第　3　部　第3章

教育目的でのブロックチェーンの応用

長尾　雄行

目　次

１．はじめに

ブロックチェーンとは、ピア・ツー・ピア・ネットワーク（Ｐ２Ｐネットワーク）上で取引履歴等の台帳を共有し、ネットワーク上の多様な参加者により共同・分散管理する技術である。ブロックチェーン専用の通信･暗号化等の機能を備えた多数の計算機（ノード）がインターネット等のネットワーク上で接続され、中央集中的な管理機能を備えた計算機がすべてのノードを制御することがなくても、各ノードが事前に定められたルールに従って自律的に稼働することで、同一の共有台帳を同期して保持することができる。ブロックチェーンの起源とされる論文がNakamoto（2009）によって発表されてから12年ほど経過した現在､送金･支払い等の目的で利用される多数の暗号資産がブロックチェーンによって実現している。一方で、ブロックチェーンの暗号資産以外への用途を開拓することはまだ未知の領域であり重要な課題となっている。

本稿では、教育機関における教育活動へブロックチェーンを応用することを想定して、ブロックチェーンの概念と基礎的な技術要素、さらに、教育への応用例についての現状を整理する。そして､イーサリアムのプライベートネットワークの構築と､その上でのスマートコントラクトを用いたアプリケーションの試作を通じて、ブロックチェーンを教育へ適用する場合における応用上の課題を抽出する。

２．ブロックチェーン

ブロックチェーンの技術要素の一つはＰ２Ｐネットワークである。Ｐ２Ｐネットワークでは多数のノードが対等な立場で接続して互いにデータを交換することで、同一のデータを同期して維持することができる｡このようなデータ交換に加え､ブロックチェーンのネットワークに参加する各ノードでは、時間の進行と共に生じる取引履歴等のデータを時系列に整列して、ブロックと呼ばれる単位に集約した上で、タイムスタンプを刻印して記録する。そして、新たな取引履歴を含むブロックを、過去のブロックへと連鎖させ、チェーンと呼ばれるデータとして記録する｡この連鎖の仕組みには､公開鍵暗号技術や暗号学的ハッシュ関数を応用して、改ざんが困難となるような設計が用いられる。

既存のチェーンに新たなブロックをつないでチェーンを伸ばす場合には、任意に作成したブロックをそのままチェーンに記録するのではなく、事前に定められた他ノードとの合意形成のためのアルゴリズム（コンセンサス・アルゴリズム）に従って有効と判断される

ブロックを生成する必要がある。有効なブロックを新たに生成してチェーンを延長する作業をマイニングと呼び、マイニングの役割を担うノードをマイナーと呼んでいる。一般的に、マイニングはブロックチェーンを維持するために必須の作業である。

現時点で最も普及しているコンセンサス・アルゴリズムはProof-of-Work（PoW）である。Proof-of-Workでは、新たなブロックを追加する際に、そのブロックに関する大きな計算量を必要とする暗号問題を他のノードと競って解き、解を得たノードがブロックの追加権を得るという仕組みが取られている。ここで用いられる暗号問題には、解を見つけることは困難であるが、いったん解を見つけてしまえば、それが解であることを第三者が容易に検証することができる問題が採用されている。解を発見したノードには報酬としてトークンが支払われる。一般のノードはその解を検証した上で新たに追加されたブロックを受け入れる。Proof-of-Workのもとでは、マイニングに多大な計算量とそれに伴う莫大な電力が必要となる。

ブロックチェーンのＰ２Ｐネットワークに参加するノードは、幾つかのノード（ピアノード）とブロックチェーンやそれに付随するデータ（接続先のピアノードの候補一覧等）を常に交換しており、どのノードも同じ台帳状態へ近づけるようにデータを同期している。接続先のピアノードは不定であり、その数はノードのソフトウェアで自動的に調整される。ノードがピアノードから新しいブロックを受信した場合には、送信元のノードが攻撃者の用意した悪意のあるノードとなっている可能性も考えられる。そこで、各ノードは受信したブロックを電子署名等によって検証する。そして、検証により有効と確認できたブロックをチェーンに記録する。この方式では、ブロックとブロックをあたかも信頼の鎖でつなぐようにデータを記録するため、全体の共有台帳データをブロックチェーンと呼んでいる。このようにして作られたブロックチェーンは、正常に稼働している場合には、新たな取引履歴等のデータを含むブロックをマイナーが追記することは可能であるが、過去の記録の削除や変更は、共有台帳上ですべて記録・共有されているため、容易には実現できない。

インターネット上で公開されたブロックチェーンを利用する場合には、共有台帳を必ずしも信頼関係の無い参加者と共同運営することになる。従って、共有台帳に掲載する情報には機密情報・個人情報等を無加工で含めることは困難となる。また、公開されたブロックチェーンには単一の所有者が存在せず、多様な参加者がコンセンサス・アルゴリズムにもとづいて合意形成を行うため、従来の中央集中的データベースのような管理者や所有者というものが存在しない。ブロックチェーンの運営は実質的にノード用ソフトウェアの開発者・マイナー・一般のノード利用者等の関係者が共同で実施するものと見なすことができる。これらの関係者のうちマイナーは特に有力な立場を有している。新規ブロックの作

成という重要な役割を担うからである。

公開されたブロックチェーンにおいては、ノードを稼働させてネットワークに参加する者であれば国籍・所属組織等の如何を問わず対等に扱われる。参加者は特定の管理者や管理組織を信頼するのではなく、ブロックチェーン・ネットワーク全体を信頼して暗号資産・トークン・データ等の取引を実施することになる。一度公開ブロックチェーンに掲載された情報については、多数の複製がネットワーク上に作成され、削除することが事実上不可能となる。また、情報の共有相手の特定も困難となる。従来から行われている Web サイト上での情報公開とブロックチェーンによる台帳の共有を比較すると、ブロックチェーンではデータの掲載時にその複製がネットワーク上に多数作成される点が大きく異なっている。従来の Web サイトであれば、誰も閲覧しなければ情報の複製は生じないが、ブロックチェーンでは情報の送信時点での複製が前提となっている点に注意が必要となる。どのような情報をブロックチェーンに掲載するのかについては、このような同期の性質を考慮して慎重に設計を行うことが求められる。Greenspan（2016）が指摘するように、ブロックチェーンは機密性を犠牲にすることで、信頼できる仲介者に依存せずに一つのデータベースを互いに信頼していない参加者間で共有する技術である。

