DEIM Forum 2024 XX-Y

データ検証可能な分散DB実現手法の提案と実現に向けたHyperledger Irohaのコマンド開発

堀　遥† 小口　正人†

† お茶の水女子大学　〒112–8610 東京都文京区大塚 2–1–1
E-mail: †haruka-h@ogl.is.ocha.ac.jp, ††oguchi@is.ocha.ac.jp

あらまし　データ駆動型社会の実現にむけ，信頼度などに異種性を持つデータを一元的に取り扱う分散データベースシステムの開発が求められる．分散環境において，格納データの安全性を保証するためには，全てのデータが検証可能であることを保証すれば良い．すなわち，データ検証機能を有し，かつ，ユーザが安全性に欠けると判断されたデータを利用する際には，その利用可否を直接問う仕組みを実装することが要求される．我々はこのようなデータ管理基盤の開発を目指し，少ないリソースでの高速動作と高い開発の自由度が特徴のブロックチェーン基盤ソフトウェアのHyperledger Irohaをプラットフォームとした分散データベースシステムを検討した．また，このシステムの実装の一部として，検証対象となるデータの生成・格納処理を行うHyperledger Irohaの新たな組み込みコマンドの開発に着手した．

キーワード　クラウド基盤、ブロックチェーン技術、データ検証技術

1 はじめに

1.1 研究背景

近年，スマートフォンやIoTデバイスの普及から膨大なデータが収集されるようになった．DXの進展に伴うデータ駆動型社会の実現に向け，このような実社会の多様なデータをその不確実性を考慮しながら活用するような新しいデータ管理基盤の実現が期待されている．このとき，信頼度や来歴，種類などに異種性のあるデータを一元的に保存し，データ利用の際にはその信頼性の保証に貢献するような分散データベースシステムが考えられる．一方で，そのようなデータ管理基盤では，複数のユーザがアクセスできるオープンな環境において，これらのヘテロなデータの健全性・安全性の保証が課題となる．これに対し，収集・保管された全ての有効なデータが検証可能であることを保証する機能を分散データベースシステムに取り入れるというアプローチが考えられる．

1.2 既存研究

嘉戸らの研究[16]では，クラウドストレージシステムに保存されたデータの完全性検証へのアプローチとして，Private PDPを採用している．PDPはデータを複数のブロックに分割し，その中からランダムに選んだブロックに対してデータの完全性検証を行う．特にPrivate PDPでは，データ保有者とクラウドサービスプロバイダのみで完全性検証を行う方式である．Private PDPにブロックチェーン技術の一つであるスマートコントラクトを応用し，データの完全性検証および完全性検証結果不一致時の合意形成を行う方法を提案している．

また，Neha Mishra らの研究[7]では，PDV：Personal Data Vaultと呼ばれる個人の生涯にわたるデジタルドキュメントを，検証可能かつ安全な方法で保管，保存，保護，共有するためのフレームワークの開発にブロックチェーンプラットフォームHyperledgr Irohaを採用している．各文書は暗号化，圧縮され，クラウドに安全に保存され，文書の保存先URLがHyperledger Irohaに入力される．また，開発システムはMarkov Treeを用いた予測プリフェッチ機能を備えており，次に発生する要求を予測，事前実行する．

いずれの場合においても，現代で有用なデータ管理基盤において，データの安全性を保証する際にはブロックチェーンの活用することが効果的であることがわかる．

1.3 研究目的

本研究では，任意のデータが検証可能であることを保証する機能を持つ分散データベースシステムの開発することを目的とする．この目的を達成するため，まずメタデータをブロックチェーンプラットフォームHyperledger Irohaに付属するオンチェーンデータベースにて管理する．オフチェーンデータベースへのデータの追加や更新などが発生し，新規メタデータが必要となる場合，一連の処理をHyperledger Irohaの組み込みコマンドで行う．コマンドの結果はブロックチェーンに記録され，これによりメタデータの安全性が確保される．そして，安全性の保証されたメタデータ，蓄積されたブロック，Hyperledger Irohaのブロック検証機能を利用し，データの検証可能性を保証する．最終的には，データ検証の結果，安全性が保証されないと判断されたデータを利用する際には，ユーザに対象のデータの利用可否を問うシステムの実現を目指す．

