of objects (e.g., transactions and accounts) and a binary search tree as its core language concepts [12].

Libraries such as Ethereum web3[12] and Hyperledger fabric-network[13] hide complexities by providing an asynchronous API to both issue transactions and query their results via a client. R3 Corda's ledger data are maintained in a relational database to enable both read and write queries using SQL. BigchainDB is an alternative design where a NoSQL query language is used to both read and write blockchain data.

D. DATA PROCESSING LAYER

In this section, we present an operational view of blockchains as a data store. Conceptually, we highlight the data processing

[10] https://www.bigchaindb.com

[11] https://etherscan.io

[12] https://web3js.readthedocs.io

[13] https://fabric-sdk-node.github.io

TABLE 1. ACID properties of blockchain transactions.

Property	Description	Compliance	Remarks
Atomicity	Transaction commits or fails entirety (i.e., all-or-nothing)	Comply	Each node independently executes each transaction and smart contract based on its ledger state. Only the successful transactions and smart contracts are then included in a block. If it fails midway (e.g., due to insufficient assets or smart contract exceeding the set fee), all asset changes are rolled back, and the transaction is rejected.
Consistency	Committed transactions are visible to all future transactions	Depend on consensus protocol	Consistency could be temporally violated in Nakamoto-consensus-based blockchains [6] (e.g., Bitcoin and Ethereum) which may temporarily fork multiple chains of blocks. Therefore, different transactions that double-spend the same asset may co-exist in forked chains temporally violating the consistency. Once the longest chain emerges, only the transaction in that chain will be confirmed. A confirmed transaction is valid and visible to all future transactions. Transactions in all other candidate chains are rejected and have to restart the process to get included in a future block.
Isolation	Uncommitted transactions are isolated from each other	Comply	Miners execute transactions sequentially. Also, a minor can see only the assets of blocks generated by the parent block and its predecessors. Consequently, any asset changes due to a transaction are visible only to the future transactions along the same chain of blocks. Therefore, concurrent transactions in the same chain are processed sequentially, while transactions in forked chains are isolated from each other. Hence, any transaction rejected in the future (due to Nakamoto and DPOS consensus) can revert their assets.
Durability	Once a transaction is committed, it is permanent	Depend on consensus protocol	Due to Nakamoto and DPOS consensus, a transaction included in a block and believed to be confirmed could be rejected later.

mechanisms by which the blockchain guarantees and maintains data consistency and durability (i.e., immutability in blockchain terms), which are critical functions of any data store. A similar concept in conventional databases would be concurrency control methods such as lock-based transaction management and recovery strategies. However, the goals and outcomes of the data processing layer in blockchain systems are very different from that of conventional databases. In a relational database, for example, the data processing strategies are designed to guarantee the ACID (Atomicity, Consistency, Isolation, and Durability) properties on transactions [13], [14]. On the other hand, data operations in a blockchain are designed to provide the blockchain-specific data qualities, namely transparency, immutability, and consistency. We believe understanding these differences will strongly underpin correct decision making in designing blockchain-based applications. Therefore, in the following, we describe how ACID properties of blockchain transactions are affected by the consensus mechanism as the primary data operation/processing layer construct.

As seen in Table 1, while a blockchain transaction can concurrently act on multiple assets, it does not fully support ACID properties. A blockchain miner will include only a successful transaction or smart contract deployment/invocation in a block. If it fails midway, all asset changes are rolled back, and the transaction is rejected. Hence, transactions within a blockchain satisfy the *atomicity* property. Moreover, blockchain platforms such as Stellar and IOTA support bundling of transactions to a single large transaction, which is handled atomically. Similar behaviour could be achieved via smart contracts where they could act on multiple assets or initiate other smart contracts within a single function call. However, depending on the implementation of the smart contract execution environment, transactions involving multiple smart contract calls may not exhibit atomic behaviour, e.g., reentrancy is a known issue in Ethereum.

As the blockchain's primary objective is to keep each asset consistent through consensus, it also satisfies the *consistency* property. However, consistency could be temporally violated in Nakamoto-consensus-based blockchains [6] when multiple chains of blocks with conflicting transactions may exist until the longest one emerge. This behaviour is similar to *eventual consistency* in NoSQL databases. However, as only one of the concurrent transactions will be included in the blockchain after finality, it does not necessarily establish a total order of all attempted transactions with time. Finality (i.e., time to declare a block as confirmed) in blockchains such as Hyperledger, Ripple, NEO, and BigchainDB is immediate, as they use Byzantine Fault Tolerance (BFT) for consensus. Whereas Delegated Proof of Stake (DPOS) based blockchains such as EOS has relatively fast finality compared to Nakamoto consensus. It is further important to understand that the finality in both Nakamoto and DPOS consensus is probabilistic and there is a non-zero probability that even the longest chain could be overtaken in the future (e.g., 51% attack).

While building a block, concurrent transactions in the same block are processed sequentially, and the transactions in forked chains are isolated from each other. This ensures *isolation* property, as any transaction rejected in the future can revert their assets. However, blockchain does not provide any guarantees about the order of transactions added to the

blocks (transactions with lower fees may never be included in a block) nor which ones will be included in the longest chain. Therefore, the confirmed transaction schedule may not be any of the legal schedules of concurrent transactions submitted to the blockchain. To prevent any application-level issues due to this behaviour, only sequential transactions are allowed in blockchains such as Bitcoin, where a second transaction related to an asset can be submitted only when the first transaction is included in a block. Whereas blockchains such as Ethereum, R3 Corda, and Stellar allow concurrent transactions given that the application sets a sequence number to specify the desired order of execution. However, if a transaction fails, all subsequent transactions are blocked until the failed one is resubmitted and included in a block. Alternatively, Hyperledger does not enforce any order at the time of issuing transactions, and the orderer node defines the global order.

Durability property is not satisfied in Nakamoto and DPOS consensus-based blockchains as a transaction included in a block and believed to be confirmed could be later rejected. Therefore, blockchain transactions are somewhat different from database transactions, and the extent they satisfy ACID properties are blockchain platform specific.

Remarks: Transactions spanning across multiple blockchains could be handled through a transaction manager similar to that in databases. For example, cryptocurrencies could be exchanged through a centralized exchange which ensures that both the transacting parties either get respective currency or no one gets anything (i.e., *all-or-nothing* property). However, as centralisation is undesirable in blockchain, as well as to support other asset types, several inter-blockchain atomic swap protocols are proposed. The most common approach is to deploy a set of interdependent smart contracts in transacting blockchains such that they form a DAG of transactions [15]. All smart contracts in the DAG use hash-locks and time-locks to ensure that participants comply to the atomic-swap protocol. Subsequently, by releasing the secret required to calculate the hash-lock at one end of the DAG, a chain of transactions along the DAG can be initiated while ensuring all smart contracts execute in the pre-deployed order ensuring atomicity. Thus, the hash-lock acts as the transaction *commit* command. Transaction *abort* could be simulated by not releasing the hash-lock where all smart contracts will timeout at the set time-lock(s) while reverting any changes made to assets. However, it is difficult to set an appropriate time-lock in practice, as a subset of the smart contracts could prematurely timeout while their commit functions may await long time to be included in a block or due to Nakamoto consensus. Zakhary *et al.* [16] proposed two-phase locking based protocol to get away with time-lock problem while Borkowski *et al.* [17] proposed claim-first transactions to overcome issues related to Nakamoto consensus. However, these solutions also have practical limitation such as the need for involving minors as witnesses, additional transaction fees, and reliance on external software components to provide inter-communication between blockchains. As applications using multiple blockchains start to emerge, it is essential to overcome these practical concerns and extend these solutions to work distributedly across more complex assets and transactions. Moreover, initiation of truly concurrent transactions and smart contracts is essential to enhance the throughput of an application.