ブロックチェーンにおいては、コンセンサス・アルゴリズムによる多数決に基づいて意思決定を行うため、ネットワークへの参加者のうち、善良な参加者が多数派となっていることが前提となる。あるグループが多数派であるかどうかは、そのグループを構成する個人や組織の数ではなく、そのグループが稼働するノード全体の計算能力にもとづいて判断される。公開されたブロックチェーンの場合には、悪意のある参加者が含まれることを前提として運用を行わなければならないが、そのような参加者が少数派にとどまっている限り、合意形成の障害とはならず、ブロックチェーンの改ざんは事実上行えない。ノードを稼働させるためには、高性能の CPU、GPU（Graphics Processing Unit）や ASIC（Application Specific Integrated Circuit）等を搭載した計算機に加え、それらを稼働させるネットワークと電力が必要となる。従って、多数派を維持してブロックチェーンの運用を継続するために、常に機材と電力を消費し続けることになる。ノードを稼働させるために各ブロックチェーンのプロトコルに対応したソフトウェアも必要となる。ビットコインの場合には、オープンソースの Bitcoin Core [1] が、また、イーサリアムの場合には、Go Ethereum [2] がその例である。一般の個人・企業・組織等はこれらのソフトウェアを稼働させることでブ

1　https://github.com/bitcoin/bitcoin
2　https://github.com/ethereum/go-ethereum

ロックチェーンへ参加することができる。

運用時には現在主流の方式である Proof-of-Work への攻撃にも注意が必要である。この方式では攻撃者が用意した悪意のあるマイナーの割合が過半数を占める場合には、ブロックチェーン全体を実質的に支配可能であることが知られている。このようなブロックチェーンへの攻撃を 51％攻撃と呼ぶ。Bitquery（2020）は、2020 年 7 月 29 日から同 8 月 1 日にかけて、マイニング専用のサービスを利用してイーサリアムクラシックに 51% 攻撃が実施され、ブロックチェーンの再編が生じ、同一暗号資産を不正に 2 重支払いすることで、暗号資産を窃取した事例を観測している。この事例が示唆するように、Proof-of-Work を用いる場合には、教育機関等で独自のブロックチェーンを公開する形で運用することは困難である。独自の公開ブロックチェーンでは参加者が少ないため、マイニング専用サービスやクラウド等を用いて過半数を超えるノードが作成しやすくなり、ブロックチェーン自体が脆弱となるからである。対抗策としては、マイナーを信頼できる特定の参加者へ限定すること、プライベートなブロックチェーンを利用すること、又は、既存の大規模ブロックチェーンを利用することが考えられる。

ブロックチェーンの主要用途はビットコインやイーサリアムをはじめとする暗号資産とその取引を実現することである。暗号資産の時価総額の一覧を掲載する Web サイトである coinmarketcap.com によると、本稿執筆時点では、時価総額が最大の暗号資産はビットコインであり、時価総額は 3,400 億ドル以上となっている。また、時価総額が 10 億ドル以上の暗号資産が 25 件となっている。従来の銀行券や貨幣とは異なり、ブロックチェーンにもとづく暗号資産には発行者が必ずしも必要ない。この点は、中央銀行や政府等の発行者が存在する従来の通貨とは性質が異なっている。ブロックチェーンに参加する多様な主体の運用するノードが対等に扱われ、チェーンの選択に関してコンセンサス・アルゴリズムに基づく多数決による合意形成を行うことができるので、中央集中的にデータやノードを管理する役割を担う信頼できる第三者が不要となる。従って、単一の発行者が存在しなくともブロックチェーンは自律的に機能する。このような自律性について、小出（2020）は「ビットコインの主な貢献は、皆が第三者を必要とせずにネットワーク全体を信用する状況を作り出す、トラストレス（trustless）の分散型信用基盤を実現したことにある。」と指摘している。また、信頼できる第三者の排除がブロックチェーンの有効活用に必要と述べている。ここでいうトラストレスとは、検証結果に合意を得るために参加者が互いに信頼し合う必要がないというブロックチェーンの性質のことである（Infante 2019）。

一般的にブロックチェーンでは、ブロックのマイニングに大きなコストが必要となるため、ブロックに掲載できるデータ量は限られている。そこで、ブロックチェーンには掲載

したいデータそのものではなく、暗号学的ハッシュ関数を用いて計算したハッシュ値（そのデータを代表する一定のデータ長の符号値）を掲載し、データ本体は別のデータベースで共有する方式が用いられる。ビットコインでは OP_RETURN スクリプト命令を用いて最大 83 バイト長の任意データをブロックに掲載することができる（Bartoletti 2017）ので、ハッシュ値の掲載に OP_RETURN 命令が用いられる。イーサリアムでは、イーサリアム仮想計算機（EVM、Ethereum Virtual Machine）の状態を保存・共有するために木構造のデータのハッシュ値をブロックヘッダーに掲載し、木構造データ本体は別のデータベースで管理する方式を採用している。そのためイーサリアムでは、各アカウントの状態として文字列・整数値等を含む構造化されたデータを保持し、スマートコントラクトと呼ばれるプログラムでそのデータを操作することが可能となっている。

