

Hyperledger Fabric を用いたブロックチェーン型 簿記・会計システムへの実践実験

板橋 雄大

1. はじめに

Deloitte (2021) では、2021年3月24日から4月10日にかけて10か所（ブラジル、中国本土、ドイツ、香港特別行政区、日本、シンガポール、南アフリカ、アラブ首長国連邦、英国、および米国）の1280人の上級管理職（回答者の内CEO、CIO、CTOは20%、VPおよび事業部門長が44%含まれている）および実務家に対して行った調査結果が公表された。サンプルとなった回答者それぞれの具体的な地位およびブロックチェーン（Blockchain: BC）、暗号通貨、およびデジタル資産に対する知識レベルは明らかになっていないが、それらに対する一般的な理解は持っていることが明らかにされている。

この調査の結果、「BC技術は幅広く拡張可能であり、主流になりつつある」という設問に対して、強く同意するかあるいはある程度同意するという回答が、全回答者の81%であった（強く同意した割合は39%）（Deloitte (2021), p.4）。この割合は、2020年の同調査（Deloitte (2020), p.30）の88%、2019年の同調査（Deloitte, 2019, p.5）の86%よりも低くなっているものの、依然としてBC技術への期待が高いことが伺える。一方で、2021年の調査では金融サービス業の回答者サンプルの中で特に「回答者の組織が、すでにBCソリューションを本格的に稼働し、かつ/またはデジタル資産を中核的な事業活動に統合している」という回答者を「パイオニア」と呼称しているが、このパイオニアの割合は全回答者の5.5%（70人）に過ぎず、現状ではまだBCの活用については先進的な一部の企業に留まり、期待と現実の乖離が広がりつつある様子がうかがえる（Deloitte, 2021, p.4）。

Deloitte (2021, p.18) では、BCの利用状況の調査結果も掲載されているが、安全な情報交換（45%）、デジタル通貨（44%）、資産追跡・管理（40%）、デジタルID（40%）、規制遵守（36%）、金融フローのトレーサビリティ（34%）、文書・記録の電子化（33%）、識別情報（32%）、資産のトークン化（31%）、製品サプライチェーンの追跡（31%）、支払い・取引プロセス（31%）、契約管理（31%）、内部統制（31%）、模倣品対策（28%）、という結果であった。

これらのアンケート結果からは、BCへの期待の大きさに比して、実際の場面における利用範囲および利用企業数は限定されている現状であることがわかる。

次に、普及に対する障害として挙げられているのは、サイバーセキュリティ（71%）、規制の壁（63%）、金融インフラ（62%）、プライバシー（59%）、古いプロセスとシステム（50%）、セカンダリーマーケット（47%）、広範な（BC技術の）採用（47%）、価値提案の欠如（43%）、人材へのアクセス（41%）となっている（Deloitte, 2021, p. 9）。

以上のアンケートは、BC技術についての内容であるが、BC技術を用いる簿記会計システム（BC型簿記会計システム）の開発においても同様の問題が発生すると考えられる。特に、BC型簿記会計システムが既存のシステムの再現に留まり、BC技術の特性を活かした新しい価値提案を行うことが出来なければ、企業にとってBC型簿記会計システムを新規に導入するインセンティブは低くなる。

そのため本稿では、実際にBC型簿記会計システムを構築し、実践¹⁾ 実験を行うことで、既存の簿記及び会計システムとBC技術をどのように融合させるのか、どのようにプライバシーを守りながら同時に取引相手の情報を確認するのか、多くの利害関係者にどのようにネットワークへの参加を可能にするのかを検討し、新しい価値提案を行いたいと考える。

そのために、本研究において構築したBC型簿記会計システムでは、

- 1) 複数の勘定科目のカテゴリーを用いてより複雑な取引を記録すること、
- 2) 記録された取引データを、計算し、集計し、財務情報の形で自動出力することを試みている。

複雑な仕訳をBCで直接記録し、そこから利用者の目的に応じた出力内容を自動的に作成することができれば、BC型簿記会計システムは、よりタイムリーで検証可能な情報をステークホルダーに提供する手段として新しい価値を提案できると考えられる。また、「情報の集約という行為は、一般にインプットされたデータがどのようになっているかに大きく依存する」（坂上、2016）ため、2) においてBC上を流れる「大量の生データを誰でも簡単に集約できる仕組み」（同上）を検討することは、1) の改善にもつながると考えられる。

特に、会計分野へのブロックチェーンへの適用に関しては、「導入事例が存在しない」（Pascual Pedreño E. et al. (2021)）ことや、分散型台帳技術を使って財務会計を管理している企業はほとんど知られておらず、多くは、暗号通貨やトークンの取引のみを記録している事業者である現状を踏まえれば、本研究の意義は大きいと考える。

2. BCと分散型台帳技術

Harish Natarajan et al. (2017, p. 2) において言及されているように、一般に、BCと分散型台帳技術（Distributed Ledger Technology: DLT）については、その用語定義自体に混乱が見られる。本稿ではHarish Natarajan et al. (2017) の定義を簡略化して、「分散型台帳とは、コンピュータサーバーの分散ネットワークを利用して、複数のデータストア（または

台帳)間でデータを記録・共有するための技術」と定義する。BCは、「DLTの一部で使用されているデータ構造で、データをブロックと呼ばれる構造に暗号化して格納、分散ネットワーク内に転送する。参加者(ノード)による合意形成を経て承認されたブロックは、時系列連続性を持った形で既存のブロックと連結されネットワーク内において同期される」²⁾ものとする。

DLTは、ノードが台帳を変更する際に、いずれかの主体からの許可が必要かどうかで、許可が必要なものを Permissioned 型、許可が不要なものを Permissionless 型に分類される。Permissioned 型は、管理者に許可された単一あるいは複数の限定的なノードが取引の検証・承認を担う行方形態である。Permissionless 型の代表例としてはビットコインが挙げられ、不特定多数の参加ノードが取引の検証・承認を担う形態である。

また台帳が誰でもアクセス可能か、ネットワークの参加ノードのみがアクセス可能かによって、パブリック型とプライベート型に分類される。パブリック型が誰でも許可なくネットワークに参加でき、管理者が存在しないのに対して、プライベート型は特定の管理者が存在し、管理者によって許可された者のみがネットワークに参加できるという違いがある。パブリック型の利点としては、ネットワーク内で情報が完全に共有されることで、いわゆる透明性が高いことと、中央集権的な管理者が存在しないためネットワークに発生する障害に対して耐性がある(システムを構成するハードウェア群において、そこが故障することでシステム全体が停止するような単一障害点が存在しない)ことが挙げられる。対して、プライベート型の利点は、取引データが外部に公開されないためプライバシーが確保されることと、ノードの数が少なく、かつ各ノードの参加を管理者が承認しており信頼性の低いノードの存在を想定しなくて済むことから、取引の承認のための合意形成にかかる時間が短く大量の取引データの処理が迅速に行える点が挙げられる。