IV. PLACEMENT AND CONFIGURATION OF DATA IN BLOCKCHAINS

In this section, we present some of the common data architectures for applications that incorporate blockchains. In addition to logical and physical layer differences, contemporary blockchains have limited performance [1], and typically charge a fee to store and manipulate data. Thus, designing a fully blockchain-based application or an application that partially relies on a blockchain requires a thorough evaluation in terms of the decisions relating to what data a blockchain would store and where it would be placed within the software architecture [4].

Understanding the impact of each design choice (e.g., cost of transaction processing fees) would mean developers and users can make informed decisions about what data to keep on the blockchain, and explore the options of storing part of the data on conventional data stores and storing only the summarised data or their attestations on the blockchain. Hence, there is a need to choose and implement a suitable data architecture as per the needs of the specific application.

In Fig. 2, we show a representative data architecture of a blockchain-based application, depicting the flow of data between the blockchain networks and external environments. Typically, the application data will flow from external data sources (aka., *off-chain*, shown in the right-hand side of the figure) to a blockchain data store (aka., *on-chain*, shown in the left-hand side of the figure). While raw data (e.g., IoT sensors) originating from an external system could directly be brought into the blockchain, in practice, most data will go through some processing steps in an external storage before it is transferred to on-chain in the form of metadata, summaries, or attestations (typically as a hash of data).

Because on-chain options could present limitations in storage capacity and performance, as well as privacy and cost concerns, most blockchain-based applications with high throughput and storage requirements would need to consider an on-chain/off-chain hybrid option similar to what is shown in the figure. A data architecture as such will result in high confidentiality (by not exposing raw data) and support a high volume of data in real-time (by externally processing the data).

Blockchains cannot directly access the data outside of the network. However, to make deterministic choices, blockchain-based applications need some external information (e.g., a real-time weather condition). An *oracle* is a data feeder that connects an external system (or data source) to the blockchain network. Oracles typically have both on-chain and off-chain components. Either a human or a server could play the role of an oracle and submit data (from an external

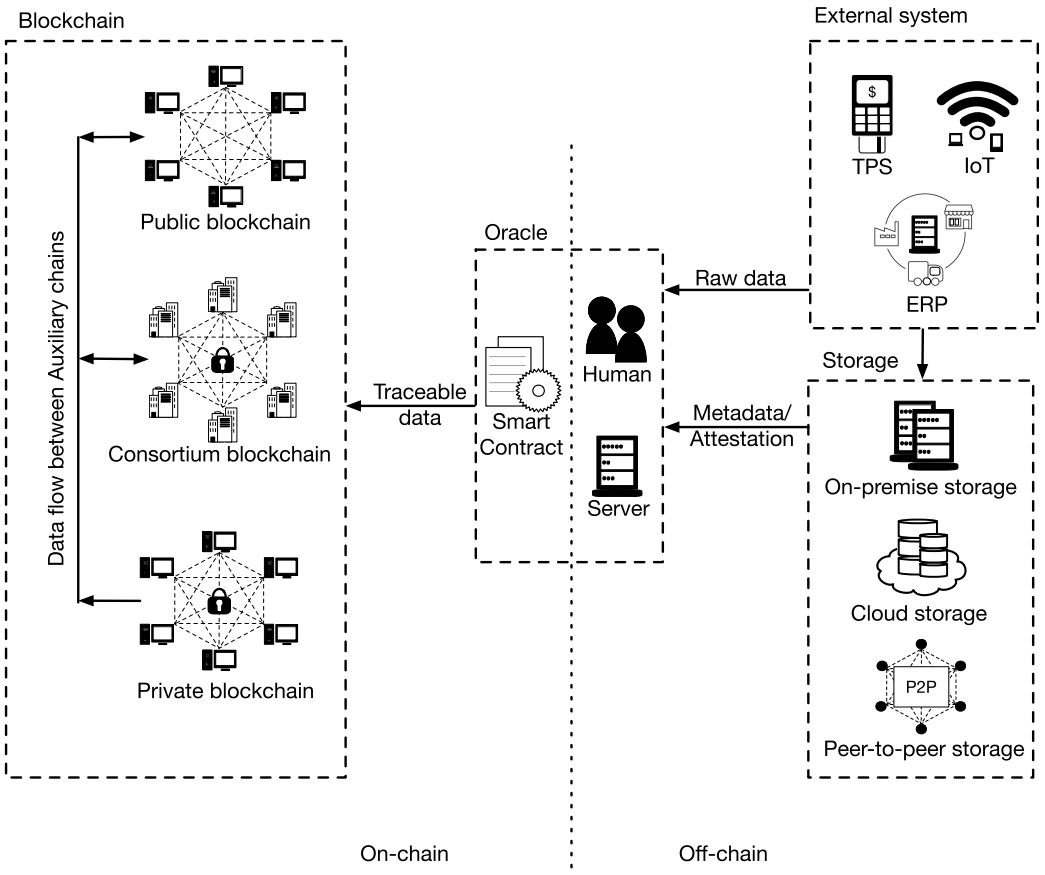


FIGURE 2. A blockchain data architecture and data flow.

system or off-chain storage) to a blockchain as a set of transactions. An oracle may also rely on a smart contract to enhance the quality of posted data by validating them or adding metadata such as reliability/reputation of the data source.

The on-chain data store may consist of multiple blockchains similar to an application using multiple databases with different features, performances, and security settings. Therefore, even within the on-chain data store, data may also flow between multiple blockchains referred to as *auxiliary chains*. For example, a private blockchain's Merkle tree could be anchored on either a public blockchain or consortium blockchain to leverage the provided security without sacrificing data privacy.

In the following, we further examine two broad architectural design choices for storing data, highlighting the main characteristics and trade-offs in each option. In particular, we will discuss the performance, cost, and privacy as the main concerns.

A. ON-CHAIN DATA STORE

In terms of storing the data on-chain, following four possibilities can be considered:

1) USING PUBLIC BLOCKCHAIN

To fully leverage transparency, immutability, and consistency, an application can utilise a public blockchain [18]. If an application uses a public blockchain as is sole data store option, all application data would be recorded and duplicated on all blockchain nodes. This means, besides the properties mentioned above, the application data will also have high availability. However, the trade-offs are the lack of privacy and confidentiality, as any node can openly participate in the network at any time. Low transaction throughput and storage scalability, as well as processing fees, are the other downsides of using public blockchains.

2) USING PRIVATE BLOCKCHAIN

As only the authorised nodes can participate in a private blockchain network, it provides confidentiality. Due to some level of trust authorisation brings in, simplified/efficient consensus processes can be used in private blockchains to reduce the time to accept and confirm transactions with large data payloads. Consequently, private blockchains are typically designed and configured to have better performance and scalability than the public ones [18]. However, having only authorised nodes as participants necessitates pre-existing trust amongst the nodes and centralisation.

3) USING CONSORTIUM BLOCKCHAIN

Similar to private blockchains, consortium blockchains are also designed for higher performance, scalability, and confidentiality than their public counterpart. The main difference between private and consortium blockchains is that a private

blockchain is generally deployed within a single organisation while a consortium blockchain can involve multiple organisations who interact with each other. For example, a channel in Hyperledger can be created for two or more organisations in the same consortium blockchain to allow confidential communications between them. It is noted, however, similarly to the private option, any performance gained in the consortium blockchains come at the cost of having a higher level of centralisation and pre-established trust amongst the participants.

4) USING AUXILIARY CHAINS

In this option, more than one blockchain is used in a larger software system [18] to build ''connected'' data stores. The motivation is either to leverage the security property of an exiting public blockchain or to improve the scalability of the overall system. Highly sensitive data of an organisation can be stored on a private blockchain owned by the organisation, while data essential for business interactions between multiple entities could be stored on a consortium blockchain. Data from both a private and a consortium blockchain could be anchored on a public blockchain to leverage the public blockchain's stronger transparency and immutability properties. For example, state channels like Lighting Network[14] on Bitcoin and Raiden[15] on Ethereum move intermediate transactions to off-chain storage and only keep the finalised transactions on the public blockchain. Cosmos[16] Inter-Blockchain Communication (IBC) protocol aims to connect different blockchain platforms through the use of an independent consensus protocol. These different layers of data sharing enable performance and flexibility in terms of data confidentiality configurations, while at the same time making use of the stronger properties provided by public blockchains.