1.4 実証実験

脱炭素を実現する循環型社会への貢献を期待し，製造業におけるカーボンフットプリント管理を対象とした実証実験を行う．

表 1　ブロックチェーンの適用事例

業界	企業名	利用目的
金融	三菱 UFJ ファイナンシャルグループ	独自のデジタル通貨管理 [14]
不動産業	積水ハウス株式会社	物件情報と不動産賃貸契約の実行 [15]
インフラ (電力取引)	国立大学法人東京大学 トヨタ自動車株式会社 TRENDE 株式会社	分散型電源を活用した電力の個人間売買システムを検証 [11]
医療	Z.com Cloud ブロックチェーン	医療カルテの共有 [5]
コンテンツ業	ソニー株式会社 株式会社ソニー・ミュージックエンタテイメント 株式会社ソニー・グローバルエデュケーション	デジタルコンテンツに関わる権利情報を処理する機能 [12]

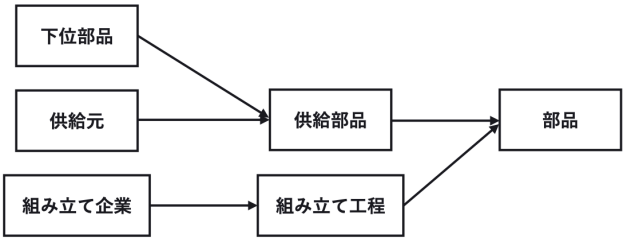

図 1　カーボンフットプリント応用のスキーマ

複数企業を接続する分散データベース上で製造工程全体のトレーサビリティを実現し，センサなどのクライアントデバイスから収集した CO_2 排出量データのメタデータを Hyperledger Iroha で管理する．これにより，収集した CO_2 排出量の検証可能性が保証され，カーボンフットプリント削減に向けた安全・安心なデータ分析・予測を支援する．

本稿では，自動車製造に関わる企業のうち部品製造・組み立て工場にフォーカスし，これらの工場間における製造時・組み立て時のカーボンフットプリント管理を想定する．図 1 は，カーボンフットプリント応用におけるスキーマであり，供給元から下位部品が供給され，組み立て工場による組み立て工程を経て，部品を構築する．本スキーマは各工場が管理しており，下位部品から上位部品を階層的に組み立てて，複数企業を横断して最上位部品の自動車を製造する．

一台の自動車には約 3 万点ほどの部品が使用されるとされている [9]．おおよその国内の自動車部品メーカーと呼ばれる企業数，過去 10 年間で発売された新車数，そしてこれらの重複などを考慮すると，1 車種あたり 100 社程度が扱う 3 万点もの部品を分散管理できるシステムが必要となることが考えられる．

ここで，スキーマに従い，部品の組み立て工場で排出された CO_2 排出量を *EMISSIONS* と呼ぶことにする．また，各部品は階層関係にあり，CO_2 排出量もまた階層関係にある．下位部品の *EMISSIONS* を再帰的に足し合わせ，部品自体の *EMISSIONS* と合計したものを *TotalEMISSIONS* と呼ぶ．本稿では各部品の *TotalEMISSIONS* を検証対象データであることを前提にシステムの実装手法を行う．

1.5　本稿の構成

本稿は以下の通り構成される．第 2 章では本研究の前提となるブロックチェーン及び Hyperledger Iroha の概要を説明する．第 3 章では本研究の提案手法の概要を，第 4 章で提案手法の実装に向けた計画を説明する．最後に第 5 章でまとめ，今後の課題を述べる．