３．ブロックチェーンの応用

現在では、ブロックチェーンの用途は暗号資産にはとどまらず、様々な分野への応用も検討されている。Greenspan（2016）はブロックチェーンの利用事例の分類として４種類を挙げている。すなわち、軽量な金融システムの構築（クラウドファンディング等）、貴重品の来歴追跡、複数組織間の記録管理、そして、多者間でのデータ集約である。Grech（2017）らは、教育へのブロックチェーンの応用はまだ初期段階にあると指摘した上で、ブロックチェーン上にデータを記録することに関して次の６点の意義があると述べている。すなわち、（１）個人が自分自身のデータを管理・制御できる自己主権性、（２）支払いや証明書の取引を行う基盤としての信頼性、（３）どの組織でも取引に参加が可能であることを把握した上で取引する透明性、（４）記録が作成された後には変更ができない不変性、（５）取引や記録を管理するための仲介者が不要であること、及び、（６）参加者同士が仲介者を経由せずに直接取引できる点である。このうち（１）、（３）、及び（６）は P2P を応用したアプリケーションとしての特徴であり、（２）、（４）、及び（５）は暗号学的ハッシュ関数や電子署名を応用してデータを検証可能にしたブロックチェーンの特性であると考えられる。

大学におけるブロックチェーンの応用に関する Fedorova（2020）らの調査では、応用分野として、（１）修了証書の発行と記録、（２）認証のための利用、（３）知的財産の保護、（４）学生と教員のための新しいネットワークの構成、（５）アカデミックパスポート（Portfolio）の構成、（６）暗号資産での支払い、（７）教育機関の認証評価、さらに、（８）教育プロセスの管理が存在していると報告されている。一般的に、（１）の教育成果についての証

明書を電子化するためにブロックチェーンを応用する事例は多数知られており、経済産業省の調査（経済産業省 2019）においても、Blockcerts[3]等の事例が列挙されている。

証明書の電子化については、Mozilla のプロジェクトとして開発されているオープンバッジ（Open Badges[4]）がその手段の一つとしてよく利用されている。オープンバッジを用いると学習成果の証明書を電子データで記録し、共有・再配布することができる。Open Badges をブロックチェーンへ応用した事例として、Blockcerts が知られている。Blockcerts はブロックチェーンを用いて教育に関する証明書を発行するためのオープンソース・ソフトウェアである。経済産業省の調査（経済産業省 2019）によると、証明書の発行者は複数の証明書をまとめてマークル木を用いて表現し、そのルートハッシュ値をブロックチェーンに記録するとされている。この方式は、イーサリアムで用いられている、状態データベースのハッシュ値をブロックチェーンへ記録する方法と類似の手法である。

Sony Global Education（2016）は、暗号化された学習履歴データを二者間で安全に交換するブロックチェーンの応用技術を 2016 年に発表している。公式サイト[5]によると、Hyperledger Fabric[6]によりネットワークを実装している。ここで、Hyperledger Fabric は Linux Foundation によって開発されている汎用的な分散台帳構築用ソフトウェアである。

Mikroyannidis（2018）らによる Smart Blockchain Badges では、このような学習成果の電子証明を更に発展させ、イーサリアムと Smart Contract を活用してデータサイエンスを学ぶ学生に仕事や学習コースの紹介を行うシステムを提案している。また、Mikroyannidis（2020）では、生涯学習のためのプラットフォームとして、セマンティック Web の分野における研究成果をもとに、ブロックチェーンを併用することで、学習者が自分自身の教育関連情報を自己管理できるエコシステムを提案している。それに加えて、Semantic Blockchain と呼ばれる提案では、従来の LMS（Learning Management System）を拡張して、ファイルの保存には分散ファイルシステム IPFS（InterPlanetary File System）を取り入れ、個人情報を自己主権的に管理するためのソーシャルプラットフォームである Solid を用い、認証には自己主権型 ID（Self-sovereign Identity）を活用した拡張を行い、さらに、データの完全性を保証する仕組みとしてブロックチェーンを利用する試みが実施されている。

3　https://www.blockcerts.org/
4　https://support.mozilla.org/ja/products/open-badges
5　https://blockchain.sonyged.com/
6　https://www.hyperledger.org/use/fabric

４．ビットコイン

Blockchain.com[7]によると、ビットコインのジェネシスブロック（最初のブロック）は2009年１月４日に記録されており、2020年12月14日現在では、マイニング済みのビットコインの流通量は約1,860万ビットコインとなっている。また､ブロックチェーンのデータ量は約316GBとなっている。さらに、これまでの総トランザクション数は約５億9,590万件となっている。また、2020年12月14日のデータについては、過去24時間における平均ブロックサイズは約1.2MBであり、単一ブロックに含まれるトランザクション数は過去24時間平均で2,168件となっている。さらに、トランザクションが承認されて共有台帳に載るまでに要する時間は過去24時間平均で約34分となっている。

これらの情報に加え、一般的にビットコインでは約10分に一度新しいブロックが作成されることを考慮すると､トランザクション数は１分間におよそ200件前後と推定される。新しいブロックについては後続の複数のブロックが承認されて初めてネットワーク全体で承認されたと考えるポリシーを採用する場合には、実際のトランザクションが完了するまでには数時間を要するものと思われる。また、ビットコインのブロックチェーンについてのデータ量は現時点で市販されている記憶装置（SSDまたはHDD）に記録することが可能な量となっている。従って、ブロックチェーン・データのみを同期するだけで、マイニングは行わないとすれば、個人や小規模組織でもノードを作成してビットコイン・ネットワークへ参加することが現時点では可能となっている。

ビットコインのブロックチェーン・データは専用のツールを利用することで誰でも取得することができる。例えば、Bitcoin Coreを用いてフルノードを稼働させれば過去に遡ってすべてのデータを取得することができる。Webサイト上でもBlockchain.comにおいて、ブロックについての情報、マイニングされたビットコインの情報、さらに、参加者間の取引情報を取得することができる。ユーザの残高を把握するには、そのユーザに関わるUTXO（Unspent Transaction Output）と呼ばれる取引データをブロックチェーンから収集して残高を計算する必要がある。

