HyperETL: Facilitating Data Analysis of Private Blockchain

Vladimir Ivković*, Aleksandar Hadžibabić*, Slavica Kordić*, Ivan Luković**
*Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia
**Faculty of Organizational Sciences, University of Belgrade, Belgrade, Serbia
{vladimir.ivkovic, hadzija, slavica}@uns.ac.rs, ivan.lukovic@fon.bg.ac.rs

***Abstract*—In recent years, blockchain technology and smart contracts gained popularity in many industry fields. Industry leaders are interested in private blockchains that are more suitable for enterprise applications. An application of blockchain brings the unprecedented capability to monitor and execute collaborative business processes (CBPs) in finance, trade, insurance, healthcare, and other fields. CBP activities can initiate smart contracts' actions. That results in creating transactions and storing data objects of interest for a particular business process. Organizations involved in blockchain networks can benefit from analyzing transaction data in terms of improving blockchain systems and increasing knowledge about CBPs. However, blockchain platforms store transactions in data formats that are highly optimized for blockchain operations. Such formats are not suitable for ad hoc querying or data analysis. Therefore, organizations need a proper way to query and analyze data generated within blockchains. In this paper, we propose HyperETL, an ETL-based approach for accessing, extracting, and transforming blockchain data into a platform-independent format suitable for data analysis. We focus on private blockchain frameworks from the Hyperledger foundation. We also present a prototype solution that follows the proposed approach. Such a solution would help data analysts query data and derive valuable insights on CBP activities in order to analyze CBPs and improve decision-making. Furthermore, extracted and transformed blockchain data can be integrated into existing data warehouse systems.**

***Index Terms*—Blockchain, DLT, Hyperledger, ETL, data warehouse systems, data analysis**

I. Introduction

In the last decade, blockchain technology along with smart contracts showed the capability of facilitating collaborative business processes (CBPs) in finance, trade, insurance, healthcare, and other fields [1]. CBPs are the ideal setting in which blockchain technologies can be adopted and exploited to their full potential [3]. Organizations collaborate with other participants in the CBP to be competitive in the market and blockchains provide a shared, immutable, and transparent data storage making them a valid technology to support such processes. Even though blockchain technology became popular when Bitcoin and public blockchains emerged, nowadays, private blockchain seems to be more suitable for enterprise applications [2]. In a private blockchain, a network is defined in advance and all members are known to each other. In such a network, members are organizations that are involved in CBP. All data required for the execution of CBP is shared and stored transparently in the blockchain system.

A blockchain is a protocol for the distributed storage of a tamper-proof sequence of transactions, maintained and verified by the nodes participating in the network [4]. The atomic unit of computation in a blockchain system is a transaction, which records changes on the blockchain (e.g. transfer of a digital asset, update of particular states). The sender cryptographically signs the transaction to provide evidence that it is not counterfeit. Smart contracts facilitate communication and actions on the blockchain network and produce transactions grouped in blocks. A smart contract was originally defined as a digital representation of an agreement between multiple parties that declares which actions should be executed when certain conditions are met. Technically, a smart contract is a program that manipulates states. Each state has an identifier and value and represents an object of interest for the application. Creation, deletion, or modification of one or more states, using a smart contract function, results in a transaction. Various private blockchain platforms, usually called Distributed Ledger Technologies (DLTs), utilized different approaches for storing and accessing transaction data. That makes transaction data hardly accessible outside of the blockchain system.

Smart contracts that implement CBPs manipulate data objects of interest for a particular business process. For example, in an insurance blockchain network, a customer can file an insurance claim for a stolen car. Other participants, such as insurance companies and public authorities can process the insurance claim through the smart contract. Thus, data and business analysts would like to have an insight into what smart contracts do and how that is related to internal (private) business processes. Those insights can help in deriving valuable business knowledge and improve decision-making in an organization. Furthermore, derived knowledge can help in addressing pain points in the collaboration between participants. On the other hand, data analysts are not interested in technology-related details. Instead, they only need to have an opportunity to examine transaction data from a business perspective.

