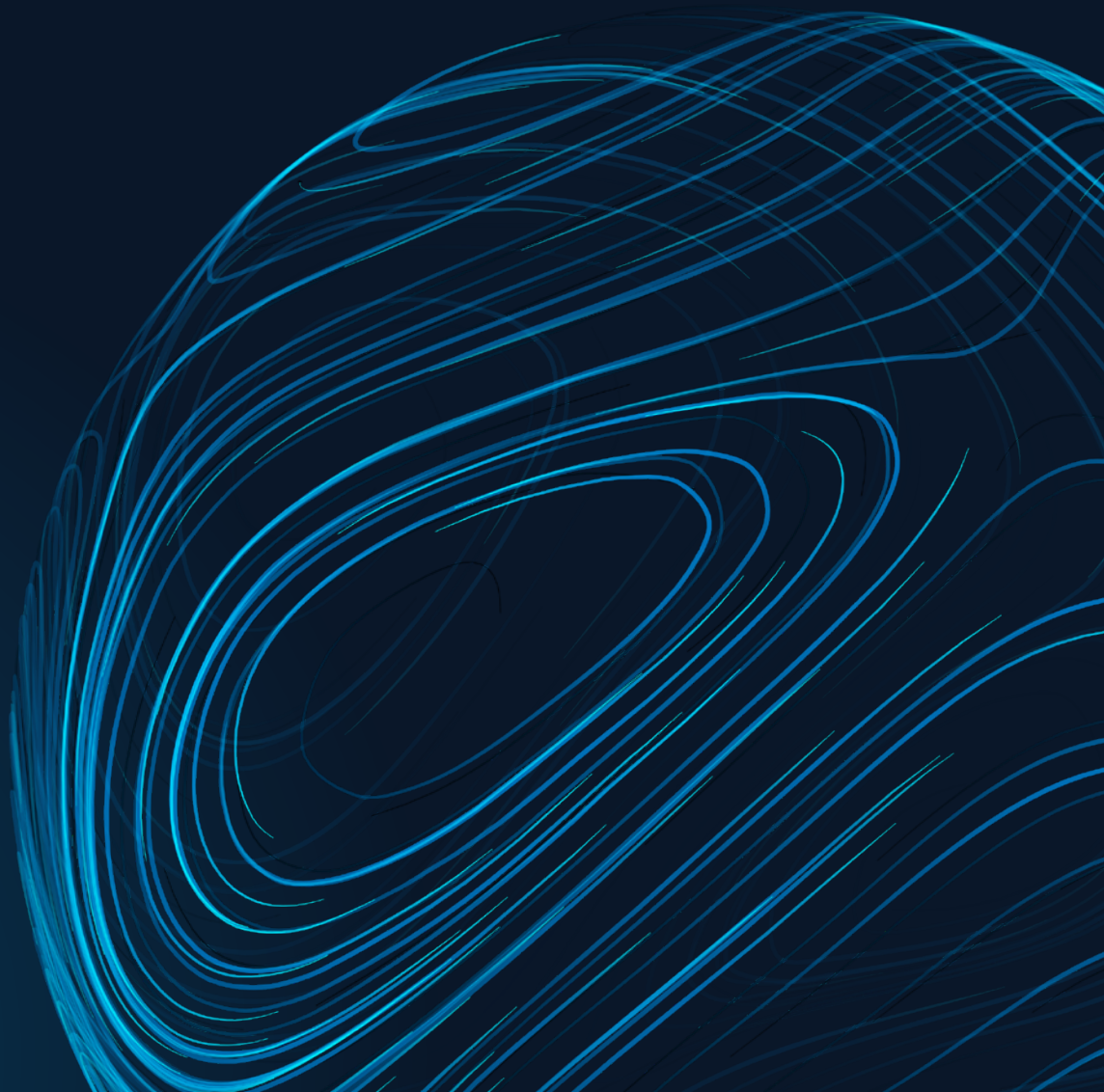
SIMBA

The Curation, Interfacing and Analysis Lifecycle of Blockchain Application Data

This paper discusses the lifecycle of data for next generation Web3 decentralized applications, describing how data can be specified, curated, and analyzed. The focus of this work is to study the different types of approaches for indexing blockchain data in a way that application data and relationships can be designed, retained and exposed.



Ian J. Taylor
CTO, SIMBA Chain

Table of Contents

Abstract

This paper discusses the lifecycle of data for next generation Web3 decentralized applications, describing how data can be specified, curated, and analyzed. The focus of this work is to study the different types of approaches for indexing blockchain data in a way that application data and relationships can be designed, retained and exposed. Blockchain transactions conform to smart contracts definitions, which can have inherent structures through standards such as NFTs, but can also be designed in a custom way for bespoke use cases, Existing Web3 approaches either focus on service-based approaches that build custom pipelines that extract data in a specific way, or by creating indexes after data has been written without prior influence on how that data should be consumed.