2　関 連 研 究

2.1　ブロックチェーン

ブロックチェーンは，2008 年に Satoshi Nakamoto により投稿された論文 [8] に基づき，暗号通貨 Bitcoin [2] の公開取引台帳としての役割を果たすために発明された．Bitcoin の成功によりブロックチェーンは仮想通貨技術として強く注目されたが，一般社団法人日本ブロックチェーン協会は広義のブロックチェーンを「電子署名とハッシュポインタを使用し改竄検出が容易なデータ構造を持ち，且つ，当該データをネットワーク上に分散する多数のノードに保持させることで，高可用性及びデータ同一性等を実現する技術」と定義している [1]．

現在では仮想通貨以外での用途も促進されており，表 1 は書籍 [13] で紹介される例を一部抜粋したものである．例えば不動産業では，煩雑で膨大な量の手続きによって浪費されるコストと時間を，ブロックチェーン活用によって削減し，より効率的にさせている．また，医療では，ブロックチェーンに備わる権限機構と威厳性を活用し，医療機関ごとに管理されているカルテをブロックチェーンで一元的に管理して，より安全に利便性を高めるといった効果をもたらしている．

2.1.1　技術的概要

ブロックチェーンは，ブロックと呼ばれるトランザクションを記録する単位を生成し，これをチェーンのように連鎖するデータ構造である．仮想通貨の基礎技術として開発されたブロックチェーンの厳格性を担保する仕組みはチェーンにある．一つのブロックにはハッシュ値が付与される．このハッシュ値は，前のブロックのハッシュ値と新規のトランザクションの内容やタイムスタンプなどから算出される．すなわち，ハッシュ値によりブロックの関連が生まれ，これがチェーンとなる．よってブロックチェーンは高い改ざん耐性を持つと言われる．また，Peer to Peer 型のブロックチェーンネットワーク上に参加する各ノードがブロックチェーンを管理するるため，欠損ブロックを他のノードから補うことができ，耐障害性と可用性に優れるといった特徴を持つ．

表 2　ブロックチェーンと集中型データベースの比較

比較項目	ブロックチェーン	中央型データベース
A.　構築の信頼性	管理者が必要ない場合がある	中央管理者が必要
B.　データの機密性	全ノードがデータ参照可能	関係者のみにアクセスを制限
C.　堅牢性& 耐障害性	データはノード間で分散される	データは中央データベースで保管
D.　パフォーマンス	合意形成に時間を要す (Bitcoin では約 10 分)	即時に実行/アップデート
E.　冗長性	(デフォルトでは) 参加する各ノードが最新のコピーを保持	管理者のみがコピーを保持
F.　セキュリティ	(デフォルトでは) 暗号化メカニズム使用	従来のアクセスコントロール

2.1.2　従来のデータベース技術との比較

M. J. M. Chowdhury らの論文 [3] では，ブロックチェーンと集中型データベースの各アプローチを表 2 のように挙げている．また，これら 6 つの比較項目に対しする比較分析から，データの信頼性，堅牢性，実証性がシステムの優先事項であれば，ブロックチェーンがより優れており，機密性とパフォーマンスが優先であれば，従来のデータベースがより優れたソリューションであるとされている．

以上より，ブロックチェーンとデータベースの両方の機能を持つことで，アプリケーションは効率性と安全性を高めることができるため，ブロックチェーンプラットフォームの多くはデータベースと統合されている．

2.1.3　ブロックチェーンのモデル

ブロックチェーンは「パーミッションレス型」,「パーミッション型」,「コンソーシアム型」という 3 種類のモデルに分別される．表 3 はそれぞれのモデルの特徴である．

Bitcoin や Etherum [4] に代表されるパーミッションレス型は非中央集権でマイニングと呼ばれる膨大な計算の承認によって取引の正当性を担保する．透明性の高さを持つ反面，時間とリソースのコストが高く，データのプライバシーは保証されない．

パーミッション型は，中央集権型のネットワークで，透明性はないもののプライバシーが確保される上に，マイニングも行う必要がないため，高いパフォーマンスを持つ．こうした特徴から，プライベート型は単一の企業や組織内での運用に有効であり，特に銀行間の取引や証券取引での活用が促進されている．次の節で説明する Hyperledger の数々のプロジェクトはパーミッション型である．