ビットコイン・ネットワークで利用される通信プロトコルは公開されている。例えば、Original Bitcoin client（2015）を参照されたい。ビットコイン・ネットワークでは、getaddrメッセージにより各ノードが保有するアクティブなピアの情報を問い合わせるこ

7　https://www.blockchain.com

とができる。Bitnodes[8]では、この仕組みを利用してパケット解析を行うことにより、ビットコインネットワーク上で稼働するノードのIPアドレスやユーザエージェント（ノード用ソフトウェアのバージョン情報）を収集している。2020年12月15日現在、ビットコインのノードは古い方式のものを除いて11,224ノード検出されている。これらのノードのうち、国籍未詳のものが約24%、米国が約17%、ドイツが約16%、そして、日本のノードが約２％である。

ビットコインの電力消費量を推計するサイトCambridge Bitcoin Electricity Consumption Index（https://cbeci.org/）によると、2020年12月11日時点で年間およそ90.72TWhの電力を消費しているとされる。小出（2020）によると2019年12月時点では73TWhと報告されているので、１年で２割以上の消費電力の推計値の増加が見られる。Enerdata（2020）では、2019年の日本国内の電力消費量は918TWhとされているので、ビットコインのために、国内の電力消費量の約10%程度に相当する量の電力が消費されていると推測される。これらの情報にもとづくと、コンセンサス・アルゴリズムとしてProof-of-Workを用いた場合には、トラストレスであってもブロックチェーンが構成できるという利点がある一方で、世界規模での電力消費に大きな影響が出るほどブロックチェーンの維持にコストが必要となるという課題が読み取れる。

５．イーサリアム

ビットコイン・ブロックチェーンでは取引履歴等のデータを記録することができることに加え、さらに、取引に関する処理をスクリプトと呼ばれるスタックベースの簡易プログラミング言語で記述し、トランザクション内で実行することができる。しかしながら、ビットコイン・スクリプトは繰り返し処理ができず、汎用的なプログラムを実行することはできないように意図的な設計が行われている。一方で、イーサリアムはチューリング完全であることを目指して設計されているため、データに加えて更に汎用的なプログラムもブロックチェーン上で共有・実行することが可能となる。

イーサリアムの特徴はブロックチェーンに参加する各ノードでEthereum Virtual Machine（EVM）と呼ばれる仮想計算機が稼働して、ブロックチェーン上に記録されたプログラムを実行することができる点にある。イーサリアムでは、ブロックチェーン上で稼働するプログラムをスマートコントラクトと呼んでいる。スマートコントラクトを実現

8　https://bitnodes.io/

するために、各アカウントに残高以外の情報（整数値、文字列、バイト列、及び、それらの組み合わせ等）も紐付けて保存することができるように設計されている。

イーサリアムはアカウントの状態をトランザクションによって遷移させる方式で設計されている。稼働中の各ノードはアカウントの状態を管理するための状態データベースを保持しており、その中に各アカウントが保有しているイーサリアムの残高やアカウント固有の情報が記録されている。この状態データは修正マークル・パトリシア木と呼ばれる特別なデータ構造で記録されており、ブロックチェーン上には、その状態データのハッシュ値のみが記録される方式となっている。ここでいう修正マークル・パトリシア木は、キーと値の組み合わせからなる連想配列の実装方法の一つであり、連想配列全体とそれに含まれる値のハッシュ値を計算することが容易なデータ構造である。詳細は Ethereum Wiki の Patricia Tree の項目[9]を参照されたい。

Wood（2020）によると、イーサリアムのアカウント状態には、そのアカウントのトランザクション総数（nonce）、保有残高（balance）、ストレージのルートハッシュ値（storageRoot）、および、アカウントに登録された EVM バイトコードのハッシュ値（codeHash）が含まれる。ここでいうストレージとは、ノードが保持している各アカウントに紐付いた連想配列のことであり、アカウント状態にはそのストレージのハッシュ値のみが記録されている。ここでいう連想配列も修正マークル・パトリシア木によって実装されている。あるアカウントから別のアカウントへ送金が行われる場合には、送金情報がトランザクションとしてブロックに記録され、ネットワークへ同期される。各ノードはそのトランザクションにもとづいて状態データを検証した上で残高を更新することにより決済が実現する。

イーサリアムでは、状態データのハッシュ値とトランザクションによる状態遷移を用いて、各ブロックにアカウントの残高を直接記録せずに、状態データを同期・検証する方式が採用されている。この方式の利点としては、残高情報以外の文字列・バイト列・整数値等を含む任意のデータを状態に記録し、トランザクションでそのデータを利用することができるので送金以外の目的でのアプリケーションが作成しやすい点にある。一方で、送金以外のデータも記録できることからデータ量が増大しやすく、データの検証により長い時間が必要となるという課題が見受けられる。

イーサリアムについての情報を公開している Web サイト Etherscan[10] によると、イーサリアムのジェネシスブロックは 2015 年 7 月 30 日に記録されている。2020 年 11 月 4 日

9　https://eth.wiki/fundamentals/patricia-tree
10　https://etherscan.io/

時点の記録では、Ethereum ノードは 9,247 ノード稼働しており、Go Ethereum クライアントに関するブロックチェーンのサイズは（2020 年 11 月 4 日のブロック番号が 11,187,359 の時点で）、過去の不要な状態を除去する標準設定のもとで約 542GB となっており、さらに、過去のすべての状態を保存するアーカイブ設定のもとでは約 5.572TB となっている。これらの情報をもとに考えると、現時点では市販の記憶装置を用いて個人や小規模組織がイーサリアムのブロックチェーンの同期を行うことも可能である。一方で、よりイーサリアムの普及が進み送金以外のデータが増大した場合には同期が困難となる可能性も考えられる。