B. COMBINING ON-CHAIN AND OFF-CHAIN DATA STORES

Generally, most applications rely on data from one or more external systems. Therefore, a more realistic architectural design of a blockchain-based application would place data on both on-chain and off-chain. Combining on-chain and off-chain data is commonly found in large software systems with a blockchain [19], [20]. Such a hybrid-design could also overcome scalability issues resulting from full data replication and limited computational capacity of blockchains.

As illustrated in Fig. 2, one architectural strategy for separating data is to keep only the application's metadata or their hashes on-chain while keeping the actual data off-chain (e.g., on-premise, cloud, or Peer-to-Peer (P2P) storage). By keeping highly sensitive, real-time data in stakeholders' off-chain storage, this configuration could provide confidentiality and better scalability while still achieving the immutability provided by the blockchain. The hash of the data would be used to verify the integrity of the data stored off-chain.

Although off-chain storage mitigates the issues of scalability and privacy to a certain extent [21], it introduces new challenges such as data aggregation and integration. Having parts of the data in different places introduce the need for a mechanism to combine data from different places seamlessly. Conventional data management uses data warehousing technologies to integrate data from multiple sources into a central repository for business intelligence analysis and reporting.

Remarks: According to ''State of the DApps'',[17] there are over 2,250 registered Decentralized Applications (DApps) as of September 2019. A majority of these DApps use a combination of traditional storage and oracles. In those configurations, the traditional data storage options such as on-premise, cloud, and P2P storage are still the primary data store for the applications, and the blockchain data store is complementary to the conventional data storage. To further reduce the size of the on-chain data, a more complex hash pointer-based data structure has been used to represent off-chain data that are of high volume, velocity, and variety [22]. An oracle is required to push the hashes of the data to the blockchain. Within the on-chain configuration, an auxiliary chain could be used to build a storage hierarchy as per the performance, cost, and privacy trade-offs.

One of the main challenges of using blockchains as a storage element is that a blockchain's logical and physical layers comprise of different types of data structures compared to traditional databases. Hence, the format of data between the blockchain and a conventional database is different; hence, an extra effort is typically required to resolve the differences.

V. BLOCKCHAIN DATA ADMINISTRATION

Next, we focus on the operational aspects of a blockchain to ensure it is deployed and maintained to achieve the desired performance, security, privacy, and reliability goals of the application. Blockchain administration is a new domain, and best practices are yet to be identified. While a blockchain includes a unique set of configuration and management challenges, we believe these tasks could well be introduced into the job role of a database administrator. Next, we discuss the deployment and maintenance aspects of a blockchain instance.

A. DEPLOYMENT

The extent of involvement of a database administrator in managing a blockchain depends on the mode of hosting and the chosen blockchain platform. If the blockchain is public, the administrator's key role would be to provide connectivity to the blockchain and an explorer either by installing a blockchain client or via a public API service. Services such as Infura[18] and Ethercluster[19] provide API authentication and pay-as-you-go model to reach public blockchains. If latency, cost, and privacy are important design considerations, it is advisable to install own blockchain client close to

[14]https://lightning.network
[15]https://raiden.network
[16]https://cosmos.network
[17]https://www.stateofthedapps.com/dapps
[18]https://infura.io
[19]https://www.ethercluster.com

the application. To achieve high read throughput and availability, the administrator should install multiple instances of the blockchain client and provide sufficient bandwidth, storage, memory, and CPU power for it to sync and keep up with the public blockchain. However, it is optional to be involved in the mining process. If involved in mining, significant computing and memory resources or stake need to be allocated.

If a consortium or private blockchain is chosen, it could be installed either on-premise or in the cloud. Enterprise server vendors are already offering specialised high-end servers to run blockchains such as Ethereum, Hyperledger Fabric, and R3 Corda. Alternatively, one could deploy a blockchain as an IaaS (Infrastructure as a Service) cloud instance using pre-built templates or use a managed Blockchain as a Service (BaaS) instance. Major cloud vendors are already offering both options. In all cases, the administrator needs to set parameters such as desired inter-block time (specify either as a time value or difficulty in proof of work), access control, and may even need to build the first block of the blockchain (aka., genesis block). Blockchain platforms such as Hyperledger also requires choosing one of the supported consensus protocols, as well as the configuration of specialized nodes such as orderers and Certificate Authorities (CAs). In the case of on-premise and IaaS based blockchains, the administrator also needs to provision suitable hardware, storage, and bandwidth. Blockchains such as R3 Corda further requires configuration of sharding. However, it is essential to note that while most blockchains are somewhat vertically scalable (due to sequential execution and ordering of transactions) [23], [24], they are not horizontally scalable unless sharding is used.

If the blockchain-based application relies on off-chain solutions such as IPFS[20] and oracles connectivity and storage allocation for such services are also needed. While immutability of a blockchain essentially makes it a data warehouse, data from the blockchain client/explorer may also need to be connected to the data-pipeline of the organisation to support data analytics.

Remarks: It has been shown that the choice of blockchain platform and its parameters have a significant impact on the performance in addition to the mode of hosting, hardware configuration, and workload [23]–[25]. Therefore, it is desirable to benchmark and fine tune the blockchain instance before production use. BLOCKBENCH [23] is proposed as a tool to benchmark private instances of Ethereum, Parity,[21] and Hyperledger Fabric blockchains. However, more work is needed in the areas of understanding blockchain workload characteristics, generating representative workloads, and extending established performance testing tools to support blockchain-based application testing.

B. MAINTENANCE

When on-premise and IaaS blockchain instances are operational, the administrator needs to monitor both the blockchain performance (e.g., throughput and latency) and resource utilisation to fine-tune the parameters such as mempool size, block size, and hash rate. As the ledger is fully replicated across multiple nodes (or all nodes in a shard), explicit backing up of data is not needed, unless the blockchain is configured to prune old blocks [6] to enhance performance, storage, and cost efficiency. The administrator may also need to initiate software updates as soft forks to fix vulnerabilities in the blockchain platform, enhance performance, and benefit from new features. In rare cases, hard forks are required to introduce significant functional changes, correct data errors, or recover after a cyber attack.

Remarks: With the rapid evolution of technological, economic, and regulatory landscapes, contemporary blockchains are all but certain to undergo significant changes. In fact, Gartner predicts "through 2021, 90% of the enterprise blockchain implementations will require replacement within 18 months to remain competitive and secure, and to avoid obsolescence" [26]. Therefore, it is quite likely that the administrator needs to eventually migrate the blockchain to an upgraded version of the same blockchain platform, a different platform, or different mode of hosting. However, it is difficult, costly, and risky to change the blockchain due to the incompatibilities in platforms and smart contract languages, mode of hosting, and blockchain properties such as transparency, immutability, and consistency. In [27], using a migration pattern-based analysis, we showed that while migrating to a private or consortium blockchain could be achieved relatively easily, recreating full blockchain history on an existing public blockchain is impractical. Nevertheless, the global state can still be recreated, which is sufficient for most practical migration scenarios. Therefore, it is essential to choose an appropriate level of data abstraction that balances competing factors such as the performance, cost, time, granularity of data, transparency, privacy, and risk. Moreover, proactive data designs such as the use of simplified data structures, not tightly embedding data into a smart contract, and use of smart contract registries are recommended. However, identifying the suitable level of data abstraction for a given application scenario, confirming the correctness of translated smart contracts, and handling private keys are among the challenges to be addressed.

VI. DATA ANALYTICS

The real value of data can be utilised only when they are analysed to derive actionable insights. In this regard, to many modern organisations with fast-growing data, being equipped with effective data analytics capabilities is vital to make better decisions and stay competitive.