SIMBA Chain is a platform that simplifies the management of a Web3 application's lifecycle and takes an alternative approach by enabling design-time specification of transaction interrelationships. SIMBA enables smart contracts to annotate data relationships between smart contract methods to effectively enable schemas to be defined a priori within the source code. By using these relationships, it then autogenerates a GraphQL schema for convenient consumption. We discuss the pros and cons of such an approach and present three application use cases in coffee tracking, NFT car titles and a supply chain scenario, where the approach was extended to also connect and search data across multiple blockchains.

Ian Taylor
SIMBA Chain Co-founder and CTO

Prior to SIMBA, Ian Taylor served as the CTO of ASB Consulting and CIO of Energy Intelligence Worldwide Corporation, both exemplifying his wealth of knowledge in architectural design, algorithmic implementation, and technological innovation. With a strong tie to academia, Taylor holds a full research professor position at Notre Dame and a professor position at Cardiff University, UK.

Alongside his CTO role, Taylor acts as President of Cleverfish Software, offering consultancy, development and research services in the distributed computing and Web sectors. Taylor holds a BSc in Computer Science and Ph.D. in Psychoacoustics, Neural Networks and Computer Science from Cardiff University.

1. Introduction

Fundamentally, blockchain's enable a decentralized mechanism for the recording of non-repudiable transactions so that no single entity has control of the data. This is achieved using a distributed consensus algorithm that results in immutable data, making it impossible for data to be tampered with once written. Ethereum[1] is an example of such a blockchain which has a network of Ethereum Virtual Machine (EVM) connected nodes. Transactions are coordinated using smart contracts, written in solidity, which define a data and logic interface to the underlying ledger.

In general, blockchain is the first technology that supports a decentralized point-to-point transition of any digital asset (token) from one entity (or state) to another without using a central coordinator. While cryptocurrency uses this to transfer monetary value, there are many other use cases where support for unhackable transitions of assets is game changing. Smart contracts and NFTs (ERC721[2]) can be used to digitally track data assets (e.g. IP, legal documents, players in a game) or real world assets (e.g. supply chain, healthcare) to secure and/or optimize business processes. NFTs, for example, can provide digital proof of ownership for anything we own or interact with. Therefore, blockchain is suited where multi-step transactions need verification and traceability.

There have been a plethora of applications deployed on public blockchains, e.g.[3], in the past several years and consequently the capability of analyzing this data has emerged as an important topic of research. However, there is a high overhead to enter into this space because blockchains encode data and the encoded data has meaning that can only be understood if you also understand the smart contract that was used to write it[4]. Therefore, even to read data, significant technical barriers have to be overcome. Over and above this, smart contracts define the behavior of the application, and also define the implicit data relationships between blockchain transactions which store the application's state. However, these relationships are not explicitly defined by a custom application. Of course standardized smart contracts have defined behaviors e.g. mint, transfer for an NFT, but beyond those standards it becomes extremely difficult to extract those relationships automatically. As we will discuss, existing approaches use manual curators or custom analysis to extract the data in a way that is usable for analytics.

In this paper, we will review some of different types of approaches for indexing data and compare it with the approach taken by SIMBA Chain[5], which takes a more proactive stance by enabling developers to explicitly ingrain those relationships into their smart contracts so that a schema for extraction can be automatically constructed for querying. We discuss this approach in detail and outline its advantages and disadvantages. We then provide three practical real-world use case examples where this technique has been used.

The rest of the paper is organized as follows. The next section discusses existing Web3 indexing approaches. Section 3 provides a background on SIMBA Chain and Section 4 describes the indexing approach that it takes. Section 5 provides three real-world use examples that make use of this approach, including tracking bags of coffee in Mexico, managing car titles and registrations using NFTs, and a multi-chain supply chain scenario where an extended relationship graph enables search across these chains. Section 6 provides a future view for this research area and Section 7 concludes. Section 8 acknowledges colleagues that contributed to SIMBA Chain.

2. Web3 Indexing Approaches

One of the key needs for next-generation Web3 applications is to index and consume Web3 data so that full provenance traces can be queried for each asset across its lifecycle. For cryptocurrency, this is reasonably straightforward because standard interfaces are exposed for transferring tokens, which can be captured to extract transactions between different wallets. However, even in this case, on-chain data needs to be decoded according to the smart contract interfaces to recreate machine readable dataset for analysis. Furthermore, some applications have tens of smart contracts interacting in complex ways with complex data relationships between methods and contracts. To analyze data there are two overarching approaches:

AFTER THE FACT: Where schemas are built and data is indexed after data is stored on a blockchain, without pre-chain design strategies.

BEFORE THE FACT: Where relationships and schemas are built beforehand, perhaps into the source code, before data is stored on the blockchain.

