International Journal of Computational Intelligence Systems (2024) 17:27
https://doi.org/10.1007/s44196-024-00404-z

RESEARCH ARTICLE

Data Mesh Meets Blockchain

Abdulaziz Almaslukh[1] · Abdulmajeed Alameer[2] · Hamad Alsaleh[1] · Fahad Alkadyan[1] · Nasser Allheeib[1] · Abdulaziz Alhadlag[1] · Yazeed Alabdulkarim[1]

Received: 8 September 2023 / Accepted: 2 January 2024

Abstract
Effective dataset management is crucial for enterprises to make informed decisions and remain competitive. However, centralized dataset management approaches often result in poor scalability, unclear governance, inaccessible data silos, and duplication of efforts. This paper proposes a distributed blockchain-based framework inspired by the data mesh architecture to address these challenges. Our proposed framework leverages blockchain's decentralized nature to enable efficient and transparent dataset sharing across enterprise business domains. By turning datasets into digital assets and business domains into peers, our framework utilizes blockchain smart contracts to allow business domains to view, request, and share datasets. In this paper, we describe the details of our framework, and we analyze it from scalability, accessibility, security, and data governance perspectives. To validate our framework, we provide a proof-of-concept implementation with a publicly available source code.

Keywords Data mesh · Blockchain · Dataset governance · Dataset sharing

1 Introduction

Gaining a deep and factual understanding of the business and its underlying operations is crucial for enterprises to support their daily decisions and stay competitive. Despite the plethora of datasets, enterprises typically do not utilize all aspects of the available datasets due to poor dataset management, unclear governance, inaccessible data silos, and lack of ownership. As a result, the value of the dataset stored in the enterprise data warehouses and lakes has yet to be fully maximized. Moreover, the decision success rate is at risk, and potential business opportunities will likely be missed.

Datasets are constantly generated from various sources and stored in different structured and unstructured formats on scattered storage systems. For instance, Google announced in 2016 that its catalog has over 26 billion datasets that are available to all Google employees, excluding datasets with restricted access. The total expected number of all datasets types could be more than double [1]. Thus, effective dataset management is critical for decision-making, answering business questions, dataset governance, and gaining a competitive advantage. In addition, it would help the enterprises to identify trends and spot problems early by making use of the available datasets intelligently.

Abdulaziz Almaslukh and Abdulmajeed Alameer have contributed equally to this work.

✉ Abdulaziz Almaslukh
aalmaslukh@ksu.edu.sa

Abdulmajeed Alameer
abalameer@ksu.edu.sa

Hamad Alsaleh
haalsaleh@ksu.edu.sa

Fahad Alkadyan
fahadkk00055@gmail.com

Nasser Allheeib
nallheeib@ksu.edu.sa

Abdulaziz Alhadlag
aalhadlag@ksu.edu.sa

Yazeed Alabdulkarim
yabdulkarim@ksu.edu.sa

[1] Department of Information Systems, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia

[2] Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia

Published online: 07 February 2024

It has been the traditional wisdom for enterprises to manage their datasets centrally [1–5]. This dataset management paradigm provides better control, reduces redundancy, and simplifies data management. However, these centralized approaches create an ineffective monolithic architecture as large enterprises' datasets increase continuously. This issue may lead to the creation of data silos and duplication of effort. For example, if a dataset is produced, it may not be easily accessible, or two domains might build the same products, such as dashboards or machine learning models. This happens due to the absence of a clear communication channel within the enterprise as whole.

Since datasets are naturally distributed as they are consumed/produced by various sectors across the enterprise, centralized solutions lead to failures in meeting the enterprise's requirements. Datasets from multiple domains and sources are stored with minimum context. Teams that manage the datasets lack domain expertise and a strong sense of ownership. This issue may impact dataset quality, allow unauthorized access, and result in data soils and poor decision-making. Moreover, tasks to ingest, clean, process, and provide datasets for consumers become the center of focus rather than the data itself. Scaling and managing these tasks are problematic. From one end, the proliferation of dataset sources increases the complexity of managing them centrally. From the other end, prompt requirements and use cases of datasets consumers are growing to meet the needs of enterprises. Furthermore, fulfilling these requirements may require changes impacting the whole data pipeline and delaying data request deliveries.

Recently, a new data management architecture called data mesh has been proposed [6, 7]. In specific, data mesh comes into play to resolve the bottleneck of the overwhelmed centralized data team to answer all the analytical questions coming from different domains. Therefore, the architecture supports decentralizing the team, moving the data closer to the domain owner, and treating the data as a product. Data governance is federated across business domains through mechanisms, including global policies such as data formats, locations, and access control.

As an emerging data management paradigm, existing data mesh research has focused on defining its concepts, suggesting guidelines, and providing reference frameworks [8, 9], with limited practical implementation efforts [10]. Specifically, inter-domain communication and governance challenges to bridge decentralized business domains must be adequately addressed. Current mechanisms focused on communication and information-sharing aspects rather than providing an approach that fits the data mesh paradigm and its principles.

In this paper, we propose a blockchain-based framework solution that is inspired by the data mesh architecture to manage the datasets and resolve the centralized dataset management issues. The proposed framework helps the enterprise maximize the value of the datasets across its domains and govern the datasets sharing process. Data mesh and blockchain synergize very well, as they are both distributed and decentralized. Blockchain technology fulfills several data mesh requirements and principles, as its features support the data mesh architecture in several ways. First, blockchain enables parties to communicate and share data efficiently and transparently. Second, blockchain's smart contracts facilitate the data as a product principle. Third, governance, especially across domains, is a significant issue in the data mesh that blockchain help to address. We demonstrate this synergy by designing a blockchain-powered framework to manage enterprise datasets with a data mesh architecture.

More specifically, the proposed framework turns the datasets into digital assets and the business domains within the enterprise into peers in a decentralized network. The peers interact with each other through a smart contract that facilitates different functionalities such as dataset cataloging, dataset sharing, and tracking the dataset across the enterprise from creation to usage until archiving or destruction. In essence, a smart contract acts as a data contract that can be established between dataset producers and consumers in a governed procedure.

The main contributions of this paper are summarized as follows:

1. Realizing data mesh architecture based on blockchain technology.
2. Proposing a blockchain-powered framework for managing enterprise datasets efficiently with detailed qualitative analysis.
3. Designing a generic smart contract with different transactions to support the data-driven enterprise functionalities.
4. Developing a proof-of-concept with publicly available source code.

The rest of this paper is organized as follows: Sect. 2 provides a thorough background and Sect. 3 reviews the related work. In Sect. 4, we explain the features of the proposed framework and detail its design aspects. Section 5 provides a qualitative analysis of the properties of our proposed framework. Section 6 details the implementation choices for a proof-of-concept of the proposed framework. Section 7, we discuss some of the applications and challenges of our framework. Section 8 concludes the paper.

2 Background

We provide background information about data mesh and blockchain to establish the proper context for our proposed blockchain-powered data mesh framework to manage datasets. Additionally, we describe the data fabric concept, representing a centralized data management architecture, as a related alternative to data mesh.

2.1 Blockchain

Blockchain technology is a distributed ledger that securely stores data entries. It enables entities to exchange information and interact without relying on a centralized governing body [11]. The ledger consists of blocks that store data entries and are bundled together using cryptographic protocols to ensure their integrity [12]. The nodes in the blockchain employ consensus mechanisms to validate and agree upon transactions, ensuring efficiency, fairness, reliability, and security.