Blockchain technology plays two distinct roles in the data analytics field. First, the data stored in a blockchain and blockchain network themselves provide a rich source of information (e.g., interaction patterns between accounts). Second, blockchains can enable trusted data analytic environments for multi-party data sharing by adding the element of assurance into data and derived analytic models. This could facilitate

[20] https://ipfs.io
[21] https://www.parity.io

the types of analytic models that may not have been feasible otherwise due to distrust in multiple data stakeholders. Therefore, in this section, we discuss these two aspects, respectively. We present some of the common analytics techniques developed over the blockchain network data and data stored on blockchains. We then discuss how a blockchain can become an ''analytic facilitator'' component in data analytics architectures.

A. ANALYSING BLOCKCHAIN DATA

The data generated by the blockchain network can be collected to visualise and analyse the historical and current system behaviour. Such visualisation and analysis are needed by the system operators/administrators to understand the system behaviour at network layer and transactions in the network, as well as operate the blockchain-based system accordingly.

In the following, we broadly categorise the blockchain data analysis techniques into two; visualisation and data mining.

1) DATA VISUALISATION

In the context of a blockchain-based system, data visualisation is essential to illustrate the behaviour of the system because a blockchain is highly dynamic with nodes connecting and disconnecting all the time. With an overall visualisation of the system, it is easier to understand the blockchain system. Other types of data are being generated in the blockchain ecosystem, such as the price and market share of cryptocurrency, which reflects the economic value of the blockchain. Such data are also crucial for the developers to select an appropriate blockchain platform and analyse the cost of blockchain-based applications. It is also useful for crypto-investors to understand the market trends and to detect fraud and market manipulation behaviours.

Several online tools visualise different aspects of blockchains. For example, Bitnodes[22] and Ethernodes [23] visualise the geographical distribution of Bitcoin and Etheruem networks, respectively. Dailyblockchain[24] and interaqt[25] visualise transactions on Bitcoin. The former shows how the transactions are dynamically linked with each other via outputs and inputs. The latter visualises the monetary size of the transaction without linkage between the transactions. BitInfoCharts[26] visualises accumulated transaction numbers, price, and difficulty of different blockchains/cryptocurrencies in real-time.

TxStreet[27] is an interesting visualisation of Bitcoin that use an analogy of passengers (as transactions) and busses (as blocks) to show different types of transactions submitted and included in Bitcoin and Bitcoin Cash networks. Etherview[28] is a relatively comprehensive visualizer of Etherum that covers the block interval time and chain forks, as well as different types of transactions, namely cryptocurrency transfers, smart contract creation, and smart contract invocation.

2) DATA MINING

The immutable data stored on a blockchain is a public data set accessible to anyone within the network; hence, can be used to analyse the historical behaviour of the system. Besides the visualisation techniques introduced earlier, the data stored on public blockchains, like Bitcoin and Ethereum, have been mined for different purposes. Analysing the data on a blockchain and the data in the blockchain network (e.g., *mempool* data, i.e., transactions awaiting to be included in a block) is important for developers to decide when to submit transactions and how much transaction fee to pay. It is also useful for regulatory bodies and law enforcement to detect illegal sales of items and market manipulations.

BlockSci [28] is a general blockchain analysis platform that uses an in-memory database. This makes BlockSci much faster than existing tools that utilise transactional databases. Such a design decision is made based on the append-only feature of the blockchain, and the fact that the snapshots required for research are static. BlockSci includes *mempool* data, which is also an important data source for analytics. For example, a node can decide what level of transaction fees to set, as higher transaction fees typically lead to faster inclusion in a block. BlockSci supports queries with complex conditions. However, blockchains with smart contracts, like Ethereum, are out of the scope of BlockSci.

Many empirical studies and analytics have been carried out on the data stored on public blockchains. The historical transactions of Bitcoin and many other cryptocurrencies using the UTXO model are essentially a graph-based data structure. Thus, such empirical studies usually start with building a graph of all blockchain addresses as vertices and their transactions as edges by parsing and processing raw blockchain data [29], [30]. From this, for example, a user-based transaction graph could be constructed by merging the addresses belonging to the same user.

The purpose of the blockchain data analytics varies, e.g., some are interested in understanding privacy risks in blockchain, while others are focused on discovering the statistical properties of all historical transactions. Recently, we see more studies that combine the blockchain data with an array of other data sources. In [31], [32] for instance, the data from e-commerce web sites, computer network telemetry, Google search trends, etc., are combined with the public blockchain data to track payments relating to a ransomware attack, or to show how the anonymity of a blockchain account could be compromised.

The blockchains using the account-balance model, like Ethereum, requires a fundamentally different methodology to analyse because the common usage scenarios involve smart contracts. As smart contracts have been used for general-purpose applications beyond financial transactions, execution of smart contracts, their source/byte code, and the

[22] https://bitnodes.earn.com
[23] https://www.ethernodes.org/network/1
[24] http://dailyblockchain.github.io
[25] http://bitcoin.interaqt.nl
[26] https://bitinfocharts.com/comparison/transactions-btc-ltc-nmc.html
[27] https://txstreet.com
[28] http://ethviewer.live

transactions from/to the smart contracts are all useful data sources for analysis. In [33], smart contract data are used to detect a Ponzi scheme contract. First, verified smart contracts are download along with their corresponding source code and associated transactions. A classification model is then learned based on the various features extracted from the bytecode and the transactions. The process mining framework proposed in [34] extracts process-related data from the transactions associated with a smart contract, then apply process mining techniques to create a high-level business process model to visualise the workflow.

B. BLOCKCHAIN-ENABLED DATA ANALYTICS

Blockchains offer many desirable features that creates a suitable environment for distributed data analytics. Blockchains, as a neutral platform secured by cryptographic techniques, can add trust to data analytics in a distributed environment. The anonymity provided by blockchains allows data owners to contribute data for analytics without completely losing their privacy. Immutability of blockchain enables secure and trustworthy data lineage that gives visibility in a data analytics process. Moreover, extending the idea of data with rules, smart contracts could specify permitted types of data analysis and sharing. In the following, we present two categories of the ideas where a blockchain is used as a component in a data analytics scenarios, highlighting what specific role a blockchain plays in them.

1) PROVENANCE AND LOGGING

Data provenance refers to a historical record of the data and its origins showing the trails of entities and processes that influenced data of interest. A typical form of a record will detail which data item is accessed and processed by whom and for what purpose. Properly managed data provenance can increase the integrity of data used for data analytics. In this regard, a blockchain can be utilised as a light-weight and loosely-coupled data store which offers an immutable storage of records. The records of data provenance can be stored on a blockchain as transactions, without utilising its computational capacity. The transactions become immutable records for the provenance trails. Similar ideas are applied in [35], [36], where a blockchain is used as a decentralised, access-control manager for people to declare the ownership of their personal data and take control of the data. What data are being collected and how they are accessed are stored on the blockchain as immutable transactions enabling data provenance. It is also worth mentioning that data analytic models themselves provide provenance data in that how a model is trained and updated is of interest to people who use the model. A blockchain can also store the history of the attributes relating to a model development process as provenance.

From a database management perspective, in more complex scenarios involving both on-chain and off-chain data [37], [38] (as discussed in Section III), the on-chain data can be considered as ''logs'' in conventional databases. In a conventional database, all changes (which transaction made what change to which data records) to application data are recorded in a system log. Any rollback and recovery operation needs this system log to restore the database to a consistent state. In a blockchain-based system, the on-chain data acts as a system log. However, it provides more effective detection of abnormal behaviour, as it is integrated with the actual system operations, data are transparently stored, and more fined grained transaction details are recorded.

2) A COLLABORATIVE PLATFORM

The neutral platform provided by blockchains could be used in distributed machine learning to enable collaborative model training and development [38], [39]. The design of the mining process and consensus protocol in blockchains can be applied to perform *distributed learning model training* where miners within the network who improve the performance of a shared data model could receive rewards. Such rewards incentivise more miners to join the collaborative network and contribute to the model training. Without introducing significant changes to the mining process and consensus protocol, a blockchain with smart contracts has been used to treat data as a digital asset for data analytics purposes. In such scenarios, a blockchain stores metadata of a data set, which is used as the criteria to determine the value of the data to enable data monetisation.