最後のコンソーシアム型には複数の管理者が存在している．そのため，パーミッションレス型の分散性とパーミッション型の迅速な大量処理が可能という機能を備えている．また，管理者が複数存在しているため，運用ルールの変更についても一定数以上の合意が必要となる．これにより，高い公平性が保たれ，セキュリティや耐障害性もプライベート型に比べると強固であるとされる．こうした特徴から，コンソーシアム型は同業他社が協力して構築するブロックチェーンシステムに有効であり，こちらも銀行間の取引や証券取引での活用が促進されている．

2.1.4　コンセンサスアルゴリズム

ブロックチェーンを構成する技術の中に「コンセンサスアルゴリズム」がある．コンセンサスアルゴリズムとは，生成ブロックをブロックチェーンに追加する前に各ノード間で合意形成を行うメカニズムである．表 4 は代表的なコンセンサスアルゴリズムである．本研究で使用する Hyperledger Iroha では効率的に少ない Peer 数で承認を得る YAC 方式をとる．

表 3　ブロックチェーンの種類

名称	管理者	参加者
パーミッションレス型	なし	制限なし
パーミッション型	複数存在	個人や単一の組織
コンソーシアム型	複数存在	グループや複数の組織

表 4　ブロックチェーンの代表的なコンセンサスアルゴリズム

名称	採用システム	決定方法
PoW	Bitcoin Ethereum	正解を最も早く算出した Peer を採用
PoS	Ethereum で検討	資産の保有量に応じて 有利な条件で PoW を実施
PBFT	Hyperledger Fabric	検証を行った Peer の 過半数から承認を得る
YAC	Hyperledger Iroha	PBFT より効率的に少ない Peer 数で承認を得る

2.2　Hyperledger Iroha

Hyperledger [6] は 2015 年 12 月に The Linux Foundation よって開始され，ブロックチェーンベースの分散台帳をサポートするプロジェクトである．仮想通貨や金融，サプライチェーンに限らず，広範囲なビジネスシーンへのブロックチェーン基盤提供し，パフォーマンスや信頼性などの多方面においてサポート・改善ことを目指す．

Hyperledger では方向性の異なる複数のプロジェクトが推進されており，GA リリースされたプロジェクトは 2024 年 1 月時点で 4 つとなる．Hyperledger Iroha は 2019 年 5 月に GA リリースされたパーミッション型ブロックチェーンプラットフォームである．ソラミツ株式会社が初期開発者として継続的に開発に貢献している．

2.2.1　特　　徴

Hyperledger Iroha の主な特徴として「簡単な導入とメンテナンス」「開発に向けたさまざまなライブラリ」「役割に基づいたアクセス制限」「コマンドとクエリの分離によって行われるモジュール型設計」「資産とアイデンティティ管理」の 5 点が挙げられる．

- 簡単な導入とメンテナンス

Hyperledger Iroha の導入は非常に容易である．Docker 環境を利用したテスト環境であれば数十分で終了する．また，全ての Hyperledger プロジェクトはオープンソースとしてソースコー

ドが公開されている．

- 開発に向けたさまざまなライブラリ

Hyperledger Iroha は Java，JavaScript，Swift，Python の 4 つのプログラミング言語に対する API を備える．API は統一されておりどのプログラミング言語からでも同様の操作を実現できる．そのため，開発者のニーズに合わせた言語選択が可能となる．

- 役割に基づいたアクセス制限

仮想通貨では権限の概念が軽薄であり，これはネットワークにアクセス可能なユーザに対して全ての資源に対するアクセス権を付与する状況にある．Hyperledger Iroha では，ドメインと呼ばれる権限が及ぶ範囲を表す概念，ロールと呼ばれる複数の権限を 1 つに集約する概念が実装されている．これにより，広範囲な用途に対して，汎用的かつ柔軟的に対応することが可能である．

- コマンドとクエリの分離によって行われるモジュール型設計

Hyperledger Iroha の API 操作は，Hyperledger Iroha に対し変化をもたらすコマンドと，Hyperledger Iroha は普遍のまま現在持つ情報を表示するクエリの 2 種類に分けられる．コマンドの実行結果は，トランザクションとしてブロックチェーンに記録される．これらの 2 種類の API を組み合わせ，システムの開発が可能である．