Blockchain networks possess several characteristics that make them well suited for several applications [13]; these characteristics include:

- *Decentralization:* There is no central authority responsible for validating and approving ledger records in the blockchain.
- *Immutability:* Records stored in the blockchain are permanent and cannot be changed, edited, or deleted by any network node.
- *Transparency:* All nodes in the blockchain network maintain a complete and auditable copy of the transaction ledger.
- *Traceability:* All transactions can be traced, enabling the retrieval of a comprehensive history for any given record.

Generally, blockchain networks can be classified into two types based on their accessibility and level of control: public and permissioned. Public blockchains, like Bitcoin and Ethereum [14, 15], allow anyone to join without restrictions. In contrast, permissioned blockchains (also known as private blockchains) restrict access to known participants. The characteristics of blockchain networks may differ between these two types. Public blockchains are often more complex due to their openness, necessitating careful design and consensus mechanisms that may impact scalability and performance. Additionally, public blockchains may not be suitable for sharing sensitive information, such as enterprise data, as shared records are visible to all participants [16]. Conversely, permissioned blockchains are better suited for sharing sensitive data and are less susceptible to attacks due to their restrictive access and the known identity of the network participants [17].

One significant aspect of blockchain technology is the concept of smart contracts [18]. Blockchain smart contracts are self-executing contracts where the agreement's terms between two parties are directly encoded into code. These contracts and their agreements exist on a decentralized blockchain network. Similar to legal agreements, smart contracts contain predetermined terms and conditions that the contract parties have agreed upon. Smart contracts are automatically executed when the agreed-upon conditions are met, without the need for a central authority [19]. This automation results in a more efficient, secure, and transparent process, as the contract terms are recorded on the blockchain and can be audited and validated by any party on the network.

2.2 Emerging Data Management Paradigms

Evolving and continual challenges of data handling [20] lead to the emergence of innovative data management designs and architectures. Some of these architectures represent a paradigm shift from existing data management approaches to address their challenges drastically. The following subsections describe two emerging data management paradigms: data mesh and data fabric. Data fabric comes from a data engineer mindset focusing on data processing aspects, such as data integration and lineage. In contrast, data mesh comes from a data owner mindset that focuses on data product aspects, such as data product deliveries. Since there is no one-size fits all solution, we briefly discuss their advantages and disadvantages to help organizations choose the best data management approach depending on the organizational goals and circumstances.

2.2.1 Data Mesh

Data mesh is an emerging data management paradigm [6, 7] that provides a domain-driven design to address the issues of centralized data platforms. It focuses on data as products that are decentralized and distributed across independent and loosely coupled business domains. Data mesh is more suitable for large-scale enterprises or cases where data are mostly used within business domains, providing enhanced adaptability to evolving needs, superior scalability, and increased agility. However, there are potential drawbacks, such as inconsistent data practices, coordination and collaboration complexities, and reliance on standardization. We describe the data mesh principles to illustrate these aspects.

The data mesh approach stands on four principles: domain-oriented decentralized data ownership and architecture, data as a product, self-serve data infrastructure

as a platform, and federated computational governance. These concepts drive a new logical view of the technical architecture and organizational structure. We provide an overview of each of these principles.

- *Domain-oriented decentralized data ownership and architecture:*
  Domain-driven design is a method of decomposing components around business domains. It enables a modular design of software microservices in bounded contexts defined by business domains. This design generates components of microservices that are loosely coupled and scalable [21]. In the same spirit, data mesh decomposes data ownership and architecture into decentralized domain-oriented components. Defined domains become responsible for managing their own datasets and providing them to consumers. Data consumers may pull datasets from different domains. These domains become the architectural components of data management rather than data pipeline stages in a centralized data approach. It is important to note that data pipelines are still present in the data mesh architecture but are decentralized and distributed inside the producing and consuming domains as internal implementation.
- *Data as a product:*
  Treating data as a product helps to resolve issues that prevent utilizing it properly. These issues relate to qualities, such as understandability, discoverability, security, trustworthiness, etc. When considering data as a product, organizations allocate the roles and set the necessary key performance indicators (KPIs) to ensure that the data products are produced and consumed properly to serve the business needs. Data products may be facilitated with data contracts [22] to ensure proper usage and provide necessary guarantees from data owners. A data contract is a formal agreement between data producers and consumers, including product description, scope, access mechanism, constraints, compatibility issues, and service-level agreements (SLAs).
- *Self-serve data infrastructure as a platform:*
  Similar to centralized data management platforms, data processing infrastructure is needed in a decentralized data mesh paradigm. In centralized approaches, data pipelines and infrastructure are the focus, with data being an outcome of them. In contrast, data pipelines and infrastructure are not the focus of the data mesh architecture. Data infrastructure is offered as a service for every domain. This design shift requires technology stacks and infrastructure that process the data with high-level abstract operations and simple provisioning. Offering self-serve data platform services eliminates the need for specialized developers and enables domain autonomy, which is not available yet in today's data technology stacks.
- *Federated computational governance:*
  The data mesh approach creates distributed and independent data products. However, data analytics use cases may require combining and linking data from various domains. For example, a use case may require combining datasets from the business services and manufacturing domains to optimize operations and enhance client satisfaction. Consequently, providing integration and interoperability operations through global standardization across all domains is essential for implementing the data mesh approach. These global decisions and standards should be managed by a federated governance model led by domain and data platform owners.

2.2.2 Data Fabric

Data fabric [23–25] focuses on centralization and unified data access. It creates a unified data integration layer by connecting existing data systems and components to provide a single source of truth for data, similar to the purpose of master data management [26]. It avoids the traditional problems of centralized systems by connecting data from different sources rather than moving the data itself to centralized storage. As a centralized architecture, data fabric simplifies data access, governance, standardization, and quality assurance.

Data fabric is ideal for enterprises with interwind business domains or that require a single source of truth for data. However, the centralized nature of the data fabric may result in potential bottlenecks, slower responsiveness to domain-specific needs, dependency on a centralized team, and scalability challenges. Furthermore, centralized data management can restrict innovation and experimentation as teams may lack the autonomy to explore new technologies and approaches tailored to their specific domain requirements.

3 Related Work

We review the research work related to data mesh and discuss its limitations in Sect. 3.1. Next, we describe and compare related papers in the context of dataset management in Sect. 3.2.

3.1 Data Mesh

The data mesh paradigm, proposed by Zhamak Dehghani in 2019 [6, 7], recently emerged from industrial experiences for managing data with limited scientific research. Netflix and Zalando [27, 28] provided presentations of data mesh implementations in the industry. [29] explained their industrial

experiences, justifying the migration from a centralized data warehouse to a decentralized data mesh architecture. In their case, data were primarily used within domains. Inter-domain requests are served by a central metadata catalog, but data-sharing mechanisms were not provided. More recently, [30] conducted a gray literature survey of 114 industry articles to provide a comprehensive review of data mesh. Moreover, they compared the service-oriented architecture to the data mesh due to their similarities. It answers research questions related to understanding the data mesh architecture, its benefits, architectural design choices, and challenges.