In order to support the application of blockchain technology and encourage organizations to evolve toward blockchain, Hyperledger Foundation[1] hosts several enterprise-grade blockchain software projects, and some of them are dedicated to building distributed ledgers. Another project,

[1]https://www.hyperledger.org

Hyperledger Explorer[2], is a tool for visualizing the blockchain operations of the Hyperledger platforms. It offers browsing blocks and transactions through a web user interface. Although Explorer can be a useful tool during the blockchain development process, it does not support the execution of complex queries that are necessary for data analysis. On the other hand, the need for learning from data through different techniques of data analysis and machine learning is rapidly growing in all industry fields. Blockchain is no exception and it becomes another data source available to derive valuable information on smart contact activities such as which members are more active than others, the frequency of submitting transactions, etc.

In this paper, we present Hyperledger Extract-Transform-Load (HyperETL), an approach to extracting transaction data usually stored in various platform-specific formats, transforming and loading it into a target database. The target database will not contain platform-specific concepts. Such a database would serve data analysts as a data warehouse database and offer to ask complex questions through SQL queries. Having in mind that SQL is the most widely used query language implies that even non-technical users can execute queries about CBPs and smart contracts deployed on the blockchain. Moreover, if some organizations already have a data warehouse system and business intelligence (BI) solution, the target database can be integrated into the existing data warehouse system. The remainder of this paper is structured as follows. In Section II, research challenges and potentials are listed and discussed. Section III contains present related works available in the literature. Sections IV and V describe used methodology and a proposed solution based on that methodology. A case study is presented in detail in Section VI. The paper concludes with Section VII including implications for further research.

II. Research Questions

Obtaining and querying data from the DLT platform can be challenging due to several reasons. First, in the permissioned blockchain network, transaction data is not publicly available. Data access is restricted to network members through authentication and authorization mechanisms. The data extraction process will include an authentication step, and appropriate data access privileges need to be provided by a network member that is interested in blockchain data analysis.

Second, each DLT platform has a specific storage format including various binary file formats or key-value stores. Retrieving raw blockchain is often supported by custom command-line interface (CLI) applications or application programming interfaces (APIs) for accessing transaction data. Further, DLTs only have limited capability in querying transaction data and usually include only block or transaction retrieval based on a provided identifier. Some blockchain frameworks, such as Hyperledger Fabric[3], offer listing history for a particular state. Querying blockchain data by other attributes including timestamp, involved members, transaction methods, or input arguments is not supported. The majority of blockchain networks do not provide SQL-like query language for the execution of ad-hoc queries directly on blockchain data.

Due to the lack of querying capabilities of transaction data in DLT platforms, data analysis could be a cumbersome process without a proper approach and a tool for extracting and transforming raw blockchain data. Our goal is to make transaction data available for analysis using a tool that will not require any platform-specific knowledge from a data analyst. After the extraction of transaction data, which usually requires the authentication step, the main challenge is to understand the structure of blocks in a raw form. Blockchain data formats are mainly designed to support storing and securing transactions in a chain of blocks. Transformation of raw data will include decoding data and retrieving various objects and attributes of interest such as changed states and their values, timestamps, etc. Finally, transformed data should be loaded into a target database and become available for asking arbitrary questions through SQL queries. To design the HyperETL approach and implement a prototype solution, we need to address the following research challenges:

- to define a consistent way of accessing blockchain data,
- to provide adapters for each blockchain platform of interest, and
- to design high-level data transformation procedures from platform-specific into a platform-agnostic format.

III. Related Work

In recent years, significant research efforts have been invested in addressing the problem of blockchain data access and making that data available for further analysis. Zhou et al. [5] motivated their work by emphasizing limitations in ledger data access in the case of the Hyperledger Fabric framework. They proposed a ledger data query platform called Ledgerdata Refiner that provides sufficient interfaces for users to retrieve blocks and transactions efficiently.

Datachain is a lightweight, flexible and interoperable framework deliberately designed to ease the extraction of data hosted on DLTs introduced by Trihinas [6]. The presented framework and tool follow the Create-Retrieve-Append-Burn (CRAB) approach for blockchain data management originally defined in BigchainDB, a blockchain-based database. Datachain tool supports data extraction from underlying blockchain platforms but does not allow loading extracted data into the target relational database.

Galici et al. [7] presented an ETL-based strategy for collecting data from the blockchain and making it available for further analysis. They focused on Bitcoin blockchain transactions, organized input and output sections of each transaction, and stored them in a relational database. They also implemented an address clustering algorithm specific to the Bitcoin blockchain.

EtherQL is a query layer for Ethereum developed and presented by Li et al. [8]. It offers query primitives for analyzing blockchain data, including range queries and top-k queries, which can be integrated with other applications.