- 資産とアイデンティティ管理

仮想通貨では通常，単一の仮想通貨を扱うが，Hyperledger Iroha では複数の通貨をアセットという概念で実装されている．アセットは作成時にドメインを指定する必要がある．

また，品質モデルにおいては次の 3 点が挙げられる．

- 信頼性

耐障害性，回復性．

- パフォーマンス効率

とりわけ時間挙動とリソースの使用効率．鍵生成アルゴリズムに安全性とパフォーマンスに優れた ED25519 を，コンセンサスアルゴリズムにより高速な YAC 方式を採用している点も効果的となっている．

- ユーザビリティ

学習可能性，ユーザエラー保護，妥当性の評価可能性．

以上の特徴を踏まえ，Hyperledger Iroha は Hyperledger プロジェクトの中でも，最も容易で短時間にブロックチェーンによる分散台帳を実現することができ，かつライトウェイトであるとされている．また，豊富な API や柔軟な概念構成から高い自由度の開発環境が提供されている．

2.2.2 ブロック検証機能

Hyperledger Iroha は起動時に全ブロックのハッシュ値の再計算を行い，ブロックを検証する．もし改ざんが発生した場合どのような挙動を示すか，ブロック欠損時，ブロックチェーン改ざん時について説明する．

まず過去ブロックの欠損時，起動時のハッシュ値再計算にてブロックの欠損を知らせるエラーメッセージが出力され，Hyperledger Iroha は起動しなくなる．複数 Peer 構成で運用する場合は，自動的に他の Peer から欠損ブロックが読み込まれる．

次にブロック情報の改ざん時，ブロック中のトランザクション処理内容であれば，ハッシュ値に変化が生じる．そのため，他の Peer とのハッシュ値の比較により改ざんが検知される．そしてブロックの作成者，すなわち電子署名が改ざんされれば，ブロック欠損時同様に Hyperledger Iroha は起動せず，改ざんが検知される．

本研究では，Hyperledger Iroha のブロック検証機能を利用し，データ検証メカニズムを実現することを検討している．

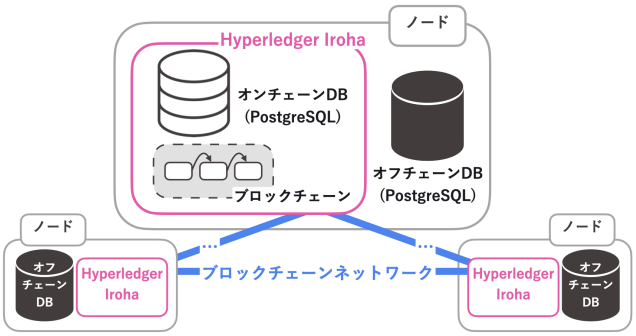

図 2　システム全体の構成

3 提案手法

本章では，提案手法を提案モデル，提案手法の手順，カーボンフットプリント応用における動作シナリオに分けて詳しく説明する．

3.1 提案モデル

まず本稿における提案モデルのシステム全体の構成を図 2 に示す．各ノードは Hyperledger Iroha とオフチェーンデータベースを備える．カーボンフットプリント応用の場合，ノードは企業と捉えてよく，全てのノードが Hyperledger Iroha によって構成されるブロックチェーンネットワークで接続される．オフチェーンデータベースと Hyperledger Iroha のオンチェーンデータベースはともに，オープンソースのリレーショナルデータベース管理システムである PostgreSQL [10] にて実装する．次の項で，各データベースの詳細を述べる．

3.1.1 オフチェーンデータベース

オフチェーンデータベースはノードに固有のデータベースである．主に各ノードで製造に関わっている部品のデータを管理する役割を持ち，検証対象データはオフチェーンデータベースに保管される．