For the first category there are many companies that offer consultancy services providing customized datasets for Web3 applications. Chainalysis[6] is one such popular service that operates in this way but there are many more. However, extracting information from a blockchain is not new, with efforts starting as soon as the difficulties of parsing the blockchain's structure became apparent. Early initiatives, such as EtherQL[7], attempted to provide an efficient query layer for Ethereum, for analyzing blockchain data, including the ability to perform range queries and top-k queries, which can be integrated with other applications.

A notable and more recent example in the Web3 community is "TheGraph" [9], which provides a decentralized incentivized mechanism for indexing and querying blockchain data. TheGraph makes it possible to query data that is difficult to query directly by enabling curators to create "subgraphs", which define the data TheGraph will index. In other words, curators take time to understand the transactions and how they relate to each other in order to create a GraphQL schema and mapping that others can use. Anyblock[10] is another tool that employs the use of Elasticsearch and PostgreSQLto support to search, filter and aggregate across multiple Ethereum based blockchains. However, the data would need to be curated and tied together for this to work, so domain knowledge of the structure and purpose of the application would be needed by the curator of this information. A survey of a number of other different methods is provided by Akcora et al.[8], but they all offer after the fact indexing.

3. SIMBA Chain

Incubated at the University of Notre Dame in 2017, SIMBA Chain (short for Simple Blockchain Applications) is an enterprise development platform to bridge and connect to Web3. SIMBA Blocks is the core offering, abstracting the complexities of blockchain development to make Web3 accessible to all. Blocks provides a low-configuration environment that auto-generates REST APIs for smart contracts on multiple blockchain protocols. Developers can choose and migrate between public, private, and hybrid chains, and optimize and future-proof their Web3 applications.

Blocks has 12 components that support and simplify blockchain integration, see Figure 1. The auth layer wraps the platform to provide bindings to authentication and authorization frameworks, including Auth0, Active Directory and for the DoD, CAC cards. The Blocks model enables a graph of relationships to be created, linking smart contracts, versions and blockchain (see next section). The other components provide enterprise grade tooling for resilient, scalable and sustainable blockchain applications.

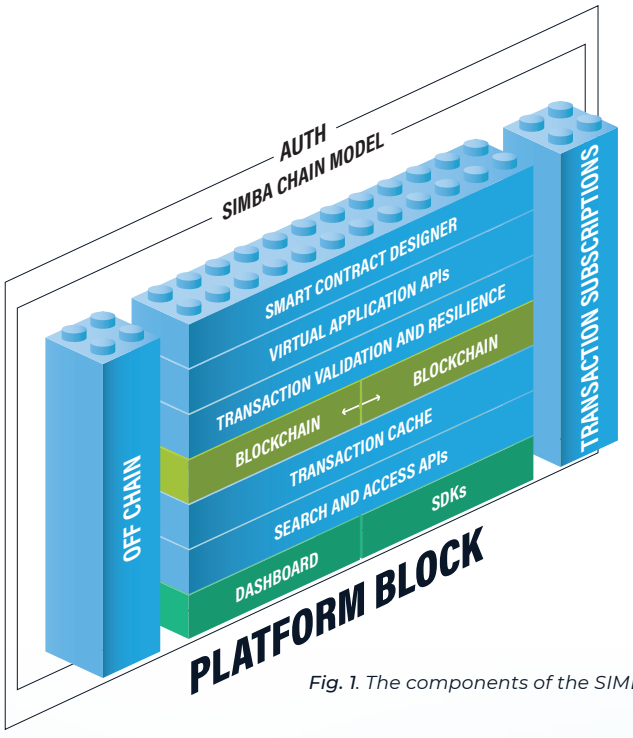


Fig. 1. The components of the SIMBA Chain Platform.

SIMBA provides a generic API to multiple blockchain systems so the system does not have a dependency on a single blockchain or distributed ledger technology. SIMBA currently supports Ethereum[11], Quorum[12], Hyperledger Fabric[13] and Polygon[14]. It has previously interfaced with Stellar[15], RSK[16], Binance[17], Ava Labs Avalanche[18], Hyperledger Burrow[19] and Hyperledger Sawtooth[20].

Smart contracts can be deployed onto any supported blockchain and an API is generated. Smart contract methods are exposed as REST endpoints using POST and GET with payloads mapping to method parameters, providing a simple, well known interface for developers. APIs are virtual and auto-generated and with no code to manage, Blocks is scalable. APIs are fully documented using Swagger or ReDoc, the Swagger interface being shown in Figure 2.

Blocks Smart contract method and parameter validation ensures payloads are valid, fixing minor errors and returning major ones before posting to blockchain. Blocks uses an intelligent nonce management mechanism using distributed locking and caching. The chain watcher component uses asynchronous queues and topics, where each transaction can be monitored for completion and trigger further processing such as notifications. When transactions are processed, Blocks adds them to the database Transaction Cache. This rigorous approach ensures resilience and supports lightning fast data search.