[2]https://github.com/hyperledger/blockchain-explorer

[3]https://www.hyperledger.org/use/fabric

EtherQL can help in overcoming issues with Ethereum's limited data access and it is designed to provide different levels of abstraction, which are suitable for data analysts, researchers, and application developers.

Vinceslas et al. [9] propose an abstraction layer architecture that enables the design of high-level analytics of DLTs and the decentralized applications that run on top. The presented work aims to establish key insights and specifications to improve the auditability and intuitiveness of DLTs by leveraging the development of future user interfaces. They designed two distributed ledger visual representations: transaction-based and account-based abstractions. To illustrate the proposed visual concepts and application architecture, a use case based on an application for the regulated sector has been explored. Although the proposed abstraction layer helps data analysts to become familiar with DLT data through visual representation, it does not provide a way to execute ad-hoc queries by end-users.

In their research, Dillenberger et al. [10] present analytics engines connected to blockchains to provide easy-to-use configurable dashboards, predictive models, provenance histories, and compliance checking. They also describe how blockchain data can be combined with external data sources for secure and private analytics, and enable artificial intelligence model creation over geographically dispersed data. The presented analytic tools are centered on the Hyperledger Fabric platform and leverage the blockchain-specific characteristics of the data such as the transactional history, identity of participants, and timestamps.

The majority of related work offers platform-specific solutions for the analysis of blockchain data. Furthermore, most existing research efforts do not consider a business perspective of smart contact and integration of blockchain data with an existing data warehouse system in an organization of interest. In this research, we aim to develop a unified approach that will facilitate the extraction and transformation of blockchain data into a platform-independent format and make it available to data analysts and other users.

IV. Methodology

The research problems described in the previous sections include a variety of blockchain platforms and data formats for storing transactions. Considering that our approach needs to offer a platform-independent solution, our intention is not to run queries directly on blockchain storage because that will requires a separate query engine for each blockchain platform.

Therefore, our solution will involve data extraction from the original blockchain storage. Platform-specific data, that is not interesting to business intelligence, has to be removed in a cleaning process. In order to reorganize transaction data and present it in a platform-agnostic format, certain transformations need to occur. Finally, blockchain data will be transferred to the target database and become available for analytics.

The described objectives will be achieved utilizing the extract, transform and load (ETL) approach [11]. ETL is a data integration process that combines data from multiple data sources into a single, consistent data store that is loaded into a data warehouse or other target system. Even though ETLs are present for more than three decades they are still relevant in modern data warehouse systems and provide the foundation for data analytics and machine learning workstreams. The main goal of the ETL process is to cleanse and organize data in a way that addresses specific business intelligence needs through a series of business rules. ETL process consists of three major steps:

- extraction - during data extraction, raw data for various internal and external sources is copied or exported from source locations to a staging area;
- transformation - in the staging area, extracted raw data is cleansed, transformed, and consolidated to meet specific data analysis needs; this step includes filtering, deduplicating, mapping, calculations, translations, or summarizations based on the raw data;
- loading - the transformed data is moved from the staging area into a target database.

V. Proposed Solution

The solution proposed in this paper follows the ETL approach. Considering that the blockchain system is the only data source, the extraction step is highly platform-specific and can be unexpectedly complex for implementation. Unfortunately, there are no standards widely adopted in the blockchain world. The same situation is with accessing blockchain data and most platforms have their own CLI tool or APIs. That implies that we need to have an adapter for each platform that performs the necessary authentication to gain access to blockchain data and extract data into platform-specific JSON format. Some platforms, like Hyperledger Iroha, support data export in their own textual format that requires conversion into JSON. Other platforms, like Hyperledger Fabric, export raw block data in binary file format, such as protocol buffers[4] format. In that case, the extraction will include an additional step for data format conversion. Moreover, the extraction module needs to keep track of the latest extracted block while performing periodical extraction. An alternative can be streaming extraction which will send block data as soon as it is committed to the blockchain.