Authors of [8, 9] provided an overview of the data mesh approach and described its motivation and concepts. The work of [8] details different architectures for the data mesh paradigm. A follow-up work [10] focused on identifying potential technologies for implementation purposes and provided a proof-of-concept demonstration. They presented various software solutions and technologies that may be used to serve the purposes of the data mesh architecture. For example, software products for data storage, processing, machine learning, and visualization were presented to support the self-serve data platform principle. For linking domains, authors extended Apache Atlas to run as a catalog for the data mesh, created Slack channels for communication purposes between data teams, and used Google Forums to request data access. Moreover, a standard folder organization is followed by all teams. These inter-domain communication methods are generic and ad-hoc, which limits their usability to support the overall data mesh paradigm and its core principles.

Previous work fails to address the data mesh's concerns of governing and bridging decentralized domains. The proposed solutions in that context focused on communication and information-sharing aspects rather than providing an approach that fits the data mesh paradigm and its principles, such as data as a product. Consequently, the components of the proposed solutions are not well-integrated cohesively. In contrast, our proposed blockchain-powered framework offers a practical solution that links and governs the data mesh's independent domains and products. Both the data mesh and blockchain are fundamentally decentralized. This core similarity makes blockchain technology a perfect fit to support the data mesh principles and solves its decentralization challenges.

3.2 Dataset Management

There has been a variety of research work that focused on managing datasets. On a conceptual level, authors in [31] envisioned an ecosystem, namely Agora, to provide access to data pipeline assets, such as data, algorithms, and computing resources, through marketplaces focusing on these assets as core components. Agora aims to make data science and its technologies more accessible to general users, as it currently requires considerable technical proficiency. Agora architecture builds around assets and consists of two layers: the Asset Layer and the Execution Layer. The connection between the two layers is seamless. The asset layer serves as an advanced ecosystem of multiple asset markets. It facilitates not just the discovery and offering of assets, but also allows for their intelligent combination using asset managers. On the other hand, the execution layer is responsible for running and optimizing the asset execution with respect to the given budget. In contrast, the data mesh architecture addresses enterprise data management challenges and focuses on data as core elements and not data pipeline components.

More tangibly, a research paper [32] proposed a data grid architecture to manage large scientific datasets. It defined requirements and components to integrate computation and storage infrastructure to process and manage large datasets. For cloud computing, [33] focused on minimizing storage costs of datasets, and [34] built a platform to manage large biomedical datasets. These papers focused on providing computational and storage infrastructure for managing datasets.

Another direction of research focused on providing tools and functionalities to manage datasets. Inspired by software version control systems, [35] proposed two integrated systems, similar to git and GitHub [36, 37]. The first system manages versions of datasets and facilitates sharing them. The second system works on top of the first one to provide advanced interaction capabilities and data pipeline tools for tasks, such as data cleaning, version search, integration, differencing, and visualization. These systems may complement the data mesh architecture by providing useful tools for use within domains and/or as part of the self-serve data platform component. Our work differs by focusing on the interactions between the decentralized domains of data mesh.

Several papers focused on building centralized systems to solve dataset challenges, such as data quality [4], data discovery, and metadata collection [1–5]. This line of work focused on the systems aspect of managing datasets and addressed specific issues related to the traditional centralized data management architecture. In contrast, our paper uses the data mesh approach that represents a paradigm shift by focusing on the data itself as the center of the architecture rather than its managing systems. For example, [1] Google presented a project for large-scale enterprises, namely Google Data Search (GOODS), to collect metadata of datasets that are scattered in various storage systems. GOODS creates a global data catalog enabling users to monitor and annotate datasets. It collects metadata from various sources in the background in a non-intrusive manner, allowing parties to freely generate datasets that will be added to the catalog afterward for others to find. Moreover, GOODS provides a dashboard for the data producers to display all their

datasets and enables browsing them by facets (e.g., owner, data center, schema). Equipped with monitoring capabilities, Goods can detect unexpected changes in data size, value distribution, and availability, alerting owners accordingly. GOODS tracks the origin and impact of datasets. It shows which datasets were used to create others (upstream) and which ones rely on them (downstream). For consumers, GOODS offers a search engine with filters to find relevant and current datasets. GOODS focused on data cataloging without including other aspects, such as data contracts and governance, which this paper covers.

4 Framework Design

Data mesh and blockchain synergize very well, as they are both distributed and decentralized. Blockchain technology fulfills several data mesh requirements and principles, as its features support the data mesh architecture in many ways. Following the decentralizing data ownership principle of data mesh, blockchain enables parties to communicate and share data efficiently and transparently. Thus, it can facilitate communication and data sharing between different data domains in the data mesh. Participants can share data and execute transactions transparently and traceably across domains, supporting inter-domain visibility and accountability. Moreover, requirements for a global data catalog are served with a replicated data ledger for all domains.

Blockchain's smart contracts facilitate the data as a product aligned with the second data mesh principle. Smart contracts enable data providers and consumers to agree on specific terms and conditions for data requests. A smart contract may digitally implement and manage data contracts between consumers and producers. Consequently, smart contracts clearly manage and define data products and services. Smart contract transactions enable participants to request a data product, view its metadata, agree on the terms of use, and others.

Governance, especially across domains, is a significant issue in the data mesh that blockchain may address. Blockchain technology facilitates governance among participants via defined consensus protocols implementing the federated computational governance principle of data mesh. These consensus protocols govern the execution of transactions to determine their validity based on specified conditions and approval requirements. Thus, the federation of governance between domains may be achieved with blockchain technology.

We demonstrate this synergy by designing a blockchain-powered framework to manage enterprise datasets with a data mesh architecture. Business domains are participating entities in the blockchain network to share data and exchange value. Data producers add datasets as data products that are managed by smart contracts. Data consumers belonging to various domains issue smart contract transactions to use these datasets and interact with them.

4.1 Framework Functionalities

Our proposed framework can provide a range of functionalities to support data mesh architecture, including dataset ownership, governance, dataset sharing, and dataset cataloging. In this subsection, we highlight the key functionalities of the blockchain-powered framework that are essential to realize the data mesh architecture, including regulating dataset ownership, sharing datasets, establishing data contracts, and maintaining the datasets catalog of the enterprise. It is important to note that we use blockchain to store the metadata of datasets and information capturing their usages. Storing the datasets themselves in the blockchain is impractical, and inefficient due to their large size.

A dataset is typically owned by the domain which produces it. Our framework supports ownership rights once the dataset reference is added to the blockchain by the domain owner. As a result, the framework gives the control to the domain owner, who can explicitly specify the rules and conditions on which the access should be granted via smart contracts (see Sect. 4.2 for details). Therefore, the governance policies are enforced automatically through the framework. By enabling domain producers to establish rules and conditions through smart contracts that meet the owner requirements, our framework maintains a robust governance structure.

On the other hand, dataset consumers can establish a data contract with the dataset domain owner. The data contract facilitates secure and transparent datasets exchange between domains. The framework inherently tracks the datasets and data contracts; as traceability is one of the main features of blockchain technology. Additionally, dataset cataloging is supported naturally to improve dataset collaboration and reduce duplication of effort. This integral aspect significantly streamlines data discovery processes to fully make use of the shared datasets while optimizing resource utilization within the enterprise ecosystem.

4.2 Smart Contract Transactions

Different smart contracts can be developed to support the framework functionalities. A generic smart contract contains various transactions that can be executed by the dataset producers and dataset consumers. A valid execution of any transaction is based on satisfying the predefined terms and conditions. With the smart contract in place, autonomous execution of transactions is possible to enhance efficiency and compliance. For instance, the Sales business domain could publish a dataset about the

customers purchasing activities in the last month. The Marketing domain would request this dataset for developing a new marketing strategy. If the producer domain (the Sales domain) makes the dataset freely available. The consumer domain (the Marketing domain) accesses the dataset immediately. However, if the owner puts restrictions on the dataset for privacy concerns, the request shall be approved by the owner (the Sales domain).