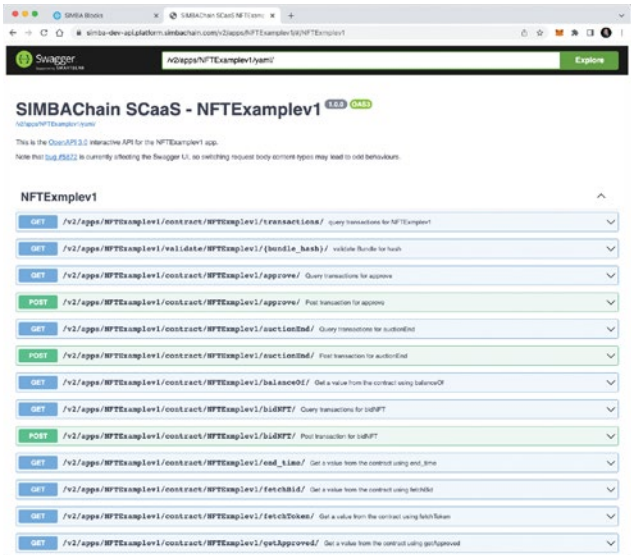


Fig. 2. *A Swagger generated interface to the REST API for a smart contract in SIMBA.*

Blocks also provides support for off-chaining data. Files attached to a POST are bundled, with each file stored and referenced, using its hashcode, in a manifest file, which is also stored off-chain, with its hashcode stored on the blockchain. The system can retrieve files by using the hashcode to retrieve the manifest, and then using each hashcode to retrieve the files. Currently, SIMBA supports Ceph[21], IPFS[22], Azure Blob Storage[23], Amazon S3[24], and flat files systems, with further data stores being simple to integrate.

SIMBA Blocks has a sophisticated notification system that can attach triggers to smart contract transactions and notify by email, SMS or using a Web endpoint to connect to other systems. SIMBA supports configurable targets for authentication e.g. SMS, SMTP and filters can be chained to create logic expressions to notify on specific situations e.g. notify if (id=8756) & (name=Ian). Finally, SDKs make client development simple and streamline client-side transaction signing. SIMBA supports multiple programming languages with SDKs in Python, Java, Node and .NET. This code snippet uses the Node SDK to send a signed data payload to a method on a smart contract hosted on Quorum.

4. SIMBA's Indexing Approach

As described in Section 2, in the Web3 community there are decentralized indexing systems that can be choreographed using data that is already written but there lacks a more proactive design- and implementation- time approach, which can embed data relationships across assets and be used to understand the semantics of the assets across the Web3 application.

The SIMBA Graph Model (SGM) approach is designed to address these issues. Complex relationships within and between assets can be stored using SGM, curated in the smart contracts and then schemas can be automatically generated for complex graph-based querying at consumption. The SIMBA Chain model conceptualizes an application's data and relationships using business process concepts; assets (nouns) represent digital entities and transitions (verbs) representing transitions of assets.

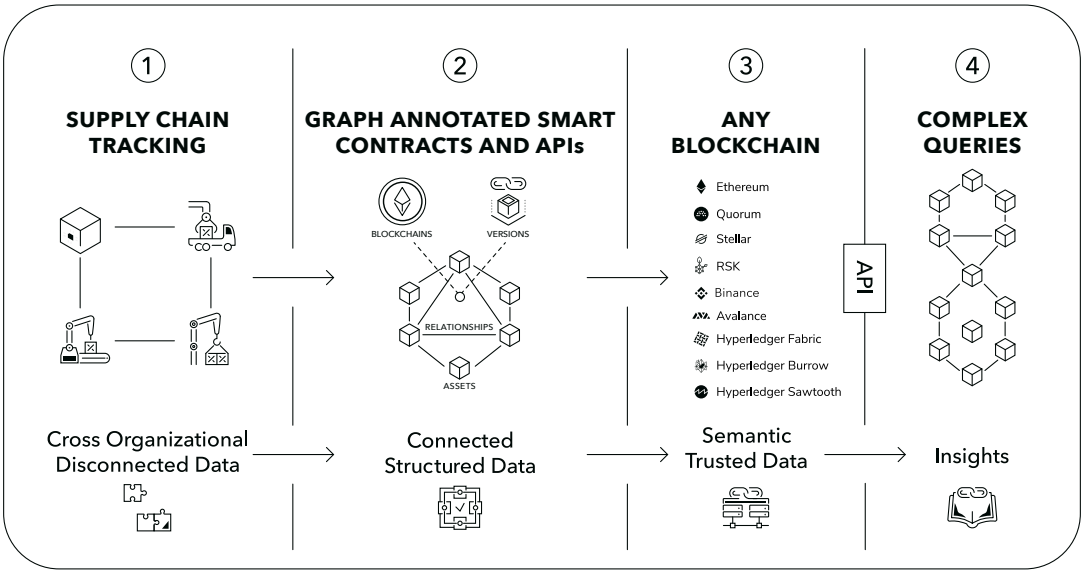


Fig. 3. A supply chain scenario to illustrate how the SIMBA model process works

To illustrate this approach, Figure 3 shows how a package can be tracked across a supply chain. Using Blocks, the relationships between the package and how it is tracked, shown on the left, are built into smart contracts using annotations and stored on the blockchain. Using these relationships, SIMBA can generate a schema automatically, using GraphQL. This enables application data and their relationships to be defined up front in the code, and queried across those relationships.