After data extraction is completed, raw blockchain data needs to be transformed and represented in a platform-agnostic format. Common blockchain concepts are identified and a corresponding entity-relationship schema for blockchain data is modeled. The target database schema is shown in Fig. 1. The transformation process consists of two major phases:

- mapping raw block data in platform-specific JSON format into platform-agnostic JSON that only contains common concepts and attributes present in the majority of DLT platforms:
  - block,
  - transactions,

[4]https://developers.google.com/protocol-buffers

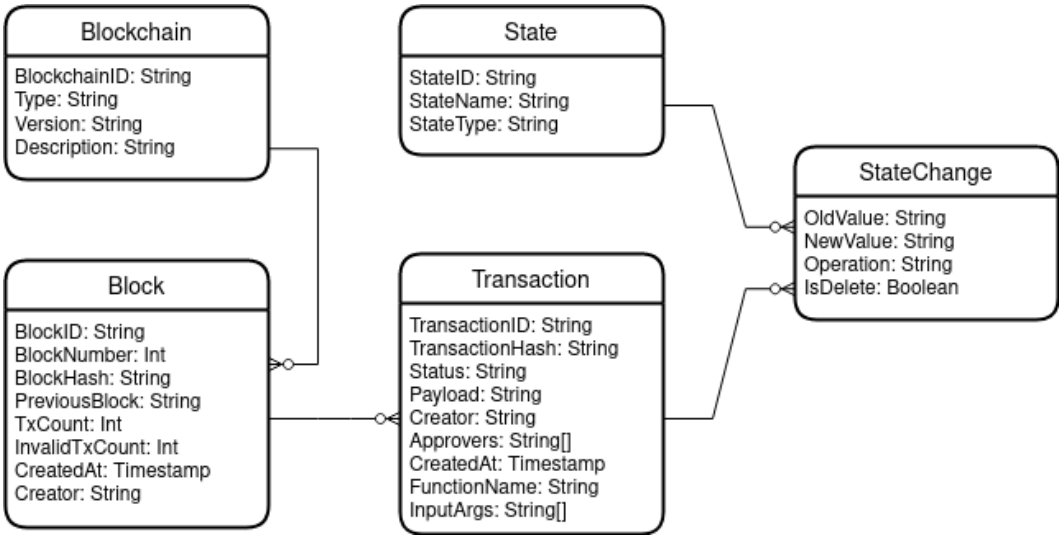

Fig. 1. Target database schema for blockchain data

  - transaction creator and validators - members that created and verified transactions,
  - states recorded on the blockchain, and
  - state changes occurred in a particular transaction.
- transforming blockchain data from JSON format into a target database designed for analytic use cases.

.

The loading step is straightforward and includes populating tables in the target database with transformed blockchain data. The design of the target database schema follows the principles of dimensional modeling and it contains concepts and attributes common to all Hyperledger frameworks. High-level architecture with all ETL components, data sources, and data targets is depicted in Fig. 2.

A tool based on the proposed solution can be realized using various technologies for each step of the ETL process. For the implementation of the HyperETL prototype, we utilized:

- shell scripts as adapters for extracting raw block data,
- python scripts for data transformation,
- Mongo[5] database as a staging area for platform-agnostic JSON documents, and
- PostgreSQL[6] as a relational database management system for the target database.

VI. Case Study

To showcase the usage of the HyperETL tool, we choose IBM's insurance blockchain sample application[7] build on the IBM Blockchain platform which is based on the Hyperledger Fabric blockchain framework.

A. Blockchain Insurance Application

The application demonstrates the use of blockchain in the insurance domain for claim processing and involves four participants: insurance, police, repair shop, and shop peer. The insurance peer is the insurance company providing the insurance for the products and it is responsible for processing the claims. Police peer is responsible for verifying the theft claims. The repair shop peer is responsible for repairs of the product while the shop peer sells the products to the consumer.

Web application offers user interaction with an Insurance, Repair Shop, Police, or Shop Peer. Customer users can buy a phone, bike, and ski through the Shop peer user interface (UI) section. In addition, customers can choose to buy a product with or without insurance contracts that feature different coverage as well as terms and conditions. Finally, a customer will be offered to complete their purchase and agree to the terms and conditions, and close the deal. The whole purchasing process is handled by the smart contract and details are immediately written to the blockchain.

The customer can briefly describe the damage to the product they purchased or select whether it has been stolen. In case the product has been stolen the claim will be processed through the police who have to confirm or deny the theft. The Police Peer can view the claims that include theft. In case the product has been reported stolen they can confirm the claim and attach a file reference number. In case no theft has been reported they can reject the claim and it will not be processed. Once the police peer confirms the claim, the insurance peer can reject the claim or choose to reimbursement option that leads directly to the payment to the customer.