In some cases, the model trainer does not have to be a miner in the blockchain network. A blockchain-based incentive scheme could work with any blockchain node to reward the data and model contributors and penalise trainers that add noise to models. Such collaborative platform can be built on an existing blockchain platform without adjusting the mining process or the consensus protocol [37], [40]. In this case, smart contracts can be used to track the accuracy and manage the rewarding process of continuously updated models.

Remarks: Even though a blockchain-based data store is useful in achieving transparency and persistency, the blockchain data and its logical models are highly diverse. Besides, many blockchain-based systems employ an off-chain storage. Hence, a typical analytic model may require accessing multiple data sources that are both on-chain and off-chain, and such access may have to be continuously or periodically managed. These are some of the factors that make the analytics techniques over blockchain data still ad-hoc and challenging. Developing more systematic approaches to analyse blockchain data is an open and emerging data management issue. One of the challenges is real-time analysis, which is essential as a blockchain is a highly dynamic system, where the topology of the P2P network keeps changing in terms of the network size, geographical distribution, and computational power distribution. The ecosystem of cryptocurrencies is also complex, especially with the exchange services that act as a gateway between the physical and digital worlds. The price of cryptocurrencies fluctuates rapidly and is affected by many factors in both the ecosystem and physical world, e.g., the disclosure of

MtGox[29] or changing of financial regulations. Thus, the results derived from analysing data within a certain period might not be valid in future. Real-time analysis of the latest data can provide more insight into the long-term credibility of cryptocurrencies for crypto-investors, as well as provide clear guidance for developers to select more appropriate blockchain platforms and analyse the overall cost of using the platform.

The characteristics of a blockchain make it a suitable enabler for distributed data analytics. The tamper-proof data structure can provide data provenance, which is essential in data analytic scenarios, especially in a distributed environment, where both data and model might be stored and executed in a distributed manner, e.g., in federated machine learning. The data provenance stored in a blockchain can help achieve reproducibility of a data analysis model. Besides, the decentralised design of a blockchain fits well with the design paradigm of distributed data analytics. Blockchains can also provide a decentralised ID management infrastructure and use smart contracts to enforce decentralised access control for data sharing and analytics.

VII. GOVERNANCE

Governance refers to comprehensive control including processes, policies, and structures, which could be applied to, e.g., IT or data assets to support right decision making in organisations. As a new technology that breaks many conventional norms (e.g., removing of a central mediator), blockchains have not yet come to terms with many areas of governance. Take cryptocurrencies, for instance. They are essentially ''tokens'' issued within a blockchain platform (e.g., BTC on Bitcoin and ETH on Ethereum) and as of September 2019,[30] there are more than 2,500 cryptocurrencies in operation. Most of the transactions on these cryptocurrency-enabled blockchains are financial; hence, have corresponding financial regulation issues. It is debatable whether cryptocurrency can be considered as cash or cash equivalent because it lacks broad acceptance as a means of value exchange. It may also not be considered a financial asset because there is no contract between the holder of cryptocurrency and another partner. Thus, none of the existing standards applies to cryptocurrencies, according to the International Accounting Standard Board (IASB).[31] Nonetheless, there is an obvious need for governance on cryptocurrencies as they are currently used to pay for transaction fees (e.g., smart contract executions). It is also worth pointing out that these governance concerns are applicable to any tokenised asset that represents some form of value or equity. For example, micro-payments, loyalty programs, raffles, and benefit dispersion applications built around blockchains may need to comply with financial regulations of respective geographies. Moreover, metadata of such payments may need more controlled manipulating and storing as outlined in standards such as Payment Application Data Security Standard (PA-DSS) [41].

Notwithstanding the broader governance concerns in blockchain platforms, in this section, we focus our governance discussion around the issues relating to data management. A blockchain as data store brings up a range of governance issues, first as a unique data processing platform that removes the need for a centralised authority and second as an append-only, permanent data storage. In the following, we examine data governance concerns and challenges in blockchains, in particular regarding privacy and data quality. For the discussion to be meaningful, we first need to point out that the original design goals of blockchain technology never aimed to meet the contemporary privacy and data quality concerns being raised by the data management community. However, considering the increasing depth and breadth of the applications the technology is being considered and adopted for, we believe it is important and timely to explore to what extent the current blockchain technology as a data store satisfies the concerns and potential approaches to mitigate them.

A. PRIVACY AND LEGAL COMPLIANCE

The concept of a data-sharing ecosystem, where multiple participants interact to provide, use, and share data, is widely adopted by many organisations. However, there is a pervasive problem of the potential data breach (data abuse or misuse) in such environments due to the complicated nature of the interactions and sophisticated information diffusion schemes within the systems [42].

Recognising this problem, recently, new regulations and amendments aiming for better protection of the user information and rights of data subjects have been introduced. One of the significant schemes is the European General Data Protection Regulation (GDPR) which became law in May 2018 [43] and applies to any organisations that interact with data subjects based in the European Union (EU). The My Health Records Amendment Act 2018 by the Australian government is another example of regulatory efforts to strengthen privacy [44]. Although not comprehensive, we can derive some common and significant privacy requirements from these regulations as follows:

- *Access/Timeliness* – The data subject has the right to access and view their personal data. Also, a data subject's request for any information relating to their personal data should be responded to without undue delay.
- *Rectification* – The data subject should be able to correct inaccurate data concerning him or her.
- *Restriction of usage* – Personal data can only be processed with the data subject's consent.
- *Portability of the personal data* – The data subject has the right to receive the personal data in a structured, commonly used and machine-readable format and to transmit the data to another service.

[29] https://www.theguardian.com/technology/2014/mar/10/mtgox-bitcoin-database-leaked-online-as-hackers-crowdsource-clues

[30] https://coinmarketcap.com

[31] https://www.ifrs.org/groups/international-accounting-standards-board/

- *Right to be forgotten* – The data subject has the right of the erasure of personal data concerning him/her without undue delay.

Blockchain technology is actively promoted for inclusion in various data-sharing ecosystem architectures, citing the increased data quality and openness as a reason to trust the technology. However, there are growing concerns about whether blockchains can comply with these recent regulations, as data privacy is still an open issue for a blockchain-based system.

In terms of the requirements identified above, access (and timeliness of it) in blockchains depends on the permissions. In a public blockchain, the data subject is free to access and obtain their personal data stored on the blockchain network on time. In fact, as there is no ''privileged'' participant in a public blockchain, where it is also referred to as a permissionless blockchain. In a permissioned blockchain, however, access and timeliness could be restricted to those with the appropriate access rights in the network. Whether any inaccurate data could be corrected or not depends on the ownership of the transaction record. If the data subject owns the transaction, he/she could issue another transaction which will rectify the error. If not, rectifying an error will depend on how the owner of the transaction would respond to a rectification request. Although there is no inherent method in blockchains to impose user consent for data usage, smart contracts provide a transparent means to encode and enforce access policies. Also, a recent development such as self-sovereign identity management scheme[32] could give a sophisticated solution for ensuring ''user-controlled'' data usage. As the blockchain data is machine-readable, it satisfies portability. However, as discussed in Section V moving the data to another blockchain platform is not yet straightforward. Since the introduction of GDPR, the term right to be forgotten has received considerable attention. The records in a blockchain are immutable by design; hence, removing data to comply with this requirement is not feasible. There is a need for discussion on how to deal with the limitations to comply with privacy regulations.

In the following, we present some of the recent and relevant ideas on how blockchains could support the compliance requirements on privacy.

1) PRIVACY BY DESIGN (PBD)

The blockchain governing body as a platform provider has to ensure that the platform is designed in a privacy-friendly manner by using the Privacy by Design (PbD) approach. The approach is characterised by the key principles such as proactive/preventative privacy as the default, privacy embedded into the design, full functionality, end-to-end security, visibility and transparency, and respect for user privacy [45]. There is a strong consensus between blockchain experts and researchers that a blockchain can be compliant to PbD under certain circumstances [46]. For example, there is a distributed ledger technology that supports a pairwise ledger. It allows each user to have a separate chain and does not require full data replication across the network. This limits the distribution of data and may increase the level of privacy [47]. Such technology can be considered during the early designing phase of a blockchain-based system. It also facilitates better access control through identity management schemes such as self-sovereign identity or decentralised trusted identity (e.g., Sovrin and ShoCard) [48].