こうした特徴から一般的に、ビジネス用途に向いているのはプライベート型であるとされる。なお、プライベート型の管理者が複数いる場合を特にコンソーシアム型と呼び、合意形成は一般的には複数の管理者による多数決によって行われる。

上述のように、DLTは分散性と(不完全ではあるが)即時性を持っている。分散性の結果として、決済仲介に第三者を介在させる必要がなくなり、取引決済の迅速化、取引コストの削減が実現される。また、即時性の結果として、情報共有が設定されたノード間では、よりリアルタイムで継続性のある情報共有が実現される。

BCの特徴の一つとして、ブロックに格納する際にデータを暗号化する結果、ハッキングやデータ漏洩のリスクを低減することが出来ることがある。また、ブロックは合意形成を経て承認されなければネットワーク内の他のブロックに連結されることはないため、連結されたブロックに格納されているデータには一定の信頼性が担保される。そして、連結されたブロック間に時系列連続性が存在する結果として、取引履歴の追跡可能性・検証可能性が実現

される。また、時系列連続性を持つ BC において、特定のブロックに含まれる情報を改竄するためには、そのブロックの情報を引き継ぐ以降のブロックすべてを、ネットワークの他のノードよりもすばやく全て改竄する必要があるため改竄耐性に優れているとされる。

その他に、特徴として挙げられることがある透明性については、台帳が誰でもアクセス可能かどうか（つまり、パブリック型かプライベート型か）という観点と、ネットワーク参加ノード内において一部の取引が取引当事者以外に確認できないか状態を提供できるか否かという観点があり、パブリック型のみが2つの意味での透明性を実現している。

3. DLT/BC プラットフォーム

さて、以上のように DLT 及び BT にはいくつかのバリエーションが存在するが、現時点において企業が求める DLT は、一般に Permissioned 型でプライベート型であるとされる。BC アナリティクス企業の Blockdata 社の調査結果（Blockdata, 2021, p.15）によれば、米国上場企業トップ100社を調査した結果、各社が採用しているプラットフォームは、表1のようになっている。³⁾

表1 プラットフォームの種類と採用企業数

プラットフォーム名	採用企業数
Hyperledger Fabric	26
Ethereum	17
Quorum ⁴⁾	11
Concord	8
Hyperledger Sawtooth	3
AxCore	3
Hyperledger ⁵⁾	3
Bitcoin, GumboNet, Ant Blockchain, Mediledger, Multichain	各2社
Theta Blockchain, Diem, FISCO BCOS, OpenSC, Dragonchain, Baseline (check), Mad-Network, DAML (SC language), Zilliqa, Flow, Fiducia, Energy Web Chain, Ripple, Cardano, FiMAX S3C, Aura, Medilegder, Mastercard Blockchain, private blockchain,	各1社
非開示	2
不明	15
採用していない	26

(Blockdata.tech 公表データより筆者作成)

なお、企業によっては複数のプラットフォームを採用している場合もある。多くのプラットフォームを採用している順に述べると、SAPが最多で5プラットフォームを採用していた。Microsoft CorporationとAccenture Plcは4プラットフォーム採用、以下、3プラットフォーム採用が8社、2プラットフォーム採用が17社、1プラットフォーム採用は29社であった。

4. 主要3プラットフォームの特徴

「はじめに」で述べたように、BC技術の普及に対する障害のひとつとして、広範な採用が実現していないことが挙げられる。異なるBCプラットフォーム間でのデータ接続に各プラットフォームが取り組んでいるとはいえ、採用企業数の少ないプラットフォームを敢えて

表2 3つのプラットフォームの特徴

	Hyperledger Fabric	Ethereum	Quorum
ネットワークへの参加許可	Permissioned	Permissionless	Permissioned
管理者の有無	管理者有	管理者無	管理者無
取引の追跡可能性	取引データを共有したいノードを含むチャンネルを作成する形式。そのためチャンネル内のノードは全ての情報を参照可能。	ネットワーク内の全ノードはブロックデータを共有しているため、追跡が可能。	暗号化によって取引の内容の秘匿化のみ可能。PMTs (privacy marker transactions) を設定する場合、取引の存在自体を非公開にできる。
取引主体の開示	一般的にはチャンネル内のノードは全トランザクションを共有し、取引主体を確認できるが、private data 設定により一部のノードのみが確認できるようにも設定可能。	トランザクションに含まれる情報に関しては匿名性無し。	取引に係る主体の電子的地址は共有される。PMTsを設定し取引データをBCから外す場合、取引の存在自体を非公開にできる。
合意形成	RAFT ⁶⁾ /KAFKA (非推奨)	POW (PoSも試験導入) ⁷⁾	Clique POA ⁸⁾ (非推奨) /RAFT (非推奨)/BFT ⁹⁾
スマートコントラクト言語	NodeJS, Golang, Java	Solidity	Solidity
独自仮想通貨	無し (Tokenの作成可能)	Ether (ETH) (Tokenの作成可能)	ETHを前提としている (Tokenの作成可能)
処理スピード	速い	遅い	中間

採用するのは、特別な意図がある場合を除いて望ましくない。そこで、Blockdata (2021, p.15) を参考に、現在主流となっているプラットフォーム3種類について、その特徴を確認し、我々の実践実験の対象とするプラットフォーム選定の参考とする。

各プラットフォームには表2のような特徴がある。現時点において多くの企業がBCに対して持つ懸念を考えれば、取引情報の追跡可能性、取引主体の情報がそれを必要とするノードのみに提供される Hyperledger Fabric (以下、HF) と Quorum は Ethereum よりも企業活動に適していると言える。

また、HF はネットワークへのノードの参加自体を管理者が管理できるが、Quorum は Ethereum をベースとした企業向けプラットフォームであるためノードの参加を管理するのではなく、取引の際にプライバシー設定を行うことで、取引情報や取引主体の情報の匿名性を確保している。つまり、現時点では HF の方が、組織単位での管理を想定しており、ネットワークの入り口を制限している分きめ細やかなノード管理手法を提供しているといえる。

処理スピードについては、現状では HF が 2000tps (1秒辺りに処理できる取引数) であるとされており、Ethereum が 100tps、Quorum が数百 tps と言われているのに対して優位性を持っているが、現在 Ethereum は PoW から PoS への移行の最終段階にあり、これが成功すると 100,000tps にも達するといわれている。その場合、Ethereum をベースとしている Quorum の処理速度も向上する可能性が高い。

以上より、企業会計報告という目的からは、管理者有でネットワークの入り口を管理することの出来る HF が適していると考え、次節より、HF を用いて実践実験に入っていくこととした。

5. Hyperledger Fabric を用いた実践実験

5.1 システム構造について

DLT/BC プラットフォームはローカル PC 上に HF v2.4.2 を構築した。ワールドステートデータベースには couchDB を使用している¹⁰⁾。使用したメインプログラムは、「Fabric test network (以下、test network)」というサンプルプログラムで、これはローカル PC 上で Fabric について学習・稼働検証を行う目的で hyperledger-fabric web サイトにおいて提供されているものである。本番環境での想定される稼働と異なる点としては、test network は①2つのピア組織と一つの管理組織という最小構成であり、②単純化のため単一ノードでの Raft による合意形成を行うという2点である。