The deployed smart contract facilitates and streamlines the transaction execution within the blockchain network. Table 1 shows a design of the smart contract transactions for our framework, along with their inputs, outputs, issuing party, and approval requirements. The smart contract may be extended to include new transactions or modify existing ones to meet the new enterprise requirements. Thus, the framework supports the flexible deployment of different smart contracts that can accommodate all the different domains' needs.

More specifically, datasets are converted into digital assets represented by a JSON object. Each dataset has a unique identifier. Figure 1 is shown an example of a dataset JSON object. The dataset JSON object provides information about the dataset, such as name, description, producer, and the dataset location (URL). The collection of all dataset JSON objects represents a clear catalog of all enterprise datasets assets. These assets are managed by the deployed smart contract on which the agreed transactions can be executed.

Similarly, the data contracts between the dataset producer and the consumer are converted into digital assets represented by a JSON object which is related to the given dataset. Figure 2 is shown an example of a data contract JSON object. The data contract JSON object primarily provides

Table 1 A proposed design of smart contract transactions and their descriptions

Transaction	Description	Input	Output	Issuing party	Approval requirements
addDataset	Create a new record of the dataset reference	datasetID, dataset JSON object	Success if match conditions; Otherwise, failure	Producer domain	None
updateDataset	Update the dataset	datasetID, dataset JSON object	Success if match conditions; Otherwise, failure	Producer domain	None
removeDataset	Remove the dataset reference	datasetID	Success if match conditions; Otherwise, failure	Producer domain	None
addContract	Create a new record of the data contract	datasetID, contractID, contract JSON object	Success if match conditions; Otherwise, failure	Consumer domain	Yes, the producer domain shall approve the data contract if the requested dataset is sensitive
updateContract	Update the contract information	contractID, contract JSON object	Success if match conditions; Otherwise, failure	Any domain	Yes, the producer domain shall approve the data contract if the requested dataset is sensitive
approveContract	Update the status of the data contract to "Active"	contractID	Success if match conditions; Otherwise, failure	Producer domain	None
rejectContract	Update the status of the data contract to "Rejected"	contractID	Success if match conditions; Otherwise, failure	Producer domain	None
terminateContract	Terminate the data contract	contractID	Success if match conditions; Otherwise, failure	Any domain	None
viewConsumers	View all the active data contract	datasetID	Return all data contract JSON objects associated with this datasets or null otherwise	Any domain	None
viewCatalog	View all the datasets in the enterprise	None	Return all datasets JSON objects or null otherwise	Any domain	None
viewHistory	View all the datasets from which the given dataset is derived	datasetID	Return all datasets JSON objects or null otherwise	Any domain	None

Fig. 1 Example of dataset JSON object

```
{
  "id": "DS5002",
  "name": "Product Sales and Inventory",
  "description": "Sales and inventory information of products",
  "producer": "SalesDept",
  "derivedFrom": "DS4199",
  "sensitive": true,
  "location": "example.org/datasets/ds5002",
  "attributes": [
    "customer_id",
    "product_id",
    "quantity",
    "price"
  ],
  "attributesDescription": [
    "ID of the customer",
    "ID of the product",
    "quantity of the sales",
    "the price of the product"
  ]
}
```

Fig. 2 Example of data contract JSON object

```
{
  "id": "DC2482",
  "datasetID": "DS5002",
  "consumer": "MarketingDept",
  "effectiveDate": "20-05-2023",
  "status": "active",
  "products": [
    "Traffic Analysis",
    "Conversion Tracking"
  ],
  "purpose": "For website performance analysis and optimization"
}
```

information about the data contract, such as contract ID, dataset ID, consumer, contract status, and the purposes of using the dataset.

To illustrate the framework functionalities, a domain produces a dataset, the reference of the dataset can be added to the ledger by invoking (*addDataset*) transaction on the deployed smart contract. The dataset is associated with the producer domain as the owner. Other domains, the consumers, can request this dataset by invoking (*addContract*). The data contract is established between the dataset producer domain and the consumer domain. The dataset producer domain receives the request, and the request is subject to the approval or rejection of the dataset owner (*approveContract*). In case of approval, the contract takes effect, and the contract's status changes to active. Afterward, the dataset is available to the consumer. Figure 3 illustrates the process.

The sharing of the actual dataset may be implemented in several ways, such as API access or enterprise storage share link. For sensitive datasets, access control may be required to prevent unauthorized entry. The producer domain should

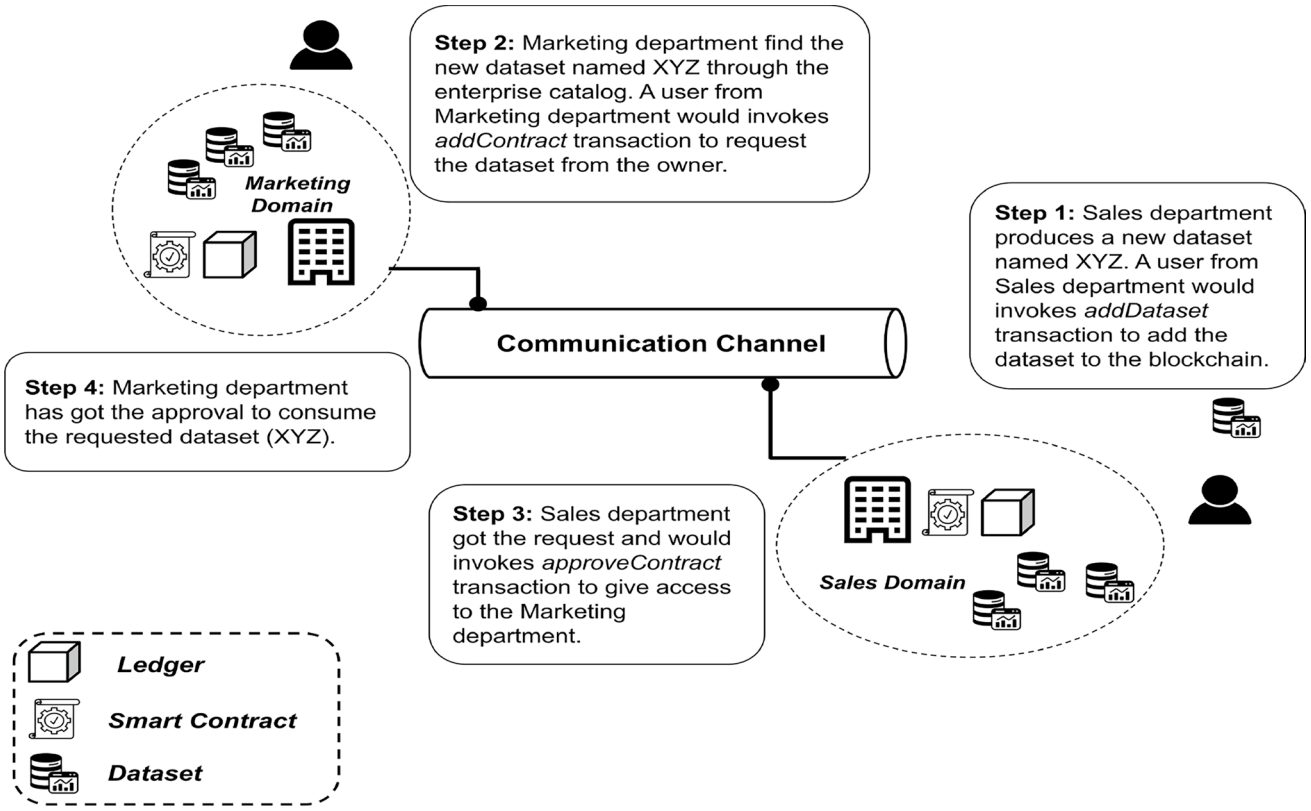

Fig. 3 Illustration of sharing a dataset between domains in a governed manner

grant access through the existing enterprise authentication and authorization system using, for example, a private API.