表 5 は，オフチェーンデータベース上で管理するテーブルの例である．テーブル CO2Emissions は，カーボンフットプリント応用において現在想定している，検証対象データ *TotalEMISSIONS* を管理するテーブルであり，PartsID は部品の ID，部品の *EMISSIONS*，*TotalEMISSIONS*，時刻を表す TimeStamp を属性としてもつ．プライマリーキーとして PartsID と TimeStamp を指定している．ノードで新規 *EMISSIONS* が得られると，Hyperledger Iroha 上で *TotalEMISSIONS* が算出され，

表 5　テーブル CO2Emissions

CO2Emissions	
PK	PartsID
	EMISSIONS
	TotalEMISSIONS
PK	TimeStamp

表 6　テーブル Metadata

Metadata	
PK	PartsID
	Link
	ChildPartsID
	TimeStamp

これらの値がタイムスタンプとともにテーブル CO2Emissions に追加される．

上記の例以外にも，部品の基本情報や，ノードに関連する情報などもここで管理される．

3.1.2　オンチェーンデータベース

Hyperledger Iroha のオンチェーンデータベースは World State View とも呼ばれ，Hyperledger Iroha の最新情報が格納される．すなわち，ブロックチェーンネットワークに参加する全ノードでこの内容が同一となる．ここで，ブロックチェーン上のブロックそのものはオンチェーンデータベースには保存されず，json ファイルで生成されて別の場所で保管されている．

提案システムでは，各ノードのオフチェーンデータベースにて保管される全ての検証対象データに対し，メタデータを作成する．全部品のメタデータは，オンチェーンデータベース上に存在するテーブル Metadata に格納・管理される．表 6 は，カーボンフットプリント応用におけるテーブル Metadata である．PartsID は部品の ID，Link は表 5 上に格納される *TotalEMISSIONS* のリンク情報，ChildPartsID は下位部品の PartsID，TimeStamp は時刻を表す．プライマリーキーとして PartsID のみを指定しているため，新規メタデータ作成のたびにテーブル Metadata にデータを追加するのではなく，対象部品のカラムを更新する．1.4 節でも触れたように，開発システムで扱う部品数は約 3 万点と推測されるため，テーブル Metadata では全部品，おおよそ 3 万もの最新のメタデータが保管されることとなる．

このように，実データでなくメタデータをブロックチェーンで管理することで，実データのデータ形式に捉われず，また，その設計次第でデータのサイズを抑えることが可能となる．

3.2　提案手法の手順

本説では，提案手法の検証対象データの格納時，データの参照時の手順を説明する．その後，カーボンフットプリント応用における動作シナリオを説明する．

3.2.1　検証対象データの格納時

まず，検証対象データ格納時の手順を説明する．

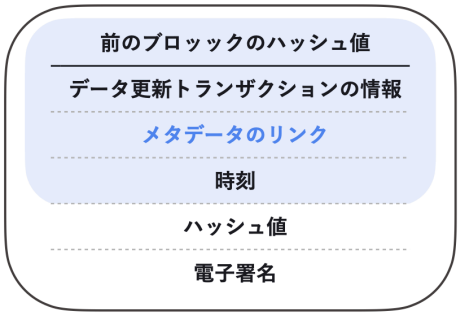

図 3　ブロック構成の概要

(1)　データ更新トランザクションがトリガーされる．
　　トリガーの条件は応用システムにより様々である．

(2)　データ更新トランザクションの実行．

　(2-1)　必要に応じたデータの加工処理などの実施．
　(2-2)　オフチェーンデータベースに格納．

(3)　(2-2) の格納場所情報 (Link) を含めて新規メタデータ作成．

(4)　(3) のメタデータをテーブル Metadata に格納．

(5)　ブロック作成，追加．

データ更新トランザクションでは，まず検証対象データ追加・更新のために必要な一連のデータ加工処理や演算を行う．これにより新しく得た検証対象データをオフチェーンデータベースに格納する．データ更新トランザクションの実装は，Hyperledger Iroha の組み込みコマンドとして開発を行う．