メインプログラムには、本稿での目的に沿っていくつかの修正を加えた。まず① hyperledger-fabric web によれば、データの検索・集計作業を BC に直接実行することはネットワークのパフォーマンスを低下させ、トランザクションを中断するため推奨されていな

い。その代わりに、BC ネットワークからデータを複製したオフチェーンデータベースに対して検索・集計作業を実行することを推奨している。オフチェーンデータベースは一度作成すれば以降のBCの変更を反映し、常にBCとの整合性は維持される。そこで、提供されているサンプルプログラム群の中から「off_chain_data」というプログラムをメインプログラムに取り入れ、実行できるように修正を行った。

②トークンの発行処理を行うため、当初サンプルには含まれていないトークン発行スマートコントラクト「token_erc_20」サンプルプログラムをメインプログラムに取り入れ、実行できるように修正を行った。

③Hyperledger Fabricでは通常Assetの創出、転送、情報の更新、消滅についての記帳を想定しているが、本稿では会計手続きに沿って処理を行うので、カテゴリーとしてAssetに加えLiability, Equity, Profit, Lossという表記を追加し、Asset同様に創出、転送、情報の更新、消滅の処理ができるようにスマートコントラクトの機能の追加を行っている。なお、各カテゴリーでの創出および情報の更新指示を行う場合は、指示に応じた「カテゴリーに所属している」ことを表すタグを自動的に付与するようにした。

④各カテゴリーのアイテムは「Asset・Liability・Equity・Loss・Profitのどのカテゴリーに所属するか」、「取引日¹¹⁾」、「(チャンネル全体で一意に定められた)追跡管理用の番号」、「登録名称」、「所有者」、「金額」のタグ情報が付与される。資産アイテムについては、上記のタグ情報に加え「数量」タグをつけている。追跡管理用の番号については、上記の各タグ情報は、創出、転送、情報の更新、消滅といった処理の過程で変更されていく可能性があるため、変更されない情報として追跡管理用の番号を付与しているものである。

これらの修正を行ったうえで、BC ネットワークおよびチャンネルの作成作業を行った。HFにおいては、各ノードはネットワークに参加するために組織に所属する必要がある。そのためtest network上には、Org1とOrg2という2つの組織が含まれ、Org1組織はpeer0.org1.example.com(以下peer0.org1)を、Org2組織はpeer0.org2.example.com(以下peer0.org2)という、それぞれ1つのノードを運営している。各ノードは取引の検証および、ブロックをBCに追加する役割を果たすが、取引の順序を決めたり新しいブロックを作成することはない。その役割を果たすのは管理ノード(下図、Orderer.example.com)でありtest networkには1つ含まれている。以上が今回構築したBC ネットワーク内のノードの説明である。

その他に、下図のようにcouchdb, caという表記が見られるがこれらはノードではない。たとえば、couchdb1はpeer0.org1.example.comの、couchdb2はorg2のそれぞれのノードが保有するワールドステートデータベースを表している。また、各管理ノードはネットワークへの参加にあたり、ネットワーク内での役割の定義と、(暗号通信を可能にする)TLS証明とネットワークへの登録のための証明書が生成される必要があるが、CA(Certificate

Authorities) がその証明書の生成を担っており、ca_orderer, ca_org1, ca_org2 はそれぞれの組織の認証を担当している。

```

ec0f03be3912 hyperledger/fabric-peer:latest Up 2 minutes 0.0.0.0:9051->9051/tcp, 7051/tcp, 0.0.0.0:9445->9445/tcp peer0.org2.example.com
"peer node start" 2 minutes ago
47907b7fff71 hyperledger/fabric-peer:latest Up 2 minutes 0.0.0.0:7051->7051/tcp, 0.0.0.0:9444->9444/tcp peer0.org1.example.com
"peer node start" 2 minutes ago
0334983eb9aa hyperledger/fabric-orderer:latest Up 2 minutes 0.0.0.0:7050->7050/tcp, 0.0.0.0:7053->7053/tcp, 0.0.0.0:9443->9443/tcp orderer.example.com
"orderer" 2 minutes ago
e2e2411e9fd7 couchdb:3.1.1 Up 2 minutes 4369/tcp, 9100/tcp, 0.0.0.0:7984->5984/tcp couchdb1
"tini -- /docker-ent..." 2 minutes ago
8f270fc6ee92 couchdb:3.1.1 Up 2 minutes 4369/tcp, 9100/tcp, 0.0.0.0:5984->5984/tcp couchdb0
"tini -- /docker-ent..." 2 minutes ago
d9f917ee99e0 hyperledger/fabric-ca:latest Up 2 minutes 0.0.0.0:8054->8054/tcp, 7054/tcp, 0.0.0.0:18054->18054/tcp ca_org2
"sh -c 'fabric-ca-se..." 2 minutes ago
e138b1f4f435 hyperledger/fabric-ca:latest Up 2 minutes 0.0.0.0:7054->7054/tcp, 0.0.0.0:17054->17054/tcp ca_org1
"sh -c 'fabric-ca-se..." 2 minutes ago
2246f0a94c72 hyperledger/fabric-ca:latest Up 2 minutes 0.0.0.0:9054->9054/tcp, 7054/tcp, 0.0.0.0:19054->19054/tcp ca_orderer
"sh -c 'fabric-ca-se..." 2 minutes ago

```

表2において述べたように、HFの特徴は取引関係者のみを含む「チャンネル」と呼ばれる構造を用いることで、チャンネル内と外での情報共有を規制する点にある。つまりBCネットワークを構成しただけでは、各ノードは互いに取引を開始することが出来ないため、チャンネルを作成し、各ノードをチャンネルに登録する必要がある。そこで、下記のようなチャンネルを作成する指示プログラムを実行する。この指示プログラム (network.sh) には、チャンネルの作成と、初期設定として各ノードおよび couchdb, ca の配置指示が入力されている。

```
./network.sh up createChannel -ca -s couchdb
```

それに対するシステム応答内容は、下図のようになる。

```

{
  "name": "mychannel",
  "url": "/participation/v1/channels/mychannel",
  "consensusRelation": "consenter",
  "status": "active",
  "height": 1
}

Channel 'mychannel' created
Joining org1 peer to the channel...
...(省略)
Joining org2 peer to the channel...
...(省略)
Anchor peer set for org 'Org1MSP' on channel 'mychannel'
...(省略)
Anchor peer set for org 'Org2MSP' on channel 'mychannel'
Channel 'mychannel' joined