In case there was just damage the claim will be processed through the repair shop. Then, the insurance company can forward the claim to a repair shop and generate a repair order. All transactions that contain changing of states during claim processing will result in a block being written to the chain.

The application is built and run using Docker Compose in a local environment. Then, randomly generated scenarios have been executed including various cases:

- user reports a theft and files a claim,

[5]https://www.mongodb.com
[6]https://www.postgresql.org
[7]https://github.com/IBM/build-blockchain-insurance-app

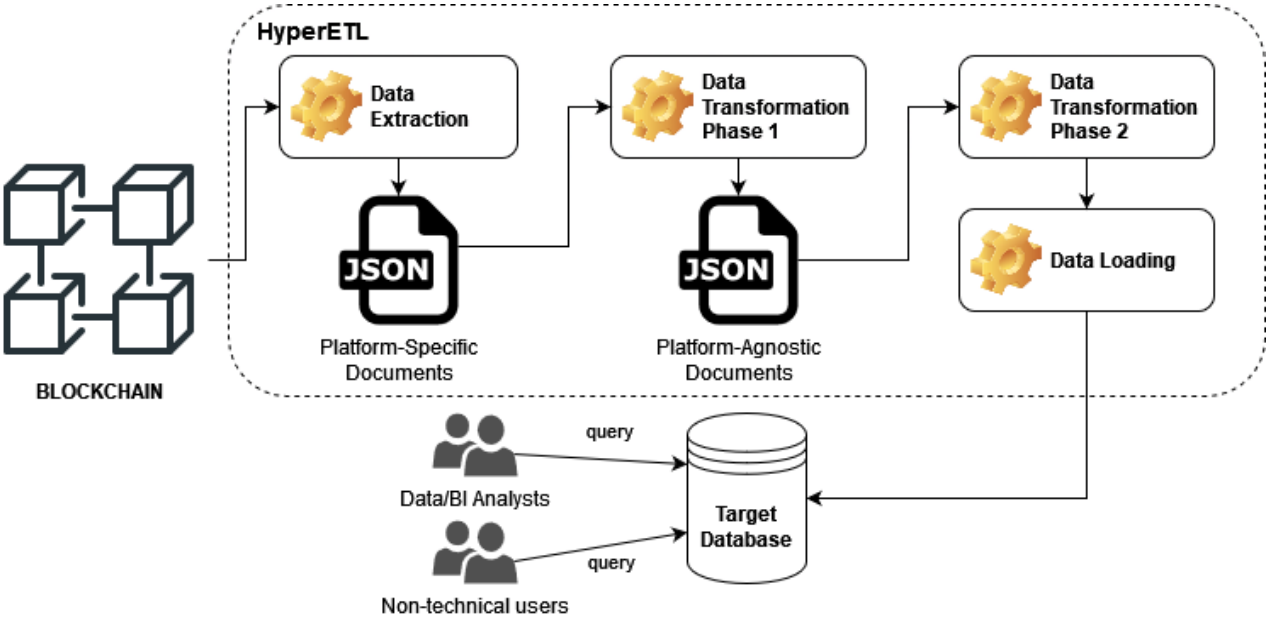

Fig. 2. High-level architecture of the proposed HyperETL solution

- user reports damage on product and files a claim,
- police resolve theft claim,
- workshop completes repair of damaged product,
- insurance reimburses the insured users, etc.

B. Blockchain Data Analysis

The blockchain data generated during the execution of the described scenarios are retrieved using Hyperledger Fabric's CLI tools for fetching raw blocks `peer channel fetch` and translating them to JSON format `configtxlator proto_decode`. The extracted data is then utilized as a starting point for the HyperETL tool[8]. Block data in form of platform-specific JSON documents are cleaned and transformed into a platform-agnostic JSON schema. Many details related to the internal organization of raw blocks are removed and some complex nested data structures are mapped to simpler ones. Then, clean and transformed JSON data is utilized for loading a target database.

When the blockchain data is loaded into a target database that provides SQL-like language, end-users are in a position to execute arbitrary complex queries. In the case study, we divided those queries into two main groups: blockchain-related (1) and CBP-related (2) reports. The blockchain-related reports will include various insights about how blockchain network works, for example:

- how many transactions a block has on average,
- how many state changes take place in a transaction,
- how long it takes between network member submits transaction until it is committed to the block,
- what are transaction trends per creator, method, or state changes, etc.