The datasets can also be removed (*removeDataset*) by the producer domain, and the data contracts can be terminated (*terminateContract*) by the participating domains. All domains can view the consumers of a given dataset (*viewConsumers*). In addition, the dataset catalog can be retrieved by any domain by invoking (*viewCatalog*) to view all the enterprise datasets.

4.3 Network Architecture

The network architecture consists of several components, including nodes, protocols, and consensus mechanisms. Sharing and exchanging datasets within the enterprise are an internal process. Therefore, a private blockchain network clearly fits our proposed framework to share data securely and efficiently across the enterprise domains which are previously known. This gives greater control over accessing and changing data on the blockchain and prevents disclosing sensitive data beyond the enterprise boundaries.

Each business domain within the enterprise turns into a node of the blockchain network. The protocols that govern the validity of transactions are defined within the deployed smart contracts. The domains should agree on a consensus mechanism to reach an agreement on the current state of the blockchain. More specifically, each domain has its own copy of the blockchain state and can approve, propose, and validate transactions. Domain members cooperatively request to execute and approve transactions through a communication channel (see Fig. 4). These communications are streamlined by the deployed smart contracts to automate the transaction execution, eliminating human interventions and minimizing human error while maximizing framework efficiency.

The main aspects of the proposed framework is the dataset sharing and governance. Each node, the business domain, could produce the dataset of which the enterprise should aware. This can be done using the node's producer identity, the deployed smart contract, and the Public Key Infrastructure (PKI) [38] of the producer domain. Upon the sharing dataset transaction success, every other nodes updates the state of its blockchain, and the dataset information is available across the enterprise and could be requested using the same deployed smart contract. However, the approval is based on the rules and conditions that are associated with the shared dataset. If the dataset is sensitive, the formal approval should be granted from the producer node.

The blockchain network offers a variety of consensus mechanisms designed to assist nodes in achieving a consistent state. However, consensus mechanisms suitable for

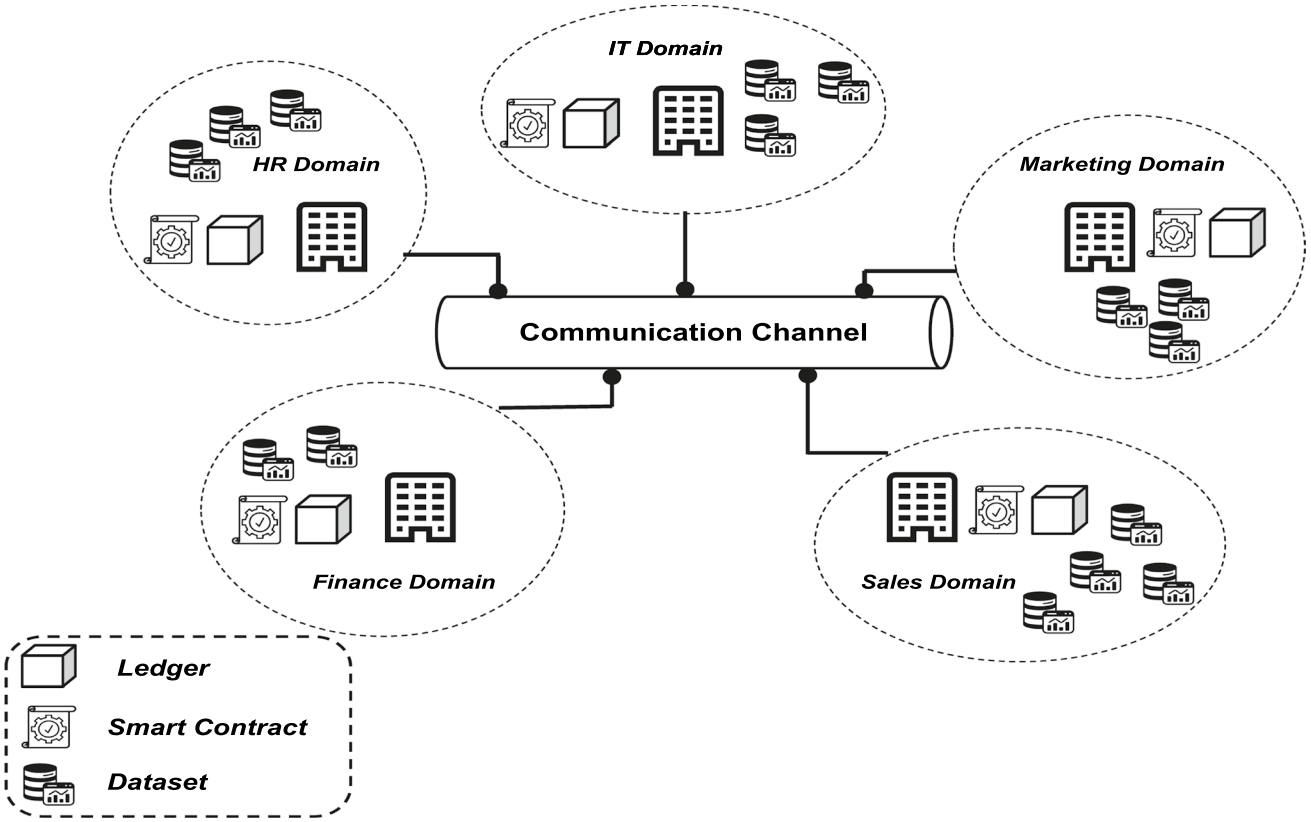

Fig. 4 Framework overview

public blockchain networks, like proof of work [39] or proof of stake [40], may not be suitable for implementation in private settings. Given that the proposed framework operates within a classified private blockchain network, Practical Byzantine Fault Tolerance (PBFT) [41] and its variants emerge as suitable consensus mechanisms among the nodes.