```

ここまでの作業により、チャンネルの作成と、各ノードおよび couchdb, ca の配置が完了した。

5.2 仕訳処理の実践実験

さて、いま下記のような取引を BC を使って記録する Xyz があるとする。

4/1 Xyz は、トークン（名称：Coin）を 50 発行し、投資家から ¥500,000 を調達した。代金は預金口座に入金し、同額を元入金とし開業した。

4/3 Bank から現金 ¥300,000 を借入れ、預金口座に入金した。

4/5 備品 ¥300,000 を Em 社より買入れ、代金は預金から支払った。

4/9 Pt 社から商品 ¥200,000（¥10,000/個）を仕入れ、代金は預金から支払った。

4/15 St 社へ商品 ¥120,000（¥12,000/個）を売り渡し、代金が預金口座へと振り込まれた。

4/30 月次決算にあたり商品在庫を確認したところ、¥100,000 であった。

まず、4/1 に XYZ は開業資金調達のため 50 単位の Coin を発行し、投資家から ¥500,000 を調達した。以下は、1-2 行目が 50Coin の発行指示¹²⁾、3-4 行目は発行に成功したというシステム応答を示している。

```
test-network $ peer chaincode invoke "${TARGET_TLS_OPTIONS[@]}" -C mychannel -n token_erc20 -c '{"function":"Mint","Args":["50"]}'
2022-09-11 17:01:06.292 JST 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful
result: status:200
```

投資家は自身の ID を確認し、これを受領者 ID として Coin 発行者に連絡する。この ID は暗号化されているため、その連絡を受けた発行者が、正規の Coin 受領者の ID かどうか確認したい場合は、連絡された ID の暗号化を復元する必要がある。復元の結果、可読性のある形でもとの受領者情報を表示することができ、正しい Coin 受領者であるかを確認できる。ただし、BC 上での Coin 転送指示は復元前の暗号化された ID で行う必要がある。

下記の 1-3 行目は、受領者から連絡を受けた暗号 ID を RECIPIENT として指定する指示、4-5 行目は 50Coin を RECIPIENT へと送付する指示、6-7 行目は送付成功のシステム応答内容を示している。

```
test-network $ RECIPIENT="eDUwOTo6Q049cmVjaXBpZW50LE9VPWNsaWVudCxpPPU5cGVyYbGVkZ2VYLFNUPU5vcnRoIENhcm9saW5hLEM9VVM6OkNOPWNhLm9yZzIuZXhhbXBsZS5jb20sTzIvcmcyLmV4YW1wbGUuY29tLEw9SHVyc2xleSxTVdIIYW1wc2hpcmUsQz1VSw=="
test-network $ peer chaincode invoke "${TARGET_TLS_OPTIONS[@]}" -C mychannel -n token_erc20 -c '{"function":"Transfer","Args":["","$RECIPIENT","50"]}'
2022-09-10 03:24:38.735 JST 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful
result: status:200
```

Hyperledger Fabric を用いたブロックチェーン型簿記・会計システムへの実践実験

また、同時に RECIPIENT 側から残高を確認すると、受領の結果 50 へと変化したことが分かる（下記 1 行目が RECIPIENT 側ターミナルでの残高確認指示、3 行目が確認指示の結果として 50 の残高を示すシステム応答内容）。

```
test-network $ peer chaincode query -C mychannel -n token_erc20 -c '{"function":"ClientAccountBalance","Args":[]}'  
50
```

投資家から調達した資金は預金口座に入金されたものとして、同額を元入金とし開業する。下記 1 行目は、資産として「取引日：20220401、追跡管理用番号：B0001、登録名称：Xyz deposit1、所有者：Xyz、金額：500,000、数量：1」というアイテムを BC 上に創出する指示で、2 行目はその応答結果である。3 行目では、同日に追跡管理用番号：B0002 のアイテムを、登録名称：XyzEquity1 金額：500,000 の純資産（元入金）カテゴリーで創出する指示を出し、4 行目がその応答結果である。

```
+ node addAssets.js 20220401 B0001 XyzDeposit1 Xyz 500000 1  
Adding asset:Date:20220401 Type:asset ID:B0001 AssetName:XyzDeposit1 Owner:Xyz Price:500000 Quantity:1  
+ node addEquity.js 20220401 B0002 XyzEquity1 Xyz 500000  
Adding equity:Date:20220401Type:equity ID:B0002 EquityName:XyzEquity1 Owner:Xyz Price:500000
```

2 行目の「Type:asset」というのは前述のカテゴリー管理用のタグであり、B0001 は addAssets.js という指示で創出されたため、プログラムによって自動的に資産カテゴリーに属しているタグ (asset) が付与されている。

BC への記帳が成功したことを確認するため下記 1 行目のように、BC 上の記録の中から追跡管理用番号 B0001 を検索し、その状態を確認する指示を出す。2 行目はシステムからの応答内容であり、「取引日：20220401、追跡管理用番号：B0001、カテゴリー：Asset、登録名称：Xyzdeposit1、所有者：Xyz、金額：500,000、数量：1」というアイテムが記帳済みであることを示している。

```
+ peer chaincode query -C mychannel -n basic -c '{"Args":["ReadAsset","B0001"]}'  
{"Date":"20220401","ID":"B0001","Type":"asset","AssetName":"XyzDeposit1","Owner":"Xyz","Price":500000,"Quantity":1}
```

4/3 には Bank から現金 ¥300,000 を借入れ、預金口座に入金する取引を記帳する。まず、下記のように Bank の資産として、BankCash1（金額：300,000）を創出する指示を出し、BC への登録が完了したシステム応答を受ける。

```
+ node addAssets.js 20220403 B0003 BankCash1 Bank 300000 1
```

```
Adding asset:Date:20220403 Type:asset ID:B0003 AssetName:BankCash1 Owner:Bank Price:300000 Quantity:1
```

次に BankCash1 を Xyz に転送（つまり、資産の所有者の変更を行う）し、受け取った Xyz 側は、BankCash1 の名称を Xyzdeposit2 に更新する。同時に、Bank に対する負債である Xyzliability1 を創出する。下記 1 行目は、追跡管理用番号 B0003 のアイテムを Xyz に転送する指示である。3 行目は追跡管理用番号 B0003 のアイテムの名称を XyzDeposit2 へと更新する指示で、5 行目は liability1 を創出する指示である（システム応答については省略する。以下同様。）。

```
+ node transferAsset.js B0003 Xyz
```

```
Transferred asset B0003 to Xyz
```

```
+ node updateAsset.js 20220403 B0003 XyzDeposit2 Xyz 300000 1
```

```
Updated asset:Date:20220403Type:asset ID:B0003 AssetName:XyzDeposit2 Owner:Xyz Price:300000 Quantity:1
```

```
+ node addLiability.js 20220403 B0004 XyzLiability1 Xyz 300000
```

```
Adding liability:Date:20220403Type:liability ID:B0004 LiabilityName: XyzLiability1 Owner:Xyz Price:300000
```