イーサリアムのトランザクションには、メッセージ・コールとコントラクト生成の２種類が用意されている。メッセージ・コールを用いると、アカウント間でのイーサリアムの取引やスマートコントラクトの関数呼び出しを行うことができる。独自のスマートコントラクトをネットワークに登録するには、コントラクト生成トランザクションを用いて事前にスマートコントラクトを実現するためのバイトコード（EVM 用の機械語）をネットワークへ登録し、コントラクト専用のアドレスを取得しておく必要がある。

イーサリアム・ノードの最もよく使われている実装である Go Ethereum（Geth）のブロックチェーンデータはファイルシステム上のデータベースに記録されている。具体的には、Google によるキーバリューストア型のデータベースである LevelDB を用いて、状態データベース等を修正マークル・パトリシア木へ符号化してディスクへ記録している。符号化の際には、RLP（Recursive Length Prefix）と呼ばれる手法で木構造をもつバイナリデータを直列化している（Wood 2020）。一方で、ディスクに記録された情報を別のプログラムから読み取ることは容易ではない。その理由の一つは、ブロックチェーン上には状態データベース等の複雑なデータ構造のハッシュ値のみが記録されていることにある。ある時点での完全な状態データベースをプログラム上で再構築するためには、ブロックチェーン上のトランザクションを実行して状態遷移を再現しなければならないため、イーサリアム・クライアントと同等の機能が必要となる。従って、ブロックチェーンや状態データベース等のデータを別のアプリケーションで利用するには、イーサリアム・クライアントと直接対話する仕組み必要となる。そこで、Geth では、JSON-RPC と呼ばれる遠隔呼び出し手続きを実装しており、HTTP、WebSocket、及び、UNIX ドメインソケット経由で問い合わせが行えるようになっている。問い合わせのためのクライアントとして、JavaScript 言語用には Web3.js[11] が開発されている。また、Python 言語用には Web3.py[12] が知られて

11　https://github.com/ethereum/web3.js/

12　https://github.com/ethereum/web3.py

いる。イーサリアムを用いたアプリケーションを開発する場合には、保存されたデータを直接利用するのではなく、ノードを稼働させ、ノードに対して遠隔呼び出しを利用することになる。

イーサリアムのプロトコルを用いて作られたネットワークは複数存在している。Ethereum の開発者向けサイト[13]によると、公開ネットワークとして、暗号資産イーサリアムの取引のための実運用環境であるメインネットとプロトコルやスマートコントラクトを試験するためのテストネットが知られており、現時点ではメインネットには Proof-of-Work が用いられている。一方で、テストネットには利用目的別に複数の種類が存在している。現在多くのテストネットでは Proof-of-Authority（Szilágyi 2017）という、限られたノードのみがブロック生成を担当する方式のコンセンサス・アルゴリズムが採用されている。一方で、Ropsten テストネットでは Proof-of-Work が採用されている。Szilágyi（2017）によると Proof-of-Authority は、PoW で稼働するテストネットへ行われる攻撃の対抗策として設計されたコンセンサス・アルゴリズムであり、ブロックの生成と署名はサイナーと呼ばれるノードが担当する。ブロックチェーン稼働当初については、サイナーはジェネシスブロックに記録された信頼できる一つ又は複数のノードが担当する。ブロックチェーン稼働中はサイナーが投票して合意形成することによってサイナーの追加・削除が可能となっている。特定のサイナーが独占してブロック生成することを回避するために、単一のサイナーが連続して署名できるブロックの個数に制限が設けられている。

６．イーサリアムのプライベートネットワーク

イーサリアムの公開ネットワークやテストネットワークを用いずに、独自のプライベートネットワークを構築して運用することも可能である。ここではプライベートネットワークの構築手順を大まかに説明する。はじめに、イーサリアムのノードで利用するデータを保持するためのディレクトリを作成し、利用するアカウントのキーペアとアドレスを生成しておく。キーペアは秘密鍵と公開鍵の対のことで、秘密鍵はアカウントの本人確認のための情報でありパスフレーズで保護されている。公開鍵は秘密鍵から生成される鍵であり、公開鍵のハッシュ値を加工したものがアドレスである。アドレスは銀行の口座番号のように利用者を識別するために利用する。

次に、コンセンサス・アルゴリズムを選定する。選択肢は Proof-of-Work（Ethash エ

13　https://ethereum.org/ja/developers/docs/networks/

ンジンを利用する方法）と Proof-of-Authority（Clique エンジンを利用する方法）の２種類である。教育機関において独自のネットワークを構成して利用する場合には、参加機関数やノードの性能と数量に限りがあるため、必然的にネットワークは小規模となる。一般的に、小規模ネットワーク上の Proof-of-Work ではブロックチェーンのセキュリティを保つことが困難である。攻撃者がクラウドやマイニング専用の計算機やサービスを用いて大量のノードを作成してマイニングすることによって 51% 攻撃を実施し、不正取引を実施することが可能となってしまうからである。従って、小規模ネットワークでは、ネットワークの接続制限を行うか、または、限られたノードのみが新ブロックを生成できる Proof-of-Authority を利用する方式が妥当と考えられる。

引き続いて、ジェネシスブロックの設定を行う。ジェネシスブロックとはブロックチェーンの最初のブロックのことであり、初期アカウントとその残高やコンセンサス・アルゴリズムの設定を記入して作成することができる。Ethereum のジェネシスブロックはブロックチェーン構築支援用の puppeth ツールで作成した設定ファイルを用いて生成することができる。同じネットワークへ参加するノードは同一のジェネシスブロックからブロックチェーンを開始する。Proof-of-Authority を利用する場合には、サイナーとして稼働するノードのアドレスをジェネシスブロックに一つ以上登録しておく。また、ブロックを生成する頻度をジェネシスブロックで設定することができる。標準設定では 15 秒間隔でブロックを生成する設定となっている。ブロック生成の間隔が短いほど記録されるデータ量が増えるため、長期間の運用を前提に適切なブロック生成頻度をネットワーク毎に検討する必要がある。