2) DECOUPLING BLOCKS AND PERSONAL DATA

As a solution to the risk of the recovery of secret information in public blockchains, a ''de-indexing'' approach similar to that of the web-based search engines is suggested. De-indexing refers to hiding a web page from the search results. It can be useful to protect the privacy-sensitive content by removing the data from the search index. Similarly, to support the right to be forgotten, blockchain platform providers can consider several potential approaches. Dori *et al.* [49] presented a process by which the removal of a transaction is supported by computing the hash of the block with the hashes of constituted transactions instead of their content. This ensures that the chain of transactions in the ledger is not broken after removing a particular transaction. Alternatively, any sensitive data can be stored outside the blockchain networks. Thus, the data can be invisible and easily removed, while preserving the consistency of the block. It is essential to consider multiple (complementary) solutions/technologies to make up for its weaknesses rather than using a sole solution.

3) IDENTITY AND KEY MANAGEMENT

Much of the personal data on blockchains are manipulated through smart contracts. A public key of a blockchain account can be considered Personally Identifiable Information (PII). From the perspective of data governance of organisations using blockchains, the content of a smart contract should be defined as per the term in the GDPR. Several technical solutions have been discussed including the level of identity (e.g., mixing keys, especially in UTXO-based blockchains), value transfer (e.g., zero-knowledge proofs/arguments), and blind signatures and data payloads (e.g., encryption and read permissions as assets) to protect PII data on a blockchain [50], [51].

B. DATA QUALITY

The value of data depends on its *quality*, which could be defined as ''the ability to satisfy the usage requirements'' [52], [53]. The data quality is often regarded as one of the key management areas in data governance [53], [54] because through the assessment and management of data quality a governing body can also correctly identify and manage data risks. Data quality can be assessed using a range of dimensions, some of which are detailed below [55]:

- *Consistency* – The degree to which data have attributes that are free from contradiction and are coherent with other data.

[32] https://sovrin.org

- *Traceability* – The degree to which data have attributes that provide an audit trail of access to data and any changes made to the data.
- *Availability* – The degree to which data have attributes that enable them to be retrieved by authorised users and applications.
- *Compliance* – The degree to which data have attributes that adhere to standards, conventions, or regulations in force and similar rules relating to data quality.
- *Confidentiality* – The degree to which data have attributes that ensure that they are only accessible and interpretable by authorised users.
- *Credibility* – The degree to which data have attributes that are regarded as true and believable by users.

The use of blockchain technologies for data sharing is a double-edged sword in terms of managing data quality. While it guarantees better consistency, traceability, and availability, it lacks support in providing compliance, confidentiality, and credibility. As a distributed database, a blockchain provides strong consistency mechanisms, reliable services, and transparency to the participants. It, therefore, increases *consistency*, *traceability*, and *availability*. In blockchain applications, *credibility* depends on the level of trust on the external data providers (e.g., oracles). As anyone can access the data stored in a public blockchain without explicit permissions, it becomes difficult for the governing body to meet the desired level of data quality when it comes to *compliance and confidentiality*.

We present two potential approaches and prior studies which address the above data quality issues as follows:

1) FINE-GRAINED ACCESS CONTROL

To improve compliance and confidentiality, a fine-grained control mechanism based on a policy granting access rights to data can be considered. Several studies have addressed this topic. Zyskind *et al.* [35] proposed a decentralised personal data management system to achieve a transparent data supply chain, data ownership, and privacy. The data provenance issue is addressed by Liang *et al.* [56]. The study proposes a provenance database and a blockchain network to provide tamper-proof records for transparency of data accountability.

2) BLOCKCHAIN ORACLE CONFIGURATION

Trust of data providers is still a critical issue in both public and private blockchains. It is directly related to data credibility, the key data quality dimension. It includes the concept of authenticity, the truthfulness of origins, attributions, and commitments [55]. Trustworthy data sources are integral parts of the blockchain networks, but it is not subject to the underlying security mechanisms of the blockchain technology. Several potential solutions have been discussed such as sourcing data from multiple oracles to mitigate the risks derived from relying on a single data source and avoid a single-point-of-failure [57].

Remarks: Setting aside the issues with the blockchain data governance, it is worth noting that the blockchain technology itself could provide solutions to general data governance problems. The unique data quality properties of blockchain technology, such as consistency, transparency, auditability, and availability, and their utility to data provenance are useful in many data governance use cases. Data stored on a blockchain can be viewed as a single source of truth (subject to the limitations of the consensus algorithm such as Nakamoto consensus and DPOS) due to these features, making it an ideal platform for building an auditing system. Blockchains could also be used to guarantee the integrity of a digital representation of a physical entity. Some of the suitable use cases are in the supply chain domain where certificates are given to food products to ensure their authenticity. The metadata about the certificates could be stored on a blockchain, and a buyer can verify the purchased product through verifying the certificates with the metadata stored on the blockchain.

As discussed earlier, there has been little attention paid to a proper governance framework for blockchain-based data sharing ecosystems in both research and industry [58]. Existing governance frameworks focus on generic information strategy or data management of organisations. While several studies have addressed a range of governance issues of data on blockchains, there is a lack of a comprehensive approach to deal with those issues and effectively orchestrate data management processes. It denotes that there is a need for a novel governance framework for both a blockchain platform and a blockchain-based application. Such a framework may play a critical role for various stakeholders. It first provides a set of guidelines for the governing body to show how data governance should be structured. It can also suggest some possible governance decisions for reducing data risks of using blockchains. The framework may encourage the desirable behaviours of the practitioners such as data stewards and custodians who will reap the benefits of compliant implementations of the governance policies. In the meantime, it is imperative to adhere to principles outlined in standards such as PA-DSS and GDPR at least as a best practice.

VIII. CONCLUSION

We are seeing the growth of blockchain applications reaching far beyond the initial craze of Bitcoin. A consequence of the fast adoption of the technology is that, in many cases, a blockchain is often used as an architectural component in a large-scale distributed software system to store data, which not only vary widely in both format and content, but also express a range of complex application domain requirements. Therefore, carefully examining blockchains to understand and assess its capabilities and issues as a data store is a timely and relevant topic to the academic and industry communities who are looking to use the technology.

To conclude, we would like to highlight some of the main lessons. First, having a clear understanding of a blockchain as a data store, and be able to comprehend and evaluate the characteristics of blockchains with regards to the conventional data stores will help application developers design and implement a blockchain-based application more effectively.

Our contributions in this regard are three folds: (i) we offered a fresh view of a blockchain as a data store, conceptualising its logical and physical layer functions compared to the conventional data stores, (ii) we analysed the various data placement options, emphasising the impact of each design option on an overall system, (iii) we showed the critical tasks and tools involved in administering/operating a blockchain as a data store. Second, if one looks beyond digital currencies, contemporary data management issues for blockchains pose both risks and opportunities. We particularly identified two categories for discussions; data analytics and data governance. Much of the focus on blockchain technology has mostly been on methodologies to develop new applications. Methods and tools for analysing blockchain data at scale, and using blockchains to enable new types of data analytics are emerging topics. Data governance is another area of importance that warrants more interests from the research and industry communities.

Admittedly, there are other topics that have not been discussed in depth but worth further research such as the efficient integration and indexing schemes designed for multiple blockchain data stores that are heterogeneous, or more detailed examination of smart contract technology and appropriate use of it in managing data. We plan to explore some of these issues in the future. For instance, our immediate future work includes looking into architectural patterns and template designs to integrate multiple blockchain data stores.

REFERENCES

[1] J. Yli-Huumo, D. Ko, S. Choi, S. Park, and K. Smolander, "Where is current research on blockchain technology? A systematic review," *PLoS ONE*, vol. 11, no. 10, 2016, Art. no. e0163477.