4/5 に備品 ¥300,000 を買い入れ、代金は預金から支払った、という取引について記帳を行う。まず、ビジネス用備品販売業社 Em 社の資産として EmInventory1 を創出する。なお、Em 社は EmInventory1 を金額：300,000 で創出するが、これはあくまでこのチャンネル内での作業のみである。現実的には、販売先に対して仕入価格を秘匿するため、自社の販売部門と在庫管理部門とを含むサブチャンネルを設定するはずである。そしてメインチャンネルにおいて資産を登録する際には、サブチャンネル内に登録された資産を直接転送して使うのではなく、情報が漏れないように新規に粗利益を含めた金額で資産を創出する（同時にサブチャンネル内では当該資産の消滅と、売上の計上が行われる）手続きを取るはずだからである¹³⁾。下記 1 行目が追跡管理用番号 B0005 として EmInventory1 を創出する指示である。

```
+ node addAssets.js 20220405 B0005 EmInventory1 Em 300000 1
```

```
Adding asset:Date:20220405 Type:asset ID:B0005 AssetName:EmInventory1 Owner:Em Price:300000 Quantity:1
```

Em 社の資産登録が完了したため、下記 1 行目のように、Xyz へと転送を行う指示を出す。3 行目は、資産を受け取った Xyz がその名称を XyzEquipment1 へと更新する指示である。

```
+ node transferAsset.js B0005 Xyz
```

Hyperledger Fabric を用いたブロックチェーン型簿記・会計システムへの実践実験

Transferred asset B0005 to Xyz

```
+ node updateAsset.js 20220405 B0005 XyzEquipment1 Xyz 300000 1
```

Updated asset:Date:20220405Type:asset ID:B0005 AssetName:XyzEquipment1 Owner:Xyz Price:300000 Quantity:1

続いて下記 1 行目は、Xyz 所有の追跡管理番号：B003 (XyzDeposit2) を購入代金として Em 社へと転送する指示を出している。それを受けて 3 行目では、Em 社が受け取った B003 (XyzDeposit2) の名称を EmDeposit1 に更新する指示を出している。

```
+ node transferAsset.js B0003 Em
```

Transferred asset B0003 to Em

```
+ node updateAsset.js 20220405 B0003 EmDeposit1 Em 300000 1
```

Updated asset:Date:20220405Type:asset ID:B0003 AssetName:EmDeposit1 Owner:Em Price:300000 Quantity:1

4/9 の Pt 社から商品 ¥200,000 (¥10,000/個) を仕入れ、代金は預金から支払ったという取引を記帳する。下記は、1 行目で Pt 社について在庫 PtInventory1 を金額：200,000 数量：20 で創出し、3 行目で Xyz に転送したことを表す。受け取った在庫は、5 行目で名称の更新のみを行っている。

BC 型簿記会計システムは記帳のみではなく、転送（資産の所有権の移転）も行うため、三分法（仕入時に資産ではなく費用として計上し、販売時には売上を計上する）による処理は行うことができない。つまり、分記法、総記法、売上原価対立法のいずれかによる処理が必要となるため、資産として受け入れたうえで、名称変更のみを行っている。分記法、総記法、売上原価対立法はいずれの方法でも BC 型簿記会計システムに適合するが、本稿では決算整理仕訳の時点まで販売益が確定しないのは利害関係者の理解が得られないと考えられることと、売上額の把握が企業側に重要であると考えられることから、売上原価対立法を用いることとした。

```
+ node addAssets.js 20220409 B0006 PtInventory1 Pt 200000 20
```

Adding asset:Date:20220409 Type:asset ID:B0006 AssetName:PtInventory1 Owner:Pt Price:200000 Quantity:20

```
+ node transferAsset.js B0006 Xyz
```

Transferred asset B0006 to Xyz

```
+ node updateAsset.js 20220409 B0006 XyzGoods1 Xyz 200000 20
```

Updated asset:Date:20220409Type:asset ID:B0006 AssetName:XyzGoods1 Owner:Xyz Price:200000 Quantity:20

続けて、仕入代金 200,000 を Pt に転送する作業となるが、この時点で Xyz の所有する預

金は B0001 (名称: XyzDeposit1, 金額: 500,000) のみである。各アイテムの金額の一部のみを指定して転送する作業 (自動分割) は出来ないため、この場合資産を分割する作業が必要となる。

①500,000 の XyzDeposit1 を消滅させ、同時に XyzDeposit3 (XyzDeposit2 の名称はこの時点で使用されていないため再度創出することは出来るが、名称の再使用はミスを引き易くなるため避けることが望ましい) として金額: 300,000 のアイテムを創出、XyzDeposit4 として金額: 200,000 を創出する。

②500,000 の XyzDeposit1 の金額を 300,000 に更新し、XyzDeposit3 (金額: 200,000) を創出する。

③資産の創出時点でこうした事態に備えて分割可能な資産については利用最小単位で分割して創出する (例えば金額: 10,000 の Deposit50 個を作っておく、といったような処理)。

以上 3 つの方法が考えられるが、削除処理を行うと追跡・検証にかかる手間が増加してしまい、利用最小単位での分割登録はトランザクション処理速度の低下を招くこととなるため、②の処理が望ましいと考えられる。

以下、1 行目は B0001 (XyzDeposit1) の金額を 200,000 へと更新する指示、3 行目は B0007 (XyzDeposit3, Xyz 所有, 金額: 300,000, 数量: 1) を創出する指示、5 行目は B0001 (Xyz Deposit1, 金額: 200000, 数量: 1) を Pt に転送する指示、7 行目は B0001 の資産名称を PtDeposit1 に更新する指示である。

```
+ node updateAsset.js 20220409 B0001 XyzDeposit1 Xyz 200000 1
Updated asset:Date:20220409Type:asset ID:B0001 AssetName:XyzDeposit1 Owner:Xyz Price:200000 Quantity:1
+ node addAssets.js 20220409 B0007 XyzDeposit3 Xyz 300000 1
Adding asset:Date:20220409 Type:asset ID:B0007 AssetName:XyzDeposit3 Owner:Xyz Price:300000 Quantity:1
+ node transferAsset.js B0001 Pt
Transferred asset B0001 to Pt
+ node updateAsset.js 20220409 B0001 PtDeposit1 Pt 200000 1
Updated asset:Date:20220409Type:asset ID:B0001 AssetName:PtDeposit1 Owner:Pt Price:200000 Quantity:1
```

4/15 の St 社へ商品 ¥120,000 を売り渡し、代金が預金口座へと振り込まれた取引については下記ようになる。1 行目は St 所有の StDeposit1 (金額: 120,000, 数量: 1) を追跡管理番号: B0008 として創出する指示、3 行目は B0008 の Xyz への転送 (所有者の変更) 指示である。この指示によって、St 側からの支払いが完了する。5 行目は受領した B0008 の名称を XyzDeposit4 へと更新する指示、7 行目は転送のために、B006 (名称: XyzGoods1) を金額: 100,000, 数量: 1 に更新し、9 行目で B009 として金額: 100,000, 数量: 1 の Xyz