5 Qualitative Analysis

In this section, we analyze the key properties of our framework from various qualitative perspectives. The following is a detailed analysis of our framework properties:

- **Decentralization**: Both data mesh and blockchain are fundamentally decentralized. In our framework, dataset ownership and control are decentralized. By eliminating the central authority that maintains data, or controls the data access, our framework allows for a direct peer-to-peer interaction and data exchange. This facilitates seamless collaboration, dataset sharing, and integration between different business domains, promoting innovation, and cross-functional insights.
- **Data Accessibility**: The design of our framework enables domain teams to easily share their data products and make them easily accessible to other business domains. Thus, it improves data collaboration and sharing, and eliminates redundancy and duplication of efforts. Using our framework, datasets will no longer be stored in isolated silos. Every domain has a replica of the datasets catalog and may easily view them and request access to any dataset by issuing smart contract transactions.
- **Transparency**: Blockchain, by design, promotes transparency within a data mesh. Blockchain creates a record of data transactions and modifications that are viewable by all nodes in the blockchain. This transparency helps establish trust among different business domains and enables participants to verify the integrity and the history of the shared datasets.
- **Federated Governance and Control**: Since our framework uses blockchain, it allows for establishing rules and policies governing datasets structure, access, and usage. By leveraging smart contracts transactions in the blockchain, our framework can enforce data governance processes, ensuring that datasets that are shared follow the predefined rules and guidelines that are agreed upon by the participating domains.
- **Data Security and Privacy**: Blockchain's cryptographic protocols can enhance data security and privacy within a data mesh. The security of our framework is granted

through the underlying blockchain framework we have used (i.e., Hyperledger Fabric). Several studies have analyzed the Hyperledger Fabric and its security features [42–44]. The results of these studies found the framework to be secure if implemented correctly. In addition, using smart contract transactions (described in Sect. 4), our framework allows access to be granted only to authorized consumers. Access to the datasets can also be revoked by terminating the contract using smart contract transactions. Our framework can be integrated with existing enterprise authentication and authorization systems to allow or deny access to sensitive data, as described in Sect. 4.

- **Scalability**: The scalability heavily depends on the blockchain network choice and the underlying consensus mechanism. The adopted blockchain network in our framework is a permissioned blockchain, which is shown to be more scalable than public blockchain networks [45, 46]. Thus, it supports high transaction throughput due to a limited number of known participants.
- **Resilience**: The decentralization nature of blockchain and data mesh reduces the risk of a single point of failure, making our framework more resilient against data access disruptions or attacks. This decentralization ensures that even if a certain data source fails or goes offline, datasets from other sources remain accessible, improving the overall system's reliability.

6 Implementation

To implement a prototype of our proposed framework [47], we use Hyperledger Fabric [48] to create a permissioned blockchain network. The blockchain network consists of two business domains, namely, the Sales and Marketing domains. Each domain in the Hyperledger Fabric network has a Membership Service Provider (MSP) that uses public key infrastructure (PKI) to manage identities and authentication. We used the Hyperledger Fabric cryptogen utility tool to generate public and private keys for all entities in the network. We created two orderer instances to provide the ordering service for the Hyperledger Fabric network. The ordering service uses the Raft consensus protocol [49] to ensure that all nodes in the network have a consistent view of the blockchain ledger.

For each domain, we create a peer node to store a copy of the blockchain ledger. Multiple peer nodes can be created per domain for availability and durability. Peer nodes issue transactions on behalf of their domains and act as endorsers, which means that they must approve transactions before they are considered valid. We also develop a smart contract in Golang [50], implementing the transactions shown in Fig. 5

In Fig. 6, we show the peer of Sales domain executing the "*addDataset*" transaction with a JSON object as input. The JSON object represents the dataset to be added to the ledger with the dataset information, including the producer, dataset description, and other information. The peer of Marketing domain executes the transaction "*addContract*" to establish

Fig. 5 A list of the implemented transactions for the smart contract

TRANSACTION	TRANSACTION
1. Add Dataset	2. Read Dataset
3. Update Dataset	4. Remove Dataset
5. Add Data Contract	6. Read Data Contract
7. Approve Contract	8. Reject Contract
9. Terminate Contract	10. Update Contract
11. View Catalog	12. View Consumers
13. Show History	

Fig. 6 A peer of sales domain issues "*addDataset*" to add the dataset reference to the blockchain

FIELD	VALUE
ID	DS5002
Name	Product Sales and Inventory
Description	Sales and inventory information of products
Producer	SalesDept
Derived from	DS4199
Sensitive	True
Location	example.org/datasets/ds5002
Attributes	customer_id, product_id, quantity,price
Attributes Description	ID of the customer, ID of the product, quantity of the sales, the price of the product

a data contract to request the dataset with its status "Pending." Then, the peer of Sales domain executes the transaction "*approveContract*" or "*rejectContract*" to either approve or reject the contract, respectively, and sets its status accordingly. Figure 7 shows the Sales domain peer approving the contract, changing its status to "Active".

In Fig. 8, we show when a peer of any domain executes the "*viewCatalog*" transaction to show the enterprise dataset catalog. Figure 9 shows the consumers of the given dataset when a peer of any domain executes the "*viewConsumers*" transaction. In Fig. 10, we demonstrate that any peer from any domain can view the history of a dataset by executing the "*viewHistory*" transaction. This transaction shows the history of the dataset, including all of the dataset that have been used to generate the given dataset.

7 Discussion

Our framework facilitates the functionalities of various organizations, specifically data-centric ones. The framework presented here can be used by organizations that rely heavily on data science or artificial intelligence applications to better manage and share datasets. It is common knowledge that these applications are mainly built on prior datasets [51]. Therefore, a "view catalog" can be

Fig. 7 A peer of Sales domain issues "*approveContract*" to grant access of dataset "DS5002" to the Marketing domain

FIELD	VALUE
ID	DC2482
Dataset ID	DS5002
Consumer	MarketingDept
Effective Date	18-01-2023
Description	Website Data Analytics Contract
Status	Active
Products	Traffic analysis, Conversion tracking
Purpose	For website performance analysis and optimization

ID	NAME	PRODUCER	DESCRIPTION
DS1000	Website Visitor Profiles	MarketingDept	This dataset contains detailed profiles of website visitors
DS3005	Website Traffic Analytics	MarketingDept	This dataset contains analytics data for website traffic
DS4199	Customer Sales Data	SalesDept	Sales information of customers using the application
DS5002	Product Sales and Inventory	SalesDept	Sales and inventory information of products

Fig. 8 A peer of any domain issues "*viewCatalog*" to view all the dataset in the enterprise

CONTRACT ID	CONSUMER	DATASET ID	PURPOSE	PRODUCTS
DC2022	SalesDept	DS3005	For website traffic analysis	Website traffic reports, visitor segmentation
DC2482	MarketingDept	DS5002	For website performance analysis and optimization	Traffic analysis, conversion tracking

Fig. 9 A peer of any domain issues "*viewConsumers*" to view all the consumers of dataset "DS5002"

```
Enter dataset ID to show dependency: DS5002

DS4199
|___DS5002
```

Fig. 10 A peer of any domain issues "*viewHistory*" to show the history of dataset "DS5002"

used by data teams within an organization to determine which datasets belong to which domain. Moreover, since domains own their datasets, data teams will be aware of any new features added by those domains.

Additionally, the framework can also be implemented by organizations without data teams specialized in data analytics and artificial intelligence, who hire third parties to manage their data-centric and artificial intelligence solutions, thus facilitating the communication between internal organization domains as well as external third parties. Typically, domains update their catalog on a continuous basis to improve their data quality using several features provided by our framework, including "*updateDataset*", which enables domain experts to provide annotations and improve data quality for building advanced analytics solutions.

Our framework is also useful for organizations planning to implement knowledge graph architecture in the exchange of information across domains. Using our framework, data teams can integrate data points from different domains, which facilitates the creation of new knowledge and information search from the uploaded data sources [52].

Despite the various functionalities offered by the proposed framework, it has a number of challenges that range from cultural to technical. The discovery of datasets is an essential component of our framework. However, producer domains must continuously update the catalog of data products to reflect any changes to existing or new data products to maintain discoverability and relevance. Therefore, a sub-challenge may arise for producer domains in describing their datasets and which domains could benefit from producer data products.