This model is used in the Smart Contract Designer (SCD) UI low-code smart contract development environment, and in the dynamic creation of the REST based application-oriented APIs. Using this approach blockchain transactions can be searched using Graph queries. Figure 4 shows an example of this UI being used to create a complex supply chain use case, with the red rectangles representing the assets, the blue ellipses representing transitions and the lines specifying the relationships between assets and assets or assets and their transitions. Each entity can be double clicked to specify the parameters to record at each stage.

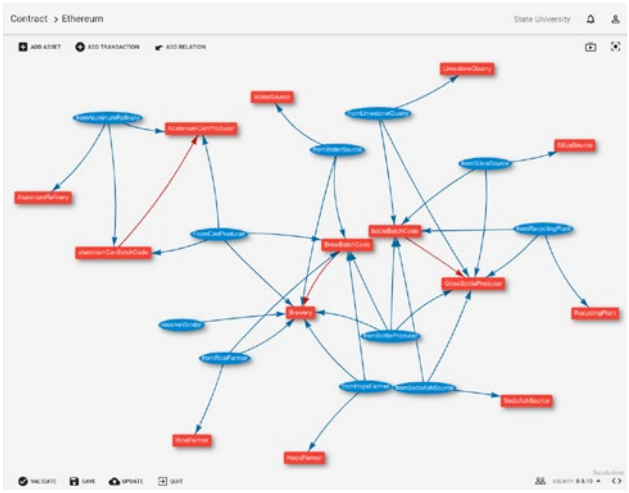
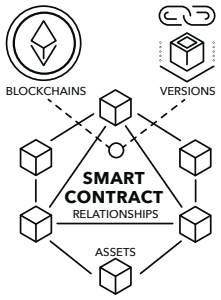


Fig. 4. A supply chain asset graph created in the smart contract designer tool



The Blocks graph can also represent smart contract versioning; each version is linked with the previous version and forms a queryable graph that can access data from all smart contract versions. Graph links can further connect smart contracts from one blockchain to another. In this way, future proof sustainable blockchain applications can be built, with the graph capturing the application's evolution; one search can potentially traverse the same application co-hosted on several blockchains with multiple smart contract versions.

The SGM can be specified using the SCD tool or can be added manually to smart contracts, to enable a coder to specify data relationships within assets, between assets to support far richer asset transition. Furthermore, using the SCD such relationships can used used at a higher level and can be used to specify the smart contract standards to be used e.g. ERC721, along with the asset's data relationships, and SIMBA is capable of generating a template "blocks package", which consists of fully audited ERC721 contracts with inter-asset relationships baked in, meaning that the design level data query schema for the entire Web3 application can be specified at it's implementation, providing rich querying capabilities for consumers automatically.

Blocks currently supports mappings for GraphQL and has prototypes for exporting to graph databases. The GraphQL query models these data relationships and can be used to directly extract the data and relationships and allows a consumer to query the data in exactly the way the developer exposed it for consumption.

The advantages of this approach are as follows. First, it enables a developer to design the data analytics schemas and interfaces specifying optimally how their Web3 application data is consumed. This allows efficient querying across an asset's life cycle and furthermore, enables the capturing of evolving data states within those assets that may not fall into the standardized interfaces provided. These Blocks smart contracts can be deployed onto any SIMBA supported blockchain and transactions stored in this way can be searched using GraphQL across data, contracts and chains. This approach will enable advanced standards-based NFT applications representing digital or physical assets to be created and queried more simply. It also will provide an advanced means of querying the data from those tokens.

The limitation of this approach is that it requires up front annotation of data and smart contracts. We'll discuss in the future work section how we are thinking about addressing these issues.

5. Use Cases

5.1. Tracking Bags Of Coffee for Sustainability

In January, 2021, small farmers in the Tacaná Natural Reserve, a volcano biosphere boarding Mexico and Guatemala, started using a secure supply chain. The problem solved was an inefficient, and sometimes unfair, supply chain. Using a blockchain coupled with physical attributes, the outcome has been a secure supply chain from individual farmers to more than 200 Toks Restaurants across Mexico.