Hyperledger Fabric を用いたブロックチェーン型簿記・会計システムへの実践実験

Goods2 を新規に創出している。11 行目で B009 を St に転送している。これによって、商品の払い出しが完了する

```
+ node addAssets.js 20220415 B0008 StDeposit1 St 120000 1
Adding asset:Date:20220415 Type:asset ID:B0008 AssetName:StDeposit1 Owner:St Price:120000 Quantity:1
+ node transferAsset.js B0008 Xyz 120000 1
Transferred asset B0008 to Xyz
+ node updateAsset.js 20220415 B0008 XyzDeposit4 Xyz 120000 1
Updated asset:Date:20220415Type:asset ID:B0008 AssetName:XyzDeposit4 Owner:Xyz Price:120000 Quantity:1
+ node updateAsset.js 20220415 B0006 XyzGoods1 Xyz 100000 10
Updated asset:Date:20220415Type:asset ID:B0006 AssetName:XyzGoods1 Owner:Xyz Price:100000 Quantity:10
+ node addAssets.js 20220415 B0009 XyzGoods2 Xyz 100000 10
Adding asset:Date:20220415 Type:asset ID:B0009 AssetName:XyzGoods2 Owner:Xyz Price:100000 Quantity:10
+ node transferAsset.js B0009 St
Transferred asset B0009 to St
```

本稿では、売上原価対立法を用いているため、つづけて売上と売上原価を計上する。以下は 1 行目で、売上として、XyzProfit1 を追跡管理用番号：B0010、金額：120,000 として創出している。3 行目では、売上原価として、XyzCostOfSales1 を追跡管理用番号：B0011、金額：100,000 として創出している。

```
+ node addProfit.js 20220415 B0010 XyzProfit1 Xyz 120000
Adding profit:Date:20220415Type:profit ID:B0010 ProfitName: XyzProfit1 Owner:Xyz Price:120000
+ node addLoss.js 20220415 B0011 XyzCostOfSales1 Xyz 100000
Adding loss:Date:20220415Type:loss ID:B0011 LossName: XyzCostOfSales1 Owner:Xyz Price:100000
```

4/30 の月次決算にあたり商品在庫を確認したところ ¥ 100,000 であった。これについては、分記法、総記法を用いる場合は仕訳が行われるが、今回は売上原価対立法を用いているため、仕訳は発生しない。

以上のような取引の結果、Xyz の財務状況がどのようになったのかを確認する。下記 1 行目は Xyz 所有の「asset」という属性を持つアイテムのみを検索する指示である。

```
+ peer chaincode query -C mychannel -n basic -c '{"Args":["QueryAssetsByOwner","Xyz"]}'
[{"Date":"20220405","ID":"B0005","Type":"asset","AssetName":"XyzEquipment1","Owner":"Xyz","Price":300000,"Quantity":1},
```

```
"Date":"20220415","ID":"B0006","Type":"asset","AssetName":"XyzGoods1","Owner":"Xyz","Price":100000,"Quantity":10,|"Date
":"20220409","ID":"B0007","Type":"asset","AssetName":"XyzDeposit3","Owner":"Xyz","Price":300000,"Quantity":1,|"Date":"202
20415","ID":"B0008","Type":"asset","AssetName":"XyzDeposit4","Owner":"Xyz","Price":120000,"Quantity":1|]
```

この検索指示は BC ネットワーク上で行ったが、こうした検索指示はトランザクションを増大させ、ネットワークの処理速度を低下させることとなる。そこで、Xyz が持つすべてのカテゴリーの全アイテムの検索は、CouchDB 上に保存される最新のワールドステートに対して行い、出力する形とした。また、同じ理由から、出力結果を集計したり、報告用の書式へ整形する作業なども BC 外のプログラムで行う形を採用した。ブラウザ上で稼働する Javascript ベースのプログラムを作成し、Xyz 社の全てのアイテムについて、各取引日終了時点での集計結果を表示したものが次の図である。

Start date: 2022 / 04 / 01 End date: 2022 / 04 / 30 参照: nihonkaizai/output/20220415.txt

Asset	Liability&Equity	Loss	Profit
20220401/XyzDeposit1/Xyz/500000	20220401/XyzEquity1/Xyz/500000		
-	-	-	-
total:500000	total:500000	total:0	total:0

Asset	Liability&Equity	Loss	Profit
20220401/XyzDeposit1/Xyz/500000	20220403/XyzLiability1/Xyz/300000		
20220403/XyzDeposit2/Xyz/300000	20220401/XyzEquity1/Xyz/500000		
-	-	-	-
total:800000	total:800000	total:0	total:0

Asset	Liability&Equity	Loss	Profit
20220401/XyzDeposit1/Xyz/500000	20220403/XyzLiability1/Xyz/300000		
20220405/XyzEquipment1/Xyz/300000	20220401/XyzEquity1/Xyz/500000		
-	-	-	-
total:800000	total:800000	total:0	total:0

Asset	Liability&Equity	Loss	Profit
20220405/XyzEquipment1/Xyz/300000	20220403/XyzLiability1/Xyz/300000		
20220409/XyzGoods1/Xyz/200000	20220401/XyzEquity1/Xyz/500000		
20220409/XyzDeposit3/Xyz/300000			
-	-	-	-
total:800000	total:800000	total:0	total:0

Asset	Liability&Equity	Loss	Profit
20220405/XyzEquipment1/Xyz/300000	20220403/XyzLiability1/Xyz/300000	20220415/XyzCostOfSales1/Xyz/100000	20220415/XyzProfit1/Xyz/120000
20220415/XyzGoods1/Xyz/100000	20220401/XyzEquity1/Xyz/500000		
20220409/XyzDeposit3/Xyz/300000			
20220415/XyzDeposit4/Xyz/120000			
	NetIncome:20000	NetIncome:20000	
total:820000	total:820000	total:120000	total:120000

最後に、4/9時点でB0001という追跡管理用番号のアイテムはPtに転送されているが、前述したように、BCに記録された情報は改竄不可能であることから取引の履歴はBCが存在する限り追跡可能・検証可能であることを示す。下記は、1行目がB0001の履歴を検索する指示、2行目以降がB0001の履歴である。履歴は上から順に、新しい取引内容となっている。

```
+ peer chaincode query -C mychannel -n basic -c '{"Args":["GetAssetHistory","B0001"]}'
[{"record":{"Date":"20220409","ID":"B0001","Type":"asset","AssetName":"PtDeposit1","Owner":"Pt","Price":200000,"Quantity":1,"txId":"86d7f5de904cad62ae92015006e12a0ea8e644b1a700998a85083f6769315d03","timestamp":"2022-09-11T08:03:22.66Z","isDelete":false},"record":{"Date":"20220409","ID":"B0001","Type":"asset","AssetName":"XyzDeposit1","Owner":"Pt","Price":200000,"Quantity":1,"txId":"c36ada26a3ed6bdd23fe0a8660b63219a38b214efe2b29a6f85eb392e298f47f","timestamp":"2022-09-11T08:03:19.474Z","isDelete":false},"record":{"Date":"20220409","ID":"B0001","Type":"asset","AssetName":"XyzDeposit1","Owner":"Xyz","Price":200000,"Quantity":1,"txId":"3ffefaac1ca21b521dd87e49d3dd1b1ef7b32983d5db354c5531386a3e1df950","timestamp":"2022-09-11T08:03:10.948Z","isDelete":false},"record":{"Date":"20220401","ID":"B0001","Type":"asset","AssetName":"XyzDeposit1","Owner":"Xyz","Price":500000,"Quantity":1,"txId":"a7fa5ba3220916b9130c3f72de2f762d8b36fc2e64783b188930833bb1667a4","timestamp":"2022-09-11T08:01:59.873Z","isDelete":false}]
```