As for the content level of the datasets, consumer domains may find it challenging to understand the semantics behind the features of the datasets produced. Thus, the producer domain should ensure that documentation of the dataset is produced to meet the needs and to facilitate the understanding of the consumer domains within the organization. The quality of content in producer datasets is another obstacle that may hinder the usability of producer datasets. Therefore, producer domains should monitor their dataset usability and address any related issues raised by other consumer domains.

In terms of data sensitivity, data in the datasets may contain sensitive information; therefore, for protection, it is imperative that data anonymization be applied to such data by producer domains. Moreover, another challenge may arise when two or more domains own a dataset. In the case of domains that add data contracts, all producers must accept them, which can be time-consuming and challenging to apply.

8 Conclusion

We introduce a blockchain-powered framework to implement data mesh architecture for managing datasets within the enterprise. The framework highlights the principles of data mesh architecture, including decentralizing data ownership, dealing with data as a product, and applying federated governance, through various functionalities implemented in a smart contract which is deployed on every domain of the enterprise. We have made the proof-of-concept implementation publicly available to allow the community to validate, exploit, and expand our proposed framework.

Acknowledgements The authors would like to thank the Researchers Supporting Project (No. RSPD2023R609), King Saud University, Riyadh, Saudi Arabia, for supporting this work.

Author Contributions AA: methodology, writing—original draft, and writing—review and editing. AA: formal analysis, and writing—review and editing. HA and AA: writing—original draft and writing—review and editing. FA: formal analysis and implementation. NA: funding acquisition, and writing—review and editing. YA: methodology, writing—original draft, and writing—review and editing.

Funding This work is supported by Researchers Supporting Project (No. RSPD2023R609), King Saud University, Riyadh, Saudi Arabia.

Data availability Data availability is not applicable. Howerver, the code is avaiable online.

Declarations

Conflict of Interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Halevy, A., Korn, F., Noy, N.F., Olston, C., Polyzotis, N., Roy, S., Whang, S.E.: Goods: organizing google's datasets. In: Proceedings of the 2016 International Conference on Management of Data, pp. 795–806 (2016). https://doi.org/10.1145/2882903.2903730
2. Brickley, D., Burgess, M., Noy, N.: Google dataset search: building a search engine for datasets in an open web ecosystem. In: The World Wide Web Conference, pp. 1365–1375 (2019). https://doi.org/10.1145/3308558.3313685
3. Fernandez, R.C., Abedjan, Z., Koko, F., Yuan, G., Madden, S., Stonebraker, M.: Aurum: a data discovery system. In: 2018 IEEE 34th International Conference on Data Engineering (ICDE), pp. 1001–1012 (2018). https://doi.org/10.1109/ICDE.2018.00094
4. Schelter, S., Lange, D., Schmidt, P., Celikel, M., Biessmann, F., Grafberger, A.: Automating large-scale data quality verification. Proc. VLDB Endow. **11**(12), 1781–1794 (2018). https://doi.org/10.14778/3229863.3229867
5. Francia, M., Gallinucci, E., Golfarelli, M., Leoni, A.G., Rizzi, S., Santolini, N.: Making data platforms smarter with MOSES. Fut. Gener. Comput. Syst. **125**, 299–313 (2021). https://doi.org/10.1016/j.future.2021.06.031
6. Dehghani, Z.: How to move beyond a monolithic data lake to a distributed data mesh. https://martinfowler.com/articles/data-monolith-to-mesh.html. Accessed on 17 June 2023
7. Dehghani, Z.: Data mesh principles and logical architecture. https://martinfowler.com/articles/data-mesh-principles.html. Accessed on 17 June 2023
8. Butte, V.K., Butte, S.: Enterprise data strategy: a decentralized data mesh approach. In: 2022 International Conference on Data Analytics for Business and Industry (ICDABI), pp. 62–66. IEEE (2022). https://doi.org/10.1109/ICDABI56818.2022.10041672
9. Machado, I.A., Costa, C., Santos, M.Y.: Data mesh: concepts and principles of a paradigm shift in data architectures. Procedia Comput. Sci. **196**, 263–271 (2022). https://doi.org/10.1016/j.procs.2021.12.013
10. Araújo Machado, I., Costa, C., Santos, M.Y.: Advancing data architectures with data mesh implementations. In: Intelligent Information Systems: CAiSE Forum 2022, Leuven, Belgium, 6–10 June 2022, Proceedings, pp. 10–18. Springer (2022). https://doi.org/10.1007/978-3-031-07481-3_2
11. Yaga, D., Mell, P., Roby, N., Scarfone, K.: Blockchain technology overview. arXiv preprint arXiv:1906.11078 (2019). https://doi.org/10.6028/NIST.IR.8202
12. Alabdulkarim, Y., Alameer, A., Almukaynizi, M., Allheeib, N., Alkadyan, F., Almaslukh, A.: Managing expatriate employment contracts with blockchain. Electronics **12**(7), 1673 (2023). https://doi.org/10.3390/electronics12071673
13. Abu-Elezz, I., Hassan, A., Nazeemudeen, A., Househ, M., Abd-Alrazaq, A.: The benefits and threats of blockchain technology in healthcare: a scoping review. Int. J. Med. Inform. **142**, 104246 (2020). https://doi.org/10.1016/j.ijmedinf.2020.104246
14. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system. Decentralized business review, 21260 (2008)
15. Wood, G., et al.: Ethereum: A secure decentralised generalised transaction ledger. Ethereum project yellow paper **151**(2014), 1–32 (2014). https://doi.org/10.3390/electronics12071673
16. Zheng, K., Zheng, L.J., Gauthier, J., Zhou, L., Xu, Y., Behl, A., Zhang, J.Z.: Blockchain technology for enterprise credit information sharing in supply chain finance. J. Innov. Knowl. **7**(4), 100256 (2022). https://doi.org/10.1016/j.jik.2022.100256
17. Kshetri, N.: Blockchain's roles in strengthening cybersecurity and protecting privacy. Telecommun. Policy **41**(10), 1027–1038 (2017). https://doi.org/10.1016/j.telpol.2017.09.003
18. Hewa, T.M., Hu, Y., Liyanage, M., Kanhare, S.S., Ylianttila, M.: Survey on blockchain-based smart contracts: technical aspects and future research. IEEE Access **9**, 87643–87662 (2021). https://doi.org/10.1109/ACCESS.2021.3068178
19. Wang, S., Ouyang, L., Yuan, Y., Ni, X., Han, X., Wang, F.-Y.: Blockchain-enabled smart contracts: architecture, applications, and future trends. IEEE Trans. Syst. Man Cybern. Syst. **49**(11), 2266–2277 (2019). https://doi.org/10.1109/TSMC.2019.2895123
20. Nargesian, F., Zhu, E., Miller, R.J., Pu, K.Q., Arocena, P.C.: Data lake management: challenges and opportunities. Proc. VLDB Endow. **12**(12), 1986–1989 (2019). https://doi.org/10.14778/3352063.3352116
21. Thönes, J.: Microservices. IEEE Softw. **32**(1), 116–116 (2015). https://doi.org/10.1109/MS.2015.11
22. How, M., How, M.: The role of the data contract. In: The Modern Data Warehouse in Azure: Building with Speed and Agility on Microsoft's Cloud Platform, pp. 163–180 (2020) https://doi.org/10.1007/978-1-4842-5823-1_6
23. Castelluccio, M.: Data fabric architecture. Strateg. Finance **103**(4), 57–58 (2021)
24. Kuftinova, N., Maksimychev, O., Ostroukh, A., Volosova, A., Matukhina, E.: Data fabric as an effective method of data management in traffic and road systems. In: 2022 Systems of Signals Generating and Processing in the Field of on Board Communications, pp. 1–4 (2022). https://doi.org/10.1109/IEEECONF53456.2022.9744402. IEEE
25. Gupta, A.: Using data fabric architecture to modernize data integration. https://www.gartner.com/smarterwithgartner/data-fabric-architecture-is-key-to-modernizing-data-management-and-integration
26. Loshin, D.: Master Data Management. Morgan Kaufmann, Burlington (2010)
27. Cunningham, J.: Netflix data mesh: composable data processing-Justin Cunningham. https://www.youtube.com/watch?v=TO_IiN06jJ4. Accessed on 17 June 2023
28. Schultze, M., Wider, A.: Data mesh in practice: how Europe's leading online platform for fashion goes beyond the data lake. https://www.youtube.com/watch?v=eiUhV56uVUc. Accessed on 17 June 2023
29. Loukiala, A., Joutsenlahti, J.-P., Raatikainen, M., Mikkonen, T., Lehtonen, T.: Migrating from a centralized data warehouse to a decentralized data platform architecture. In: Product-Focused Software Process Improvement: 22nd International Conference, PROFES 2021, Turin, Italy, November 26, 2021, Proceedings 22, pp. 36–48. Springer (2021). https://doi.org/10.1007/978-3-030-91452-3_3
30. Goedegebuure, A., Kumara, I., Driessen, S., Di Nucci, D., Monsieur, G., Heuvel, W.-j.v.d., Tamburri, D.A.: Data mesh: a systematic gray literature review. arXiv preprint arXiv:2304.01062 (2023)
31. Traub, J., Kaoudi, Z., Quiané-Ruiz, J.-A., Markl, V.: Agora: bringing together datasets, algorithms, models and more in a unified ecosystem [vision]. ACM SIGMOD Rec. **49**(4), 6–11 (2021). https://doi.org/10.1145/3456859.3456861
32. Chervenak, A., Foster, I., Kesselman, C., Salisbury, C., Tuecke, S.: The data grid: towards an architecture for the distributed management and analysis of large scientific datasets. J. Netw. Comput. Appl. **23**(3), 187–200 (2000). https://doi.org/10.1006/jnca.2000.0110
33. Yuan, D., Yang, Y., Liu, X., Li, W., Cui, L., Xu, M., Chen, J.: A highly practical approach toward achieving minimum data sets storage cost in the cloud. IEEE Trans. Parallel Distrib. Syst. **24**(6), 1234–1244 (2013). https://doi.org/10.1109/TPDS.2013.20
34. Bessani, A., Brandt, J., Bux, M., Cogo, V., Dimitrova, L., Dowling, J., Gholami, A., Hakimzadeh, K., Hummel, M., Ismail, M., et al.: Biobankcloud: a platform for the secure storage, sharing,