The process is described as follows. A bag of coffee (about 100 kg) is delivered from a farm to the cooperative. The cooperative uses an app on a tablet to register the bag of coffee, measuring the weight and humidity amongst other attributes, and then the farmer digitally signs to hand it off in return for payment. A QR code for the bag is generated and registered on the blockchain. The bag of coffee then is sent to the desheller (also registered on chain) and when it returns, the new weight and humidity is measured again. The new weight will be roughly 70Kg but each bag is different. Also, the humidity for each bag will vary too. These new measurements are recorded on the blockchain and associated with the QR code.

This coupling of the physical attributes makes counterfeiting extremely challenging. A counterfeiter cannot simply copy the QR code onto a fake bag of deshelled coffee because it would be very unlikely that the weight and humidity would match. This scheme therefore provides authenticity and a robust anti-counterfeit mechanism. For the restaurant customers, the secure supply chain ensures the coffee is sustainably sourced and for the farmers, this ensures that counterfeit coffee cannot enter into their supply chain.

This tracking of the coffee at different supply chain locations forms the asset (bag of coffee) and transition (how the state of that bag changed) relationships across the supply chain. With SIMBA Chain, we capture these relationships and encode them into smart contracts. The result is that TOKs can query the entire provenance of a bag of coffee with a single query, using GraphQL under the hood, which will return every touch point for that bag of coffee across the supply chain.

5.2. Car Titles And Registrations

Tracking automotive sales, resales and titles use archaic paper systems that are riddled with errors and fraud. SIMBA Chain has been working with California auto dealers to track titles across the vehicle life cycle using VIN numbers and tokenization technology.

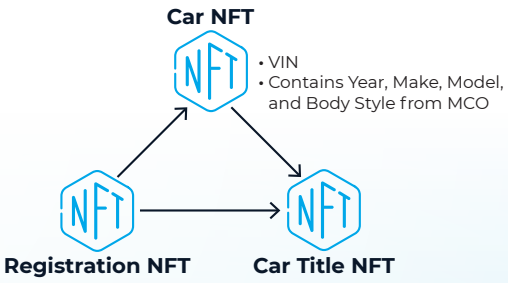


Fig. 5. The three NFTs used to model a car, its title and the registration.

As shown in Figure 5, we used three NFTs to represent the different components of the application. A car NFT represents the car itself by using the VIN number as the NFT identifier. A car title NFT represents the information of the title but also has a relationship with the car NFT to tie back to the original entity. A registration NFT contains the registration information, which ties back to the car NFT and also references the car title NFT information. This is a simple scenario but it shows how complex relationships can become for real-world applications. Car titles may only need 3 NFTs but if you factor in liens, leases, and so forth it can get more complex. Imagine what an NFT structure for the purchase of a house might look like.

5.3. Supply Chain across Multiple Blockchains

In supply chains, applications can include tracking parts from creation to installation to ensure their transition across the supply chain is fully traceable and verifiable, to make counterfeiting impossible and ownership, including the origins of those parts, fully transparent. And since a blockchain is immutable, it makes it impossible for anyone, including adversaries, to tamper with the data. Figure 6 shows a supply chain scenario which connects three blockchain networks using the same GraphQL schema interface that we use for the querying of method and contract relationships.

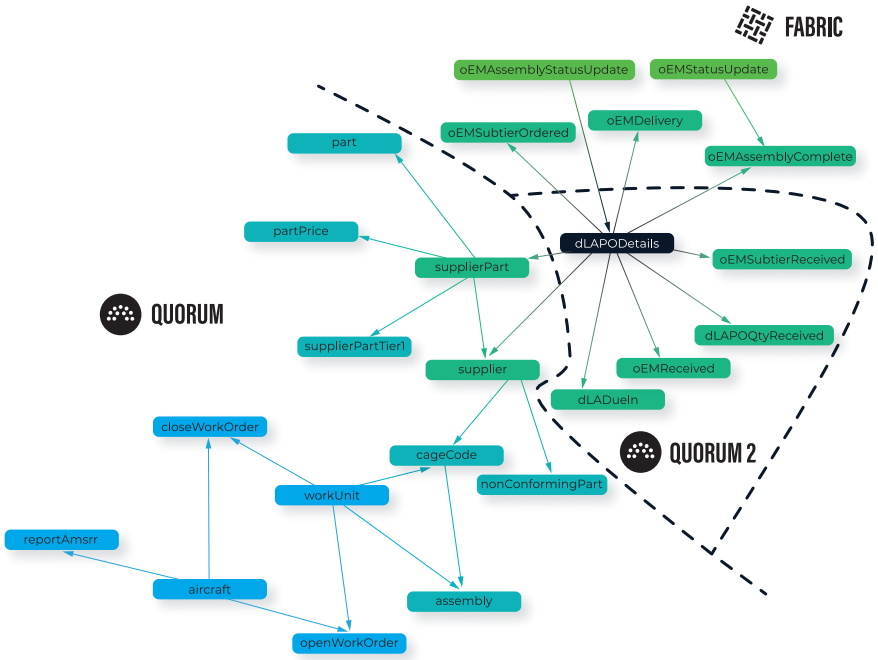


Fig. 6. Shows how we can aggregate information from three different sources stored on different blockchain networks into a single GraphQL query.