履歴の内容は4つの取引内容を表している。

```
①{"record":{"Date":"20220409","ID":"B0001","Type":"asset","AssetName":"PtDeposit1","Owner":"Pt","Price":200000,"Quantity":1,"txId":"86d7f5de904cad62ae92015006e12a0ea8e644b1a700998a85083f6769315d03","timestamp":"2022-09-11T08:03:22.66Z","isDelete":false}}
```

まず最新の履歴内容については、2022年9月11日の08:03:22.66Z¹⁴⁾(Zは協定世界時であることを表している)に行われた(86d7f5de904ca…というトランザクションID(txId)で管理される)取引の結果、このB0001という追跡管理用番号を付されたアイテムは、「2022/4/9日付の取引記帳で、追跡管理用番号：B0001、カテゴリー：asset、名称：PtDeposit1、所有者：Pt、金額：200,000、数量：1」として存在していることが分かる。なお、消滅している場合は末尾は"isDelete":trueと出力され、上記①の各タグの情報は空欄となる。この取引内容が一番上に表示されていることから、この取引が最新のものであることが分かる。

```
②{"record":{"Date":"20220409","ID":"B0001","Type":"asset","AssetName":"XyzDeposit1","Owner":"Pt","Price":200000,"Quantity":1,"txId":"c36ada26a3ed6bdd23fe0a8660b63219a38b214efe2b29a6f85eb392e298f47f","timestamp":"2022-09-11T08:03:19.474Z","isDelete":false}}
```

同日の08:03:19.474Zに行われた取引では、このB0001という追跡管理用番号を付されたアイテムは、「4/9日付取引、カテゴリー：asset、名称：XyzDeposit1、所有者：Pt、金

額：200,000、数量：1]へと情報が更新されたことを表している。

```
③ |"record":{"Date":"20220409","ID":"B0001","Type":"asset","AssetName":"XyzDeposit1","Owner":"Xyz","Price":200000,"Quantity":1,"txId":"3ffefaac1ca21b521dd87e49d3dd1b1ef7b32983d5db354c5531386a3e1df950","timestamp":"2022-09-11T08:03:10.948Z","isDelete":false}
```

同日の 08:03:10.948Z に行われた取引の結果、この B0001 という追跡管理用番号を付されたアイテムは、「4/9 日付取引、カテゴリー：asset、名称：XyzDeposit1、所有者：Xyz、金額：200,000、数量：1]へと情報が更新されたことを表している。

```
④ |"record":{"Date":"20220401","ID":"B0001","Type":"asset","AssetName":"XyzDeposit1","Owner":"Xyz","Price":500000,"Quantity":1,"txId":"a7fa5ba3220916b9130c3f72de2f762d8b36fc2e64783b188930833bbb1667a4","timestamp":"2022-09-11T08:01:59.873Z","isDelete":false}
```

同日の 08:01:59.873Z に行われた取引の結果、この B0001 という追跡管理用番号を付されたアイテムは、「4/1 日付取引、カテゴリー：asset、名称：XyzDeposit1、所有者：Xyz、金額：500,000、数量：1]として創出（この取引内容が一番下に記載され、この取引より古い履歴は存在しないため）されたことを表している。このように各アイテムはブロックチェーン上で、その創出、転送、情報の更新、消滅までを変更不可能な形で記録されているのである。

6. おわりに

これまで検証してきた通り、JSON 形式でのデータ保存と、検索用のタグの設定、それに分記法、総記法、売上原価対立法のいずれかを使用する事により、BC 上に格納されたデータは、複式簿記システムと統合させることが可能となる。こうした統合は、企業の簿記会計処理および会計報告にとって重要であるのみならず、複式簿記の持つ誤謬の発見機能を活用することで BC 技術の発展にも資することが期待される。

また BC を含む DLT を用いることでネットワークの参加者にとって、企業の取引データの検証可能性、追跡可能性は向上することも明らかとなった。ネットワーク参加者は（プラットフォームごとのトランザクション処理能力に依存するものの）タイムリーに企業の個別の取引を検証し、その流れを追跡することが可能になる。

当然企業側としては無制限の検証、追跡が望ましくない取引データも存在することが考えられるが、少なくとも企業向けの各プラットフォームには情報共有を制限する方法が存在している。HF の場合であればチャンネルを複数作成することで共有範囲を制限することが出来¹⁵⁾、GoQuorum であれば暗号化や PMTs の設定により取引の存在自体を非公開にできる。ただ、こうして多数の取引を非公開の状態にすると、年次の会計報告の数字と DLT のデータから利害関係者が作成した集計数字との間に齟齬が生じることとなる。この齟齬が大きけ

れば当然その企業は DLT 上での情報開示への積極性を疑われる結果を招き、資本コストの上昇を招くことが考えられる。したがって、企業は取引にかかわるプライバシーの確保の重要性と、利害関係者への開示の品質向上とそれに伴う資本コストの低下による利益のバランスを取る形で、情報の共有範囲および共有する内容を適切に設定することが必要となるであろう。

また、実践実験を通じて明らかになったのは、BC 上に取引に関するデータが改竄不可能で追跡可能な形で格納されることの会計報告上の革新性である。これまでの会計報告は、取引の結果を報告していたことから、結果に対する説明責任を果たしていれば問題はなかった訳だが、BC を用いた会計報告の場合は、各カテゴリーの科目について創出、転送、情報の更新、消滅までの過程の全てがいつでも確認可能となるため、取引の経過に対する説明が求められることになると考えられる。

例えば、企業経営者が、保有する証券を売却し、それを土地の購入に充てるとき、果たしてその取引は投資家から預かった資産に対する管理責任を十分に果たしたといえる取引であったのか。こうした問題に対して、既存の会計報告は十分な説明責任を求めてこなかった。既存の企業会計は一つの取引ごとに投資家に対してその是非を問うのではなく、説明は4半期ごとの財務報告で行い、最終的に株主総会においてその結果の是非を事後的に問うという形で管理責任を果たしているかどうかを判断してきた。

しかし、BC によってその取引を一つずつ確認することが可能になる場合には、結果説明責任のみを4半期ごとに果たすという形は説得力を持たない可能性がある。つまり、BC を用いた会計報告の時代には、個別の取引ごとに企業経営者の意図・判断を示す情報が提供されることが求められるようになると考えられるのである。