[8]https://github.com/vladimirivkovic/hyperETL

The sample dashboard, shown in Fig. 3, contains blockchain-related reports, e.g. a time-series line chart for transaction counts by the organization and a horizontal bar chart for transaction distribution by transaction method. The second report is based on the simple SQL query:

```
SELECT method, count(*)
FROM transactions GROUP BY method
```

The second group of reports is more specific to the CBP and its corresponding smart contract that is utilized. For the insurance company such reports could be some of the following:

- how many products are bought without an insurance policy,
- which products are most frequently being stolen or damaged,
- what is the percentage of false theft claims,
- how long it takes for a product to be repaired, etc.

Answers to these or similar questions can help data analysts in the insurance company to better understand customers' needs and habits, as well as the collaboration with the police and the workshop. Finally, those insights might lead to a better strategy that will offer more suitable insurance contracts and plans for the majority of customers.

VII. Conclusion

In this paper, we proposed HyperETL, an ETL-based approach for accessing, extracting, and transforming blockchain data into a platform-independent format suitable for data analysis and further data integration. We focused on private blockchain frameworks from the Hyperledger foundation. We also present a prototype solution developed using the proposed approach. Such a solution would help data analysts query data and derive valuable insights on CBP activities in order

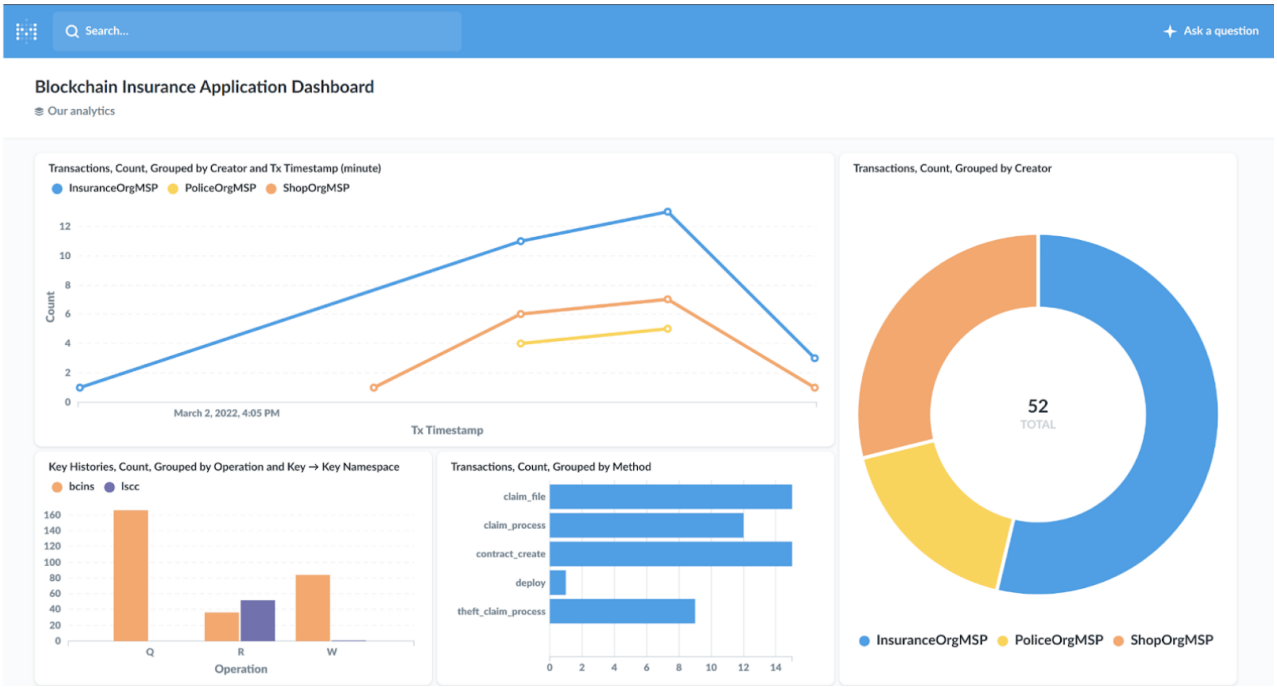

Fig. 3. Sample dashboard for Blockchain Insurance Application

to analyze CBPs and improve decision-making. Furthermore, extracted and transformed blockchain data can be integrated into existing data warehouse systems.