図 3 は (5) で作成，追加されるブロックの構成概要である．メタデータのリンクとは，テーブル Metadata のインデックスやプライマリーキーに該当するデータを表す．前のブロックのハッシュ値，更新トランザクションの情報，メタデータのリンク，時刻情報から，ブロックのハッシュ値は生成される．

以上の手順で検証対象データを格納することにより，メタデータの厳格性が担保される．

3.2.2　データ参照

データ参照時の手順は次の通りである．

(1)　メタデータを参照し，Link を取得．

(2)　(1) からオフチェーンデータベースにアクセスし，データを参照．

3.2.3　実証実験におけるデータ更新トランザクション

カーボンフットプリント応用では，*TotalEMISSIONS* を検証対象データとする．*TotalEMISSIONS* は，下位部品の *TotalEMISSIONS* の合計値と *EMISSIONS* を加算することで求まる．

本稿では，データ更新トランザクションのトリガー条件を新

規 *EMISSIONS* の取得とする．*EMISSIONS* の新しい値が得られた部品の *TotalEMISSIONS* を更新するため，データ更新トランザクションが実行される．データ更新トランザクションの手順は次の通りである．

(1) テーブル Metadata を参照し，対象部品の下位部品の PartsID を取得.

(2) テーブル Metadata で (1) の PartsID を検索し，Link を取得.

(3) Link からオフチェーンデータベース上の下位部品の *TotalEMISSIONS* を取得.
これを対象部品の全下位部品で行う.

(4) (3) で取得した *TotalEMISSIONS* らを合算する.

(5) (4) で算出した値に対象部品の新規 *EMISSIONS* を加算し，これが新規 *TotalEMISSIONS* となる.

(6) (5) をオフチェーンデータベースに格納.

データ更新トランザクションとその後のメタデータ生成・更新の一連の操作は，Hyperledger Iroha のコマンドにより実装する．また，ブロックに含まれるメタデータのリンクは常に対象部品の PartsID となる．

4 提案手法の実装に向けた計画

提案手法実装のため，第一にデータ検証トランザクションを実装する．具体的には，データ更新トランザクションの処理を提供する Hyperledger Iroha の組み込みコマンドの作成である．そのために Hyperledger Iroha の構成ファイルの開発が必要となる．

データ更新トランザクションを実現に向け，そのベースとなるコマンド *Update_testTable* の開発を行なう．開発環境，コマンド Insert の概要の二点について以降の節で説明する．

4.1 開 発 環 境

表 7 は実装環境の詳細である．仮想環境として Docker を使用し，Ubuntu22.04LST コンテナ上に Hyperledger Iroha の動作環境をビルドする．PostgreSQL コンテナでは，オンチェーン DB とオンチェーン DB としての機能を提供する．表 6 に示したテーブル Metadata は PostgreSQL コンテナ上で動作するデータベース上に存在する．

4.2 コマンド *Update_testTable* の概要

コマンド *Update_testTable* は，Hyperledger Iroha 上のオンチェーンデータベースに保管されるテーブル testTable に対して，UPDATE 命令を実行するコマンドである．表 8 にテーブル testTable の詳細を示す.

コマンド *Update_testTable* では，属性 Val に格納される値の更新を行う．実行の際には，プライマリーキーの ID と Val の更新値を与え，これらを持ってテーブル testTable に対して，UPDATE 命令を実行する.

表 7 実 装 環 境

コンテナ	Hyperledger Iroha ver.1	オン/オフチェーン DB
	Ubuntu イメージ：Ubuntu:22.04	PostgreSQL イメージ：postgres
仮想環境	Docker	
OS	Ubuntu20.04LTS	
サーバ	CPU：Intel(R) Xeon(R) Silver 4314 CPU @ 2.40GHz メモリ：192GB	

表 8 テーブル testTable

testTable	
PK	ID
	Val
	TimeStamp

コマンド *Update_testTable* 開発おける目標として，以下の 3 点が挙げられる.

- コマンド実行結果がテーブル testTable に反映されている.
- コンセンサスアルゴリズムを経てブロック作成がアクセプトされている.
- 作成されたブロックにコマンド実行情報が記録されている.