注

- 1) ここでの実践という表現は、BC および分散型台帳技術 (DLT) システム上に実際に稼働する簿記会計システムを構築し会計処理を行う、という意味で用いている。
- 2) 日本ブロックチェーン協会による定義は、「ビザンチン障害を含む不特定多数のノードを用い、時間の経過とともにその時点の合意が覆る確率が0へ収束するプロトコル、またはその実装をブロックチェーンと呼ぶ」。「電子署名とハッシュポイント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を広義のブロックチェーンと呼ぶ」とされている。
- 3) ただし、このアンケートは利用目的および利用状況についての区別を行わずに実行されたものであるため、BC 技術の利用可能性を検討する目的での利用も含んでいる。またその他に、顧客に提供するサービスの一部、IOT、生産のトレーサビリティ、通運の追跡といった目的など多様な利用形態が含まれている可能性がある。
- 4) 調査時点の名前。現在は GoQuorum と呼ばれる。
- 5) Hyperledger には Fabric や Sawtooth をはじめとした複数のプロジェクトがあるため、Hy-

- perledger という表記にはそのうちのいずれか、もしくは、複数の採用の可能性がある。
- 6) Raft は、チャンネル内の管理を担当するノードの中からリーダーが動的に選出され、フォロワーノードはリーダーノードによって伝播されたデータを信頼して複製する形の合意形成手法。仮に、リーダーノードが停止した場合は、十分な数のノードが残っている限り、リーダーの再選出が行われ、システムは維持される。KAFKA との違いは、KAFKA が単一組織での運用が前提であるのに対して、RAFT では複数組織間での運用が可能なため、ネットワークの停止が発生しにくくなる。
 - 7) PoW (Proof-of-Work) とは、BC に新しいトランザクションを追加する権利を得るために、PC の処理能力を用いて複雑な数学的パズルを解くことを求める方法。Ethereum では新たに PoS (Proof-of-Stake) を用いるチェーンの導入を進めている。PoS とは、BC に新しいブロックを追加する権利を得るために、ETH を「賭ける (Stake)」することを求める方法である。Stake に参加したノードの中からブロックを作成する権利が抽選されるが、同時に、他者が作成することになったブロックについても検証する責任が生じる。この検証を怠ると Stake した金額の一部を失うことが、また故意に共謀して不正行為に加担すると全金額を失うことがある。
 - 8) PoA (Proof of authority) とは、信頼済みのノードの中からブロックの生成を行うノードが選ばれる方法。ブロックの生成を担当するノードは順次交代していく。参加ノードの信頼性について懸念がある場合に適しているがブロックの分岐 (フォーク) が発生する可能性がある。Clique POA は、Quorum に統合された PoA の個別名称である。
 - 9) BFT は、ネットワーク内のノードの故障や、誤った情報伝達が予想される状況において、全体として正しい合意形成を可能にする方法。リーダーノードはトランザクション処理要求を受け取ると、他のノードに処理要求を転送する。各ノードはトランザクションの検証を行い、結果について相互に通信を行う。一定の台数から処理要求について問題が無いという返信が集まった時点で、トランザクション処理の実行が行われる。
 - 10) HF において各ノードは BC の他にワールドステートを保存している。BC が全ての取引の履歴であるのに対して、ワールドステートはネットワークの最新の状態を表すデータベースである。スマートコントラクトはワールドステートに対して取引提案を処理している。ワールドステートの利用により、取引履歴全体を検索し計算しなくとも、最新の値に直接アクセスすることができる。HF のワールドステートには LevelDB と CouchDB の使用が可能であるが、CouchDB を使用することでデータを JSON 形式で保存し、データに対して JSON クエリを発行できる。会計報告を目的とする場合、データの入力・保存はもちろんだが、検索・集計・理解というプロセスが必要となる。この際、人間にとって理解が容易な形式である必要があるため、JSON 形式での保存が可能な CouchDB を採用した。
 - 11) 取引日は、記帳者が入力する取引の日付情報であり、実際にこの記帳処理を実行した日付である必要はない。
 - 12) 正確には、コインの発行を担うスマートコントラクトに対して、必要な変数を渡し、呼び出している。「創出」、「転送」、「更新」、「消滅」といった指示も、同様にその機能を担うスマートコントラクトに対して必要な情報を渡して、呼び出すことを指している。
 - 13) このように取引先等が、その社内で利用するネットワークと BC ネットワークの統合が果たせない場合には、出島のようにメインネットワーク上に小規模な取引用組織を構築し、企業内部

Hyperledger Fabric を用いたブロックチェーン型簿記・会計システムへの実践実験

のネットワークにおいて記録されている資産を削除すると同時に、取引用組織が資産を新たに記録する形を取ることが可能である。ただし、両者のネットワークを統合できる場合と比較して、資産情報の連続性は切断され、切断の前と後での追跡可能性は低下する。

- 14) 「2022-09-11T08:03:22.66Z」はシステムが記録した実際にこの記帳処理が実行された日時であり、各アイテムが持つ日付タグ情報の方は、記帳者が入力する取引の日付情報である。
- 15) 複数チャンネルの応用可能性については板橋・ガルシア（2021）を参照のこと。

参考文献

- BLOCKDATA, 2021, 「BLOCKCHAIN ADOPTION BY THE WORLDS TOP 100 PUBLIC COMPANIES (REPORT)」, BLOCKDATA.
- Deloitte, 2019, 「Deloitte's 2019 Global Blockchain Survey」, Deloitte Insights.
- Deloitte, 2020, 「Deloitte's 2020 Global Blockchain Survey」, Deloitte Insights.
- Deloitte, 2021, 「Deloitte's 2021 Global Blockchain Survey」, Deloitte Insights.
- Ethereum web:<https://ethereum.org/ja/developers/>, アクセス日時 2022.2.1
- GoQuorum (旧名 Quorum) web:<https://consensys.net/docs/goquorum/en/latest/>, アクセス日時 2022.2.1
- Harish.Natarajan, Solvej.Krause, Helen.Gradstein, 2017, 「Distributed Ledger Technology and Blockchain (Working Paper)」, *FinTech Note No. 1*, World Bank.
- hyperledger-fabric web, https://hyperledger-fabric.readthedocs.io/en/latest/test_network.html, アクセス日時 2022.2.1.
- hyperledger-fabric off_chain_data sample, https://github.com/hyperledger/fabric-samples/tree/main/off_chain_data, アクセス日時 2022.2.1.
- hyperledger-fabric token-erc-20 sample, <https://github.com/hyperledger/fabric-samples/tree/main/token-erc-20/chaincode-go>, アクセス日時 2022.2.1.
- Pascual Pedreño E., Gelashvili V., Pascual Nebreda L., 2021 「Blockchain and its application to accounting.」 *Intangible Capital*, 17(1), 1-16. <https://doi.org/10.3926/ic.1522>.
- 板橋雄大・ガルシア, 2021, 「分散型台帳技術の企業会計への適用に関する検討：情報移動手段としてのサイドチェーンの活用」, 『東京経済大学会誌（経営学）』 310号, pp. 95-108.
- 坂上学, 2016 『事象アプローチによる会計ディスクロージャーの拡張』, 中央経済社。