以上の準備の後、ノードを初期化してファイルシステム上にデータベースを構築することでイーサリアムの実行環境の準備が整う。利用するネットワークについては、TCP 及び UDP ポートの開放設定も行う必要がある。利用するポート番号は設計時に任意に設定することができる。開発段階のネットワークでノードの発見を支援するための手段として、bootnode と呼ばれるツールが用意されている。これは接続相手の IP アドレスを配信する機能のみを持つ簡素化されたノードである。

プライベートネットワークの構成例を図 １に示す。この構成例では、ノード１からノード５の５台からなるプライベートネットワークが稼働しており、そのうち、ノード１とノード３がサイナーである。実線はイーサリアムプロトコルによる通信を意味し、点線はそれ以外の通信を意味している。このプライベートネットワークを Microsoft Teams、Slack、あるいは LINE 等の外部サービスへ連携させる場合には、図 １のように中継サーバをイーサリアムの JSON-RPC API でいずれかのノードへ接続し、中継サーバと SNS サイト等

を Web Hook API で連携させる手法が考えられる。

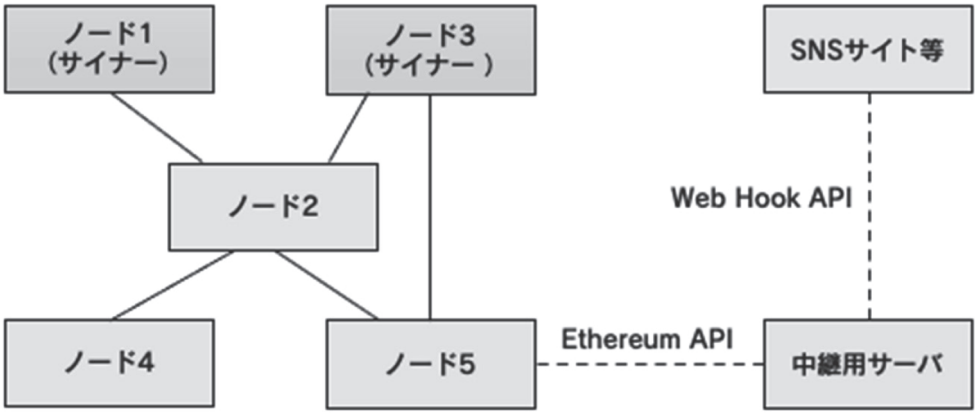

図１　イーサリアムのプライベートネットワークの例

７．スマートコントラクト

スマートコントラクトとは、イーサリアム・ブロックチェーン上で稼働するプログラムのことである。スマートコントラクトを単にコントラクトとも呼ぶ。ブロックチェーン上のアカウントに任意のデータを保存したり、そのデータを利用したアプリケーションを実装したりすることが、コントラクトを用いて実現できる。イーサリアムのスマートコントラクトはブロックチェーン上のバイトコードとストレージで表現される。バイトコードは EVM 用の機械語であるため、プログラマが直接作成することは困難である。そこで、スマートコントラクトの実装のための専用のプログラミング言語を用いる。その一つが Solidity 言語である。この言語では Java 言語や C# 言語のようなオブジェクト指向言語に類似した文法が採用されている。

Solidity 言語でコントラクトを実装し、バイトコードへとコンパイルしたものを特別なアドレス（ゼロアドレス）宛のトランザクションの添付データとして送信することで、コントラクトをブロックチェーン上へ配備して利用を開始することができる。コントラクトの配備時にはコントラクトアドレスと呼ばれる専用のアドレスが生成される。コントラクトアドレスに対してトランザクションを送信することでコントラクトのバイトコードを実行することができる。バイトコードを実行するとアカウントの内部状態が更新され、その結果がストレージ情報として記録される。そして更新後のストレージ情報のハッシュ値がブロックに掲載される。ネットワーク上の多数のノードで同時に実行及び検証することに

よって、ネットワーク全体でコントラクトが実行される。

Solidity の公式ドキュメンテーション[14]によると、Solidity 言語のデータ型には、最大256 ビットの整数型・マッピング型（簡易的な連想配列）・構造体・コントラクト型等が準備されているが、小数が利用できないなど、一般的なプログラミング言語とは仕様が異なる点に注意が必要となる。標準で用意されているのは、符号付き・符号なしの整数値型（8 ビットから 256 ビットまでの 8 ビット単位のビット長が利用可能）、ブール型、アドレス型（20 バイトでイーサリアムのアドレスに対応）、コントラクト型（オブジェクト指向言語のクラスに類似のもの）、固定長のバイト列（1 バイト長から 32 バイト長まで）、可変長のバイト列、配列、文字列型、構造体、マッピング型（キーと値の対からなる簡易的な連想配列）、及び、enum 型が用意されている。日付・時刻表現のための型は標準では用意されておらず、ブロックのタイムスタンプでは uint 型を用いたエポック秒が利用されている。小数については、バージョン 0.8.0 の段階では、2 進固定小数点方式での設計の一部が用意されているが、まだ未実装となっているため、アプリケーション側で整数型を用いた 10 進固定小数点方式を実装するなどの対処が必要と思われる。また、標準のマッピング型では、キーではなくそのハッシュ値が記録されており、キー全体の巡回ができなくなっている点に注意が必要である。Solidity では値として Null が許されないことから、従来のデータベースとは異なり、値が無いことを示す方法が別途必要となることにも注意が必要となる。整数と構造体を用いたデータ構造については、従来の言語と大きな違いは無いが、文字列・日付・時刻・小数・連想配列の取り扱いには設計上の工夫やライブラリの開発が必要と思われる。