[2] M. J. M. Chowdhury, A. Colman, M. A. Kabir, J. Han, and P. Sarda, "Blockchain versus database: A critical analysis," in *Proc. 17th IEEE Intl. Conf. Trust, Secur. Privacy Comput. Commun., 12th IEEE Intl. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Aug. 2018, pp. 1348–1353.

[3] S. Tai, J. Eberhardt, and M. Klems, "Not ACID, not BASE, but SALT: A transaction processing perspective on blockchains," in *Proc. 7th Intl. Conf. Cloud Comput. Services Sci. (CLOSER)*, Apr. 2017.

[4] X. Xu, I. Weber, M. Staples, L. Zhu, J. Bosch, L. Bass, C. Pautasso, and P. Rimba, "A taxonomy of blockchain-based systems for architecture design," in *Proc. IEEE Intl. Conf. Softw. Archit. (ICSA)*, May 2017, pp. 243–252.

[5] X. Xu, C. Pautasso, L. Zhu, V. Gramoli, A. Ponomarev, A. B. Tran, and S. Chen, "The blockchain as a software connector," in *Proc. 13th Working IEEE/IFIP Conf. Softw. Archit.*, Apr. 2016, pp. 182–191.

[6] S. Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[7] G. Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger: Byzantium Version, Ethereum Project Yellow Paper*. Accessed: Mar. 2019. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf

[8] S. Porru, A. Pinna, M. Marchesi, and R. Tonelli, "Blockchain-oriented software engineering: Challenges and new directions," in *Proc. IEEE/ACM 39th Intl. Conf. Softw. Eng. Companion (ICSE-C)*, May 2017, pp. 169–171.

[9] H. Rocha and S. Ducasse, "Preliminary steps towards modeling blockchain oriented software," in *Proc. 1st Intl. Workshop Emerg. Trends Softw. Eng. Blockchain (WETSEB)*, May/Jun. 2018, pp. 52–57.

[10] M. Marchesi, L. Marchesi, and R. Tonelli, "An agile software engineering method to design blockchain applications," in *Proc. 14th Central Eastern Eur. Softw. Eng. Conf. Russia*, Oct. 2018, pp. 3:1–3:8.

[11] A. B. Tran, Q. Lu, and I. Weber, "Lorikeet: A model-driven engineering tool for blockchain-based business process execution and asset management," in *Proc. 16th Intl. Conf. Bus. Process Manage. (BPM)*, 2018, pp. 56–60.

[12] S. Bragagnolo, H. Rocha, M. Denker, and S. Ducasse, "Ethereum query language," in *Proc. 1st Intl. Workshop Emerg. Trends Softw. Eng. Blockchain (WETSEB)*, May 2018, pp. 1–8.

[13] T. Haerder and A. Reuter, "Principles of transaction-oriented database recovery," *ACM Comput. Surv.*, vol. 15, no. 4, pp. 287–317, Dec. 1983.

[14] S. Gilbert and N. Lynch, "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant Web services," *ACM SIGACT News*, vol. 33, no. 2, pp. 51–59, Jun. 2002.

[15] M. Herlihy, "Atomic cross-chain swaps," in *Proc. ACM Symp. Princ. Distrib. Comput.*, 2018, pp. 245–254.

[16] V. Zakhary, D. Agrawal, and A. El Abbadi, "Atomic commitment across blockchains," 2019, *arXiv:1905.02847*. [Online]. Available: https://arxiv.org/abs/1905.02847

[17] M. Borkowski, C. Ritzer, D. McDonald, and S. Schulte, "Caught in chains: Claim-first transactions for cross-blockchain asset transfers," Vienna Univ. Technol., Vienna, Austria, white paper, version 4, Feb. 2018.

[18] X. Xu, I. Weber, and M. Staples, *Architecture for Blockchain Applications*. Cham, Switzerland: Springer, 2019.

[19] D. N. Dillenberger, P. Novotny, Q. Zhang, P. Jayachandran, H. Gupta, S. Hans, D. Verma, S. Chakraborty, J. J. Thomas, M. M. Walli, R. Vaculin, and K. Sarpatwar, "Blockchain analytics and artificial intelligence," *IBM J. Res. Develop.*, vol. 63, nos. 2–3, pp. 5:1–5:14, Mar. 2019.

[20] J. Weng, J. Weng, J. Zhang, M. Li, Y. Zhang, and W. Luo, "Deepchain: Auditable and privacy-preserving deep learning with blockchain-based incentive," *IEEE Trans. Depend. Sec. Comput.*, Nov. 2019, doi: 10.1109/TDSC.2019.2952332.

[21] J. Eberhardt and J. Heiss, "Off-chaining models and approaches to off-chain computations," in *Proc. 2nd Workshop Scalable Resilient Infrastructures Distrib. Ledgers (SERIAL)*, Dec. 2018, pp. 7–12.

[22] I. Weber, Q. Lu, A. B. Tran, A. Deshmukh, M. Gorski, and M. Strazds, "A platform architecture for multi-tenant blockchain-based systems," in *Proc. IEEE Int. Conf. Softw. Archit. (ICSA)*, Mar. 2019, pp. 101–110.

[23] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "Blockbench: A framework for analyzing private blockchains," in *Proc. ACM Intl. Conf. Manage. Data*, May 2017, pp. 1085–1100.

[24] M. Schäffer, M. di Angelo, and G. Salzer, "Performance and scalability of private Ethereum blockchains," in *Proc. Blockchain Forum, 17th Int. Conf. Bus. Process Manage. (BPM)*, Sep. 2019, pp. 1085–1100.

[25] Hyperledger Performance and Scale Working Group. *Hyperledger Blockchain Performance Metrics*. Accessed: Oct. 2018. [Online]. Available: https://www.hyperledger.org/resources/publications/blockchain-performance-metrics

[26] N. Heudecker and A. Chandrasekaran. *Debunking the Top 3 Blockchain Myths for Data Management*. Accessed: Apr. 2018. [Online]. Available: https://www.gartner.com/en/documents/3871956

[27] H. D. Bandara, X. Xu, and I. Weber, "Patterns for blockchain migration," Jun. 2019. *arXiv:1906.00239*. [Online]. Available: https://arxiv.org/abs/1906.00239

[28] H. Kalodner, S. Goldfeder, A. Chator, and M. Möser, and A. Narayanan, "BlockSci: Design and applications of a blockchain analysis platform," 2017, *arXiv:1709.02489*. [Online]. Available: https://arxiv.org/abs/1709.02489

[29] F. Reid and M. Harrigan, "An analysis of anonymity in the Bitcoin system," in *Security And Privacy In Social Networks*. New York, NY, USA: Springer, 2013, pp. 197–223.

[30] D. Ron and A. Shamir, "Quantitative analysis of the full Bitcoin transaction graph," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, 2013, pp. 6–24.

[31] S. Goldfeder, H. Kalodner, D. Reisman, and A. Narayanan, "When the cookie meets the blockchain: Privacy risks of Web payments via cryptocurrencies," *Proc. Privacy Enhancing Technol.*, vol. 2018, no. 4, pp. 179–199, 2018.

[32] D. Y. Huang, M. M. Aliapoulios, V. G. Li, L. Invernizzi, E. Bursztein, K. McRoberts, J. Levin, K. Levchenko, A. C. Snoeren, and D. McCoy, "Tracking ransomware end-to-end," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 618–631.

[33] W. Chen, Z. Zheng, J. Cui, E. Ngai, P. Zheng, and Y. Zhou, "Detecting Ponzi schemes on Ethereum: Towards healthier blockchain technology," in *Proc. World Wide Web Conf. (WWW)*, 2018, pp. 1409–1418

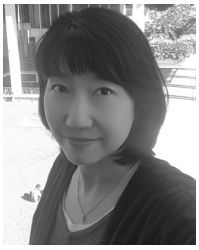
[34] C. Klinkmüller, A. Ponomarev, A. B. Tran, I. Weber, and W. van der Aalst, "Mining blockchain processes: Extracting process mining data from blockchain applications," in *Business Process Management: Blockchain and Central and Eastern Europe Forum*, C. Di Ciccio, R. Gabryelczyk, L. García-Bañuelos, T. Hernaus, R. Hull, M. Indihar Stemberger, A. Ko, and M. Staples, Eds. Nature Switzerland: Springer, 2019, pp. 71–86.