Some of the main benefits of applying described solution are making hardly accessible blockchain data easier to query and analyze (1), and integrating blockchain data with internal data sources into a unified BI solution (2). Transformed and loaded blockchain data will be organized in such a way that will offer arbitrary complex queries. The main goal is to provide data analysts with better insights on historical changes of states, the actual life-cycle of states, activities of other blockchain network members, blocks and transactions from a time perspective, and all kinds of activities initiated by examined smart contracts.

Future research efforts can continue in several directions. First, the HyperETL solution should be extended to support other popular blockchain platforms (e.g. Quorum, R3 Corda). To achieve that, adapters for data extraction need to be implemented for each platform of interest. Another research direction could be identifying and describing common blockchain-related BI reports. Such reports would contain block and transaction generation trends, breakdown of particular member activities, or various statistics about states and their changes.

ACKNOWLEDGMENT

This paper has been supported by the Ministry of Education, Science and Technological Development through the project no. 451-03-68/2022-14/200156 “Innovative scientific and artistic research from the FTS (activity) domain.”

REFERENCES

[1] F. Casino, T. K. Dasaklis, and C. Patsakis, “A systematic literature review of blockchain-based applications: Current status, classification and open issues,” Telematics and Informatics, vol. 36, pp. 55–81, Mar. 2019, doi: 10.1016/j.tele.2018.11.006.

[2] I. Weber, X. Xu, R. Riveret, G. Governatori, A. Ponomarev, and J. Mendling, “Untrusted Business Process Monitoring and Execution Using Blockchain,” in Business Process Management, Cham, 2016, pp. 329–347. doi: 10.1007/978-3-319-45348-4_19.

[3] J. Mendling et al., “Blockchains for Business Process Management - Challenges and Opportunities,” ACM Trans. Manage. Inf. Syst., vol. 9, no. 1, p. 4:1-4:16, Feb. 2018, doi: 10.1145/3183367.

[4] C. Di Ciccio, G. Meroni, and P. Plebani, “On the adoption of blockchain for business process monitoring,” Softw Syst Model, Jan. 2022, doi: 10.1007/s10270-021-00959-x.

[5] E. Zhou, H. Sun, B. Pi, J. Sun, K. Yamashita, and Y. Nomura, “Ledger-data Refiner: A Powerful Ledger Data Query Platform for Hyperledger Fabric,” in 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS), Oct. 2019, pp. 433–440. doi: 10.1109/IOTSMS48152.2019.8939212.

[6] D. Trihinas, “Interoperable Data Extraction and Analytics Queries over Blockchains,” in Transactions on Large-Scale Data- and Knowledge-Centered Systems XLV: Special Issue on Data Management and Knowledge Extraction in Digital Ecosystems, A. Hameurlain, A. M. Tjoa, R. Chbeir, Y. Manolopoulos, H. Ishikawa, S. Ilarri, and A. Papadopoulos, Eds. Berlin, Heidelberg: Springer, 2020, pp. 1–26. doi: 10.1007/978-3-662-62308-4_1.

[7] R. Galici, L. Ordile, M. Marchesi, A. Pinna, and R. Tonelli, “Applying the ETL Process to Blockchain Data. Prospect and Findings,” Information, vol. 11, no. 4, Art. no. 4, Apr. 2020, doi: 10.3390/info11040204.

[8] Y. Li, K. Zheng, Y. Yan, Q. Liu, and X. Zhou, “EtherQL: A Query Layer for Blockchain System,” in Database Systems for Advanced Applications, Cham, 2017, pp. 556–567. doi: 10.1007/978-3-319-55699-4_34.

[9] L. Vinceslas, H. Pithadia, S. Dogan, S. Sundareshwar, and A. M. Kondoz, “Abstracting data in distributed ledger systems for higher level analytics and visualizations,” arXiv:2102.10133 [cs], Feb. 2021, Accessed: May. 14, 2022. [Online]. Available: http://arxiv.org/abs/2102.10133

[10] D. Dillenberger et al., “Blockchain Analytics and Artificial Intelligence,” IBM Journal of Research and Development, vol. PP, pp. 1–1, Feb. 2019, doi: 10.1147/JRD.2019.2900638.

[11] IBM Cloud Education. “What Is ETL (Extract, Transform, Load)?” IBM, https://www.ibm.com/cloud/learn/etl. (accessed May. 14, 2022)