イーサリアムのコントラクトは不変的性質を持ち、一度登録したコントラクトは修正することができない。コントラクトを修正して再度配備すると、修正前とは別の新しいコントラクトと新しいコントラクトアドレスが生成され、修正前と修正後の両者が利用可能となる。コントラクトを利用する場合には利用対象のコントラクトアドレスへトランザクションを送信する（状態変化を伴う場合）か、または、関数呼び出しを行う（状態変化を伴わない場合）。状態変化を伴う関数を設計する場合には、トランザクションが非同期的に実行される点にも留意して設計を行うが必要となる。例えば、状態変化を伴う関数の戻り値がトランザクションの場合には戻り値としては取得できないため、イベント機能を用いて状態変化後のデータを読み出す工夫が必要となる。イーサリアムのコントラクトアド

14　https://docs.soliditylang.org/en/v0.8.0/types.html

レス一覧はブロックチェーン上に記録されているため、ネットワークへの悪意のある参加者がコントラクトアドレスを収集して、攻撃用のトランザクションを送信してくることも想定して攻撃耐性のあるコントラクトを設計及び実装する必要がある。

図２にスマートコントラクトの実装例を示す。このコントラクトは、指定された任意の文字列に対して正の整数を順に割り当てるものであり、ユーザ ID を文字列として受け取ると、それに対して１から始まるシリアル番号を割り当てるものである。アプリケーション側ではユーザ ID を事前に匿名化してから利用することを想定している。

このコントラクトでは、内部状態として count 変数により最後に発行したシリアル番号が記録されている。getCount() 関数は count の値を読み出すための状態変化を伴わない関数である。ユーザ ID とシリアル番号の対応はマッピング型の toSerial 変数によって表現されており、getSerial() 関数によって特定のユーザ ID に割り当てられたシリアル番号を問い合わせることができる（割り当てが無い場合には、0 が戻される）。getSerial() も内部状態を変化させない関数である。新しい文字列を登録するための関数が register() であり、こちらは状態変化を伴う。count 変数を増加させ、toSerial 変数にもキーと値を追加するからである。トランザクション送信により関数 register() を開始すると、ネットワーク上で承認される毎に Registered() イベントが発生する。このイベントの発生は Web3.js 等のクライアント上で捕捉することができる。現在の Web3.js の実装で試したところ、１回のトランザクションにつき 24 回までの承認イベントが捕捉可能であった。

```
pragma solidity ^0.8.0;
contract Serializer {
    uint private count = 0;
    mapping (string => uint) private toSerial;

    event Registered(uint serial);

    function getCount() public view returns (uint) {
        return count;
    }

    function getSerial(string calldata _uid)
        public view returns (uint) {
```

```
        return toSerial[_uid];
    }

    function register(string calldata _uid) public {
        require(toSerial[_uid] == 0, "User is already registered.");
        toSerial[_uid] = ++count;
        emit Registered(count);
    }
}
```

図 2　Solidity 言語の利用例（文字列にシリアル番号を割り当てるコントラクト）

一般に、ブロックチェーン上の状態データの更新頻度には制限がある。イーサリアムでは標準では 15 秒程度を目安にブロックが生成される。プライベートネットワークでは、あらかじめジェネシスブロックにおいてブロック生成周期を設定することができるが、オンラインのリアルタイムアプリケーションが要求するほどの高頻度なブロック生成は不可能と思われる。従って、イーサリアムのデータを利用したアプリケーションや可視化手法については、更新頻度が 15 秒程度を前提にシステムを設計する必要がある。このようなデータの低頻度更新に適合する機器の一例としては、電子ペーパーが挙げられる。電子ペーパーでは、一般に画像や文字の表示のための画面書き換えに数秒から数十秒程度の時間が必要となる。画面書き換え速度は利用する電子ペーパーモジュールの色数にも依存し、色数が多いほど書き換えに要する時間が長くなる。ブロックチェーンと電子ペーパーを連携させて新たな教育用のツールを開発することも可能と思われる。次項ではそのための試作システムを紹介する。

8．試作システム

本研究では、イーサリアムのプライベートネットワークと外部サービス及び電子ペーパーの併用が可能であることを確認する目的で、プライベートネットワークをクラウド上に構築し、LINE ビーコンとスマートフォンを連携させて、スマートフォンに利用者固有のシリアル番号を通知するサービスを試作した。このサービスは、授業やイベントなどで来訪者の人数を計測し、来訪者に先着順の番号を割り当てるサービスを意図している。

利用したクラウド環境は Amazon Web Services（AWS）の EC2 であり、Ubuntu Linux による３台のイーサリアムノードを準備してプライベートネットワークを構築した。試作システムの概要を図３に示す。この試作システムでは、LINE Messaging API の Web Hook へ接続するための中継サーバを EC2 上に別途準備した。ビーコン端末は Raspberry Pi Zero W v1.1 と電子ペーパーモジュール（WaveShare 2.13 インチ 212 × 104 E-Ink 赤 / 黒 / 白 ３色）を用いて作成した。このビーコン端末では Bluetooth Low Energy（BLE）の無線通信を用いて定期的にビーコン ID を発信している。スマートフォンは無線通信でビーコン ID を受信すると、LINE アプリ経由で LINE 社のサービス（図 3 では LINE と表記）へそのビーコン ID を通知する。LINE 社のサービスから中継サーバへはビーコン ID と匿名化されたユーザ ID 等の情報が通知される。中継サーバでは、匿名化されたユーザ ID をもう一度匿名化した上でスマートコントラクト内の register() 関数の引数として設定し、JSON-RPC 経由でトランザクションとして送信する。トランザクションの承認イベントを検出すると、新しく発行されたシリアル番号を LINE Messaging API 経由でスマートフォンへ送信する（図５）。ビーコン端末では、定期的に getCount() 関数を呼び出して発行済みのシリアル番号の件数を取得して電子ペーパーの画面へ表示する（図４）。