5 まとめと今後の課題

本稿では，信頼度などに異種性を持つヘテロなデータを一元的に保管・利用するための分散データベースシステムの実現を目標とし，提案システムの検討を行った．このような，分散環境においてデータの安全性を保証するため，本稿では任意のデータに対するデータ検証機能をシステムに組み込むというアプローチを採用した．データ検証機能実現に向け，システムのプラットフォームにはブロックチェーン基盤ソフトウェアの Hyperledger Iroha を用い，Hyperledger Iroha にて検証対象データのメタデータを管理するためのメタデータ設計も行った．また，実装の一部として，Hyperledger Iroha の組み込みコマンドの開発に着手した.

今後は，第一にデータ更新トランザクション実装に向けた，コマンド *Update_testTable* の開発を行う．コマンド *Update_testTable* はオンチェーンデータベース内に閉じた処理であるため，これの開発が完了次第，Hyperledger Iroha からオフチェーンデータベースへのアクセス手法の検討・および実装に取り掛かる．その後は，提案手法におけるブロック構成と検証対象データ格納シナリオを順に実装する予定である.

また，Hyperledger Iroha のブロック検証機能の有用性についても調査を行なっていく.

謝　　辞

本研究は一部，JST CREST JPMJCR22M2 の支援を受けたものである．

文　　献

[1] Japan Blockchain Association. ブロックチェーンの定義. https://jba-web.jp/news/642.
[2] Bitcoin. https://bitcoin.org/ja/.
[3] Mohammad Jabed Morshed Chowdhury, Alan Colman, Muhammad Ashad Kabir, Jun Han, and Paul Sarda. Blockchain versus database: A critical analysis. In *2018 IEEE TrustCom/BigDataSE*, pp. 1348–1353, 2018.
[4] Ethereum. https://ethereum.org/ja/.
[5] GMO インターネット株式会社. 医療機関カルテ共有システム. https://www.gmo.jp/news/article/5736/.
[6] Hyperledger foundation. https://www.hyperledger.org/.
[7] Neha Mishra and Dr. Haim Levkowitz. Pdv: Permissioned blockchain based personal data vault using predictive prefetching. In *BIOTC '21: Proceedings of the 2021 3rd Blockchain and Internet of Things Conference*, pp. 59–69.
[8] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.
[9] Toyota クルマ教室. https://global.toyota/jp/kids/faq/parts/001.html.
[10] Postgresql: The world's most advanced open source relational database. https://www.postgresql.org/.
[11] TOYOTA 自動車株式会社. 東京大学、トヨタ、trende が、次世代電力システムの共同実証実験を開始. https://global.toyota/jp/newsroom/corporate/28227543.html.
[12] ソニー株式会社, 株式会社ソニー・ミュージックエンタテインメント, 株式会社ソニー・グローバルエデュケーション. ブロックチェーン基盤を活用したデジタルコンテンツの権利情報処理システムを開発. https://www.sony.com/ja/SonyInfo/News/Press/201810/18-1015/.
[13] 佐藤栄一. Hyperledger Iroha 入門 -ブロックチェーンの導入と運用管理-. 株式会社 オーム社, 2020.
[14] 三菱 UFJ フィナンシャル・グループ. Mufg におけるブロックチェーンの取組み. https://www.boj.or.jp/paym/fintech/data/rel180214a6.pdf.
[15] 積水ハウス株式会社. 積水ハウス 業界の新たなスタンダード構築へ 業界初 ブロックチェーンで賃貸入居の煩雑なプロセスをワンストップ化. https://www.sekisuihouse.co.jp/library/company/topics/datail/__icsFiles/afieldfile/2020/06/08/20200608.pdf.
[16] 嘉戸裕一, 廣友雅徳, 瀧田愼, 掛井将平, 白石善明, 毛利公美, 森井昌克. ブロックチェーンを用いたクラウドストレージのプライベートな完全性検証方式. In *Computer Security Symposium 2023*, pp. 727–734, 2023.