and processing of large biomedical data sets. In: Biomedical Data Management and Graph Online Querying: VLDB 2015 Workshops, Big-O (Q) and DMAH, Waikoloa, HI, USA, August 31–September 4, 2015, Revised Selected Papers 1, pp. 89–105. Springer (2016). https://doi.org/10.1007/978-3-319-41576-5_7
35. Bhardwaj, A.P., Bhattacherjee, S., Chavan, A., Deshpande, A., Elmore, A.J., Madden, S., Parameswaran, A.G.: Datahub: collaborative data science & dataset version management at scale. In: Seventh Biennial Conference on Innovative Data Systems Research, CIDR 2015, Asilomar, CA, USA, January 4–7, 2015, Online Proceedings (2015). http://cidrdb.org/cidr2015/Papers/CIDR15_Paper18.pdf
36. Spinellis, D.: Git. IEEE Softw. **29**(3), 100–101 (2012). https://doi.org/10.1109/MS.2012.61
37. GitHub: GitHub. https://github.com/. Accessed on 17 June 2023
38. Diffie, W., Hellman, M.E.: New directions in cryptography. Democratizing cryptography: The work of Whitfield Diffie and Martin Hellman (1st ed.). ACM, New York, NY, USA, pp. 365–390 (2022). https://doi.org/10.1145/3549993.3550007
39. Xiao, Y., Zhang, N., Lou, W., Hou, Y.T.: A survey of distributed consensus protocols for blockchain networks. IEEE Commun. Surv. Tutor. **22**(2), 1432–1465 (2020)
40. King, S., Nadal, S.: Ppcoin: peer-to-peer crypto-currency with proof-of-stake. Self-published paper, August **19**(1) (2012)
41. Ongaro, D., Ousterhout, J.: In search of an understandable consensus algorithm. In: 2014 USENIX Annual Technical Conference (USENIX ATC 14), pp. 305–319 (2014)
42. Brotsis, S., Kolokotronis, N., Limniotis, K., Bendiab, G., Shiaeles, S.: On the security and privacy of hyperledger fabric: challenges and open issues. In: 2020 IEEE World Congress on Services (SERVICES), pp. 197–204 (2020). https://doi.org/10.1109/SERVICES48979.2020.00049
43. Alshalali, T., M'Bale, K., Josyula, D.: Security and privacy of electronic health records sharing using hyperledger fabric. In: 2018 International Conference on Computational Science and Computational Intelligence (CSCI), pp. 760–763 (2018). https://doi.org/10.1109/CSCI46756.2018.00152
44. Stamatellis, C., Papadopoulos, P., Pitropakis, N., Katsikas, S., Buchanan, W.J.: A privacy-preserving healthcare framework using hyperledger fabric. Sensors (2020). https://doi.org/10.3390/s20226587
45. Guggenberger, T., Sedlmeir, J., Fridgen, G., Luckow, A.: An in-depth investigation of the performance characteristics of hyperledger fabric. Comput. Ind. Eng. **173**, 108716 (2022)
46. Nasir, Q., Qasse, I.A., Abu Talib, M., Nassif, A.B.: Performance analysis of hyperledger fabric platforms. Secur. Commun. Netw. (2018). https://doi.org/10.1155/2018/3976093
47. Almaslukh, A., Alameer, A., Allheeib, N., Alkadyan, F., Alhadlag, A., Alabdulkarim, Y.: Data mesh meets blockchain. https://github.com/FahadKK/Datamesh-Blockchain
48. Fabric, H.: Hyperledger Fabric—Hyperledger. https://www.hyperledger.org/use/fabric. Accessed on 19 June 2023
49. Ongaro, D., Ousterhout, J.: In search of an understandable consensus algorithm. In: Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference. USENIX ATC'14, pp. 305–320. USENIX Association, USA (2014)
50. Go: The Go Programming Language. https://go.dev/. Accessed on 19 June 2023
51. Rodríguez-García, J.D., Moreno-León, J., Román-González, M., Robles, G.: Introducing artificial intelligence fundamentals with learningml: artificial intelligence made easy. In: Eighth International Conference on Technological Ecosystems for Enhancing Multiculturality, pp. 18–20 (2020)
52. Martin, S., Szekely, B., Allemang, I.D.: O'Reilly media: The rise of the knowledge graph (2021)