To achieve this, we essentially extend the graph to include a connection from a parameter on a contract on one blockchain with a parameter on a contract on another. In the example in Figure 6, we connect a PO order from one system on Quorum with a PO number on a contract on a supply network on Hyperledger Fabric to track the progress of the production of that part. We then connect the Cage code of the supplier and part to the SupplierPart method on a contract on a further Qurum network to map those to resolvable public data from PUBLOG[25]. This enables us to embed extra part information and supplier details into a single query, just by simply connecting different applications on different chains together.

6. Future Directions

In the future, we imagine far more complex NFT interactions, where each NFT could represent a digital or physical object with evolving internal states as they transition from one state to another; imagine a complex supply chain for an aircraft wing that contains multiple assemblies each having subassemblies, which are each manufactured by a different supply chain tier. Such interactions require complex modeling of the internal data state and how that state impacts other assets as they move around the supply chain. NFTs can represent assemblies and can be linked through to subassembly NFTs to create a structured tracking mechanism for each step of production, and each supply chain tier.

In fact, NFTs in the future could represent anything we own or legally bind ourselves to, from cars, homes, or even, as some suggest, marriage [26], or digital assets like data files, IP usage, or any contractual exchange. NFTs could also represent physical assets by binding physical to digital codes (e.g. RFID, QR Codes, NFC, or image feature prints) tracked using blockchain, to enable the tracking of any physical parts (e.g. wings, tailhooks etc).

The above approaches provide some underlying techniques for the curation and querying of blockchain relationships. However, GraphQL is not a flexible graph model, it rather provides a schema to access information. This means that multi-level relationships, for example where wallets transact with wallets across several levels, are not possible to implement, because recursion is not supported. What is needed is a full graph database approach that can support recursion to provide the capability of creating a global graph of cryptocurrency and blockchain interactions and relationships.

Another area which we are investigating is the idea of bypassing the annotations for smart contracts and looking at using code inspection to automatically extract relationships or execution trails. This is still under investigation but has the potential to provide the power of the annotation approach without needing to do this manually. If these behaviors can be dynamically built and embedded into a revolving schema, this could provide a more turnkey approach for this in the future.

7. Conclusions

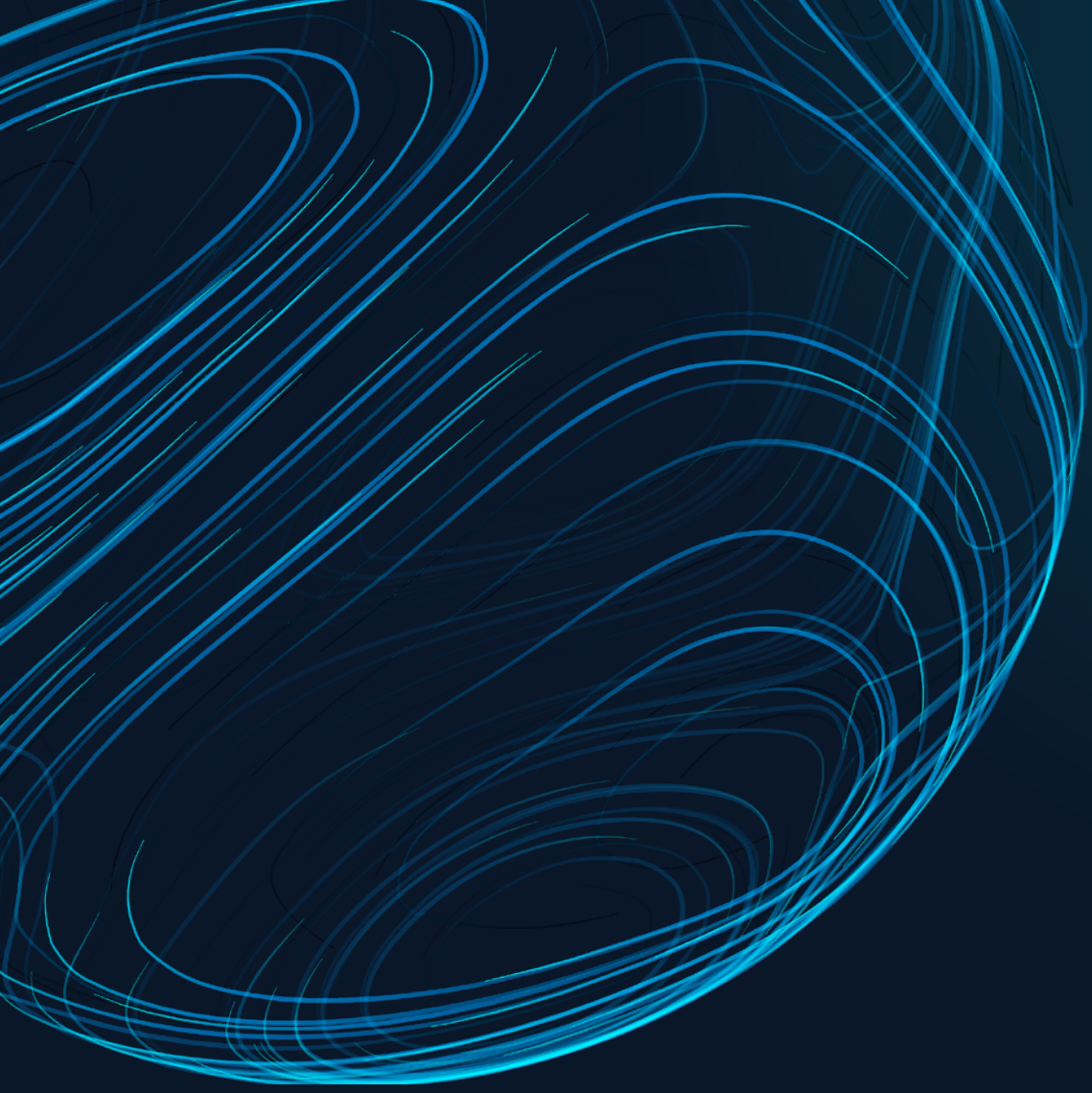
In this paper, we discussed the existing Web3 indexing approaches and compared those with the approach that SIMBA Chain takes, offering a “before the fact” model rather than trying to retrospectively retrofit a schema for reading the data. We provided a background into SIMBA and discussed this indexing approach in detail and how it differed from the state of the art. We then provided three real-world use examples that make use of this approach, by illustrating how each benefitted from having a single query to consume the data. These use cases included tracking bags of coffee, managing car titles and registrations using NFTs, and an extended supply chain scenario that uses this approach to enable search across three blockchains.