[35] G. Zyskind, O. Nathan, and A. S. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," in *Proc. IEEE Secur. Privacy Workshops*, San Jose, CA, USA, May 2015, pp. 180–184.
[36] A. Ouaddah, A. A. Elkalam, and A. A. Ouahman, "Towards a novel privacy-preserving access control model based on blockchain technology in IoT," in *Proc. Eur. MENA Cooperation Adv. Inf. Commun. Technol.*, Á. Rocha, M. Serrhini, and C. Felgueiras, Eds. Cham, Switzerland: Springer, 2017, pp. 523–533.
[37] J. D. Harris and B. Waggoner, "Decentralized & collaborative AI on blockchain," in *Proc. IEEE Intl. Conf. Blockchain (Blockchain)*, Jul. 2019, pp. 180–184.
[38] A. Baldominos and Y. Saez, "Coin.AI: A proof-of-useful-work scheme for blockchain-based distributed deep learning," *Entropy*, vol. 21, no. 8, p. 723, 2019.
[39] Q. Wang, M. Li, W. Zhang, P. Wang, Z. Shi, and F. Xu, "BDML: Blockchain-based distributed machine learning for model training and evolution," in *Proc. 2nd Intl. Symp. Found. Appl. Blockchain*, Apr. 2019, pp. 10–21.
[40] A. B. Kurtulmus and K. Daniel, "Trustless machine learning contracts; evaluating and exchanging machine learning models on the Ethereum blockchain," 2018, *arXiv:1802.10185*. [Online]. Available: https://arxiv.org/abs/1802.10185
[41] PCI Security Standards Council. (May 2016). *Payment Application Data Security Standard—Requirements and Security Assessment Procedures*. [Online]. Available: https://www.pcisecuritystandards.org/document_library
[42] S. U. Lee, L. Zhu, and R. Jeffery, "A data governance framework for platform ecosystem process management," in *Proc. Int. Conf. Bus. Process Manage. (BPM)*, 2018, pp. 211–227.
[43] The European Parliament and of the Council. *General Data Protection Regulation, GDPR*. Accessed: Oct. 1, 2019. [Online]. Available: https://gdpr-info.eu/
[44] Australian Government. (2018). *My Health Records Amendment (Strengthening Privacy) Act*. [Online]. Available: https://www.legislation.gov.au/Details/C2018A00154
[45] A. Cavoukian, "Privacy by design: The 7 foundational principles," Inf. Privacy Commissioner, Ontario, ON, Canada, Tech. Rep., 2009, vol. 5.
[46] S. Schwerin, "Blockchain and privacy protection in the case of the European general data protection regulation (GDPR): A delphi study," *J. Brit. Blockchain Assoc.*, vol. 1, no. 1, p. 3554, 2018.
[47] M. de Vos, M. Olsthoorn, and J. Pouwelse, "Devid: Blockchain-based portfolios for software developers," in *Proc. IEEE Intl. Conf. Decentralized Appl. Infrastructures (DAPPCON)*, Apr. 2019, pp. 158–163.
[48] P. Dunphy and F. A. P. Petitcolas, "A first look at identity management schemes on the blockchain," *IEEE Security Privacy*, vol. 16, no. 4, pp. 20–29, Jul./Aug. 2018.
[49] A. Dorri, S. S. Kanhere, and R. Jurdak, "MOF-BC: A memory optimized and flexible blockchain for large scale networks," *Future Gener. Comput. Syst.*, vol. 92, pp. 357–373, Mar. 2019.
[50] Y. Yu, Y. Li, J. Tian, and J. Liu, "Blockchain-based solutions to security and privacy issues in the Internet of Things," *IEEE Wireless Commun.*, vol. 25, no. 6, pp. 12–18, Dec. 2018.
[51] M. C. K. Khalilov and A. Levi, "A survey on anonymity and privacy in bitcoin-like digital cash systems," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2543–2585, Mar. 2018.
[52] *Information Technology—Governance of IT—Governance of Data—Part 1: Application of ISO/IEC 38500 to the Governance of Data*, Standard ISO/IEC 38500, Apr. 2017. [Online]. Available: https://www.iso.org/standard/56639.html
[53] V. Khatri and C. V. Brown, "Designing data governance," *Commun. ACM*, vol. 53, no. 1, pp. 148–152, 2010.
[54] K. Weber, B. Otto, and H. Österle, "One size does not fit all—A contingency approach to data governance," *J. Data Inf. Qual.*, vol. 1, no. 1, p. 4, 2009.
[55] *ISO/IEC Data Quality Model*. [Online]. Available: https://iso25000.com/index.php/en/iso-25000-standards/iso-25012?limit=5&limitstart=0
[56] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla, "Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability," in *Proc. 17th IEEE/ACM Intl. Symp. Cluster, Cloud Grid Comput. (CCGRID)*, May 2017, pp. 468–477.
[57] J. Adler, R. Berryhill, A. Veneris, Z. Poulos, N. Veira, and A. Kastania, "Astraea: A decentralized blockchain oracle," in *Proc. IEEE Int. Conf. Internet Things (iThings), IEEE Green Comput. Commun. (GreenCom), IEEE Cyber, Phys. Social Comput. (CPSCom), IEEE Smart Data (SmartData)*, Jul./Aug. 2018, pp. 1145–1152.
[58] S. U. Lee, L. Zhu, and R. Jeffery, "Data governance for platform ecosystems: Critical factors and the state of practice," in *Proc. 21st Pacific–Asia Conf. Inf. Syst.*, 2017.

HYE-YOUNG PAIK received the Ph.D. degree in computer science from the University of New South Wales (UNSW), Sydney, Australia. She is currently a Senior Lecturer with the School of Computer Science and Engineering, UNSW, and collaborates with the Architecture and Analytics Platforms (AAP) Team, Data61, CSIRO, Sydney, as a Visiting Academic. Her expert areas include service-oriented software design and architecture and distributed data/application integration. She is a member of ACM.

XIWEI XU received the Ph.D. degree from the University of New South Wales (UNSW). She is currently a Senior Research Scientist with the Architecture and Analytics Platforms (AAP) Team, Data61, CSIRO, Sydney, and also a Conjoint Lecturer with the School of Computer Science and Engineering (CSE), UNSW. Her main research interest includes software architecture. Since 2015, she has been working on blockchain. She also does research in the areas of service computing, business process, cloud computing, and dependability.

H. M. N. DILUM BANDARA received the M.S. and Ph.D. degrees in electrical and computer engineering from Colorado State University, USA. He is currently a Research Scientist with the Architecture and Analytics Platforms (AAP) Team, Data61, CSIRO, Australia. Prior to that, he was a Senior Lecturer with the University of Moratuwa, Sri Lanka. His research interests include distributed systems (Blockchain, Cloud, and P2P), the IoT, data engineering, and computer security, as well as multidisciplinary applications of those technologies. He is also a Charted Engineer at IESL.



SUNG UNE LEE received the Ph.D. degree in computer science from the University of New South Wales (UNSW), Sydney, Australia. She is currently an Associate Lecturer with the School of Information Systems, Queensland University of Technology (QUT), Brisbane, Australia. She was a Research Associate with the Data61, CSIRO, Sydney. Her research interests include data governance, platform ecosystem, business process, project management, and software process and quality management. She is a member of the Association for Information Systems (AIS).

SIN KUANG LO is currently pursuing the Ph.D. degree with the Architecture and Analytics Platforms (AAP) Team, Data61, CSIRO, Sydney, and the School of Computer Science and Engineering, University of New South Wales (UNSW), Sydney, Australia. His current research interests include software architecture and blockchain, specifically on the role of blockchain within a larger software systems.