8. Acknowledgments

Thanks so much to the SIMBA team for realizing the vision of the SIMBA Chain platform and product.

9. References

1. G.Wood,“Ethereum:A secure decentralised generalised transaction ledger - e94ebda,” 2018, https://github.com/ethereum/yellowpaper/
2. NFT ERC721 Standard, https://eips.ethereum.org/EIPS/eip-721
3. 34 Blockchain Applications and Real-World Use Cases, https://builtin.com/blockchain/blockchain-applications
4. Brinckman, Evan; Kuehlkamp, Andrey; Nabrzyski, Jarek; Taylor, Ian J. Techniques and Applications for Crawling, Ingesting and Analyzing Blockchain Data, Proceedings of the 2019 International Conference on Information and Communication Technology Convergence (ICTC), pages 717-722, 2019
5. SIMBA Chain, https://www.simbachain.com/
6. Chainalysis, https://www.chainalysis.com/
7. EtherQL: Yang Li, Kai Zheng, Ying Yan, Qi Liu & Xiaofang Zhou, EtherQL: A Query Layer for Blockchain System. Part of the Lecture Notes in Computer Science book series (LNISA,volume 10178).
8. Akcora, Cuneyt Gurcan, Matthew F. Dixon, Yulia R. Gel, and Murat Kantarcioglu. “Blockchain data analytics.” Intelligent Informatics 4 (2018).
9. The Graph Documentation.URL, https://thegraph.com/docs/
10. Any Block Analytics.URL, https://www.anyblockanalytics.com/docs/main/
11. Ethereum, https://ethereum.org/
12. Quorum, https://consensys.net/quorum/
13. Hyperledger Fabric, https://www.hyperledger.org/use/fabric
14. Polygon, https://polygon.technology/
15. Stellar, https://stellar.org/
16. RSK, https://www.rsk.co/
17. Binance Chain, https://www.bnbchain.org/en
18. Ava Labs Avalanche, https://www.avalabs.org/
19. Hyperledger Burrow, https://www.hyperledger.org/project/hyperledger-burrow
20. Hyperledger Sawtooth, https://www.hyperledger.org/use/sawtooth
21. Ceph, https://ceph.io/en/
22. IPFS, https://ipfs.tech/
23. Azure Blob Storage, https://azure.microsoft.com/en-us/products/storage/blobs/
24. Amazon S3, https://aws.amazon.com/s3/
25. PUBLOG, https://www.dla.mil/Information-Operations/Services/Applications/PUB-LOG/
26. How India’s First Couple Got Married On The Blockchain, https://www.indiatimes.com/technology/news/india-first-couple-marriage-on-blockchain-561474.html



simbachain.com info@simbachain.com

SIMBA