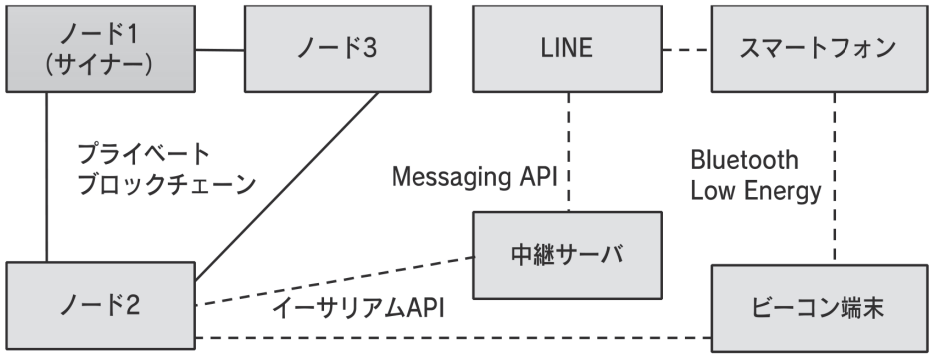

図３　LINE ビーコンを用いてスマートフォンにシリアル番号を通知するサービス

試作システムでは、１秒に１回ブロックを生成する仕様でジェネシスブロックを作成し、PoA によるプライベートネットワークを構築した。ノード３台のうち、１台をサイナーに割り当てた。特に問題なくブロック生成が行われ、ビーコンを用いたシリアル番号の発行や電子ペーパーの画面表示の更新も想定通り動作した。試作システムの構成では１台のみとなっているサイナーに障害が発生するとシステム全体が新しいブロックを生成できなくなり、シリアル番号の発行が行えなくなる。耐障害性・保守性を向上させるためには、

サイナーを複数台用意する設計に変更する必要がある。中継サーバでは JavaScript API である Web3.js を用いてトランザクション処理を実装した。一方で、ビーコン端末では Python API である Web3.py を用いて類似の処理を実装した。試作システムの実装の範囲ではこの両者の API に大きな違いは見られなかった。

図４　ビーコン端末

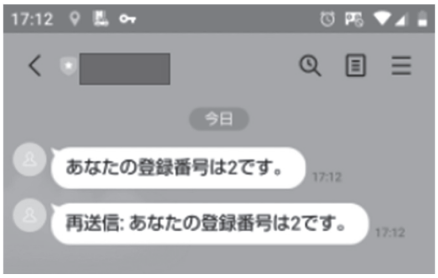

図５　シリアル番号の通知

９．Dapps

イーサリアムはプログラム自体を非集中的に稼働させるプラットフォームとして設計されている。データベースとしてブロックチェーンを利用し、そのデータをスマートコントラクトと呼ばれるプログラムで処理することで決済等の処理をネットワーク上で実施することができる。このような仕組みを利用してイーサリアム・ブロックチェーン上で稼働する非集中型の分散アプリケーションを Dapp（decentralized application）と呼んでいる。

Etherscan の Dapps 一覧[15] によると、2020 年 12 月 28 日時点で 416 件の Dapp が登録されており、カテゴリ別にゲームが 50 件、金融が 42 件、マーケットプレイスが 14 件、プラットフォームが 24 件、サービスが 7 件、その他が 275 件、そして、カテゴリなしが４件となっている。この中には、すでに活動を停止しており、公式 Web サイトが閲覧できないものも含まれている。Etherscan の Web サイト上では、各 Dapp に概要と紹介文が記されている。概要のテキストに含まれる単語のうち、出現回数が 20 回以上であるものの一覧を表１に示す。ただし、出現頻度の計算は Scikit-learn の CountVectorizer を用いて実施し、一般的な英語語句を除外する設定のもとで行った。前述の 416 件の Dapp について概要を確認したところ、本稿のテーマである教育機関に関わる Dapp は登録されておらず、Dapp の主な用途はトークンの構成・暗号資産の取引となっていた（この点は表１の結果からも推測できる）。トークンや暗号資産以外のテーマとしてはオークション・

15　https://etherscan.io/dapp/listing

創作物の売買・ゲーム内のデジタルアセットの取引・IoT 等が見られる。

表１　etherscan.io に登録されている Dapps の概要に含まれる主なキーワード

キーワード	出現回数
blockchain	108
token	101
platform	67
ethereum	59
decentralized	54
crypto	48
Game	45
Digital	43
Based	43
Smart	41
Tokens	37
World	35
Protocol	32

キーワード	出現回数
Network	31
Trading	29
exchange	28
cryptocurrency	27
contract	27
ecosystem	26
users	25
assets	25
use	23
real	23
project	23
defi	23

これらの Dapps は公開されたネットワーク上でイーサリアムを活用するものであり、ブロック生成のためのマイニング費用が無視できない。教育に関わるデータやスマートコントラクトを公開ブロックチェーンに掲載して稼働させることは、教育関係のデータを常に攻撃者に晒すことを意味する。スマートコントラクトも常に不正使用の可能性を考慮して設計し、24 時間 365 日監視し続けなければならない。これらを考えると、教育機関で Dapps を一般公開して維持管理することは困難である。一方で、今回の試作システムのように PoA によるプライベートネットワーク上で Dapp を稼働させる場合には、ブロック生成のためのマイニング費用が不要となり、ノードを稼働させるサーバとネットワークの維持費が中心となるため、単一の大学の構内で利用する場合や、複数の大学を連携させて小規模に活用する場合には、共同管理が行える共有データベースとプログラミングの環境として活用することができると考えられる。

10. おわりに

本稿では、教育での応用を想定して現在最も利用されているブロックチェーンであるビットコインとイーサリアムの概要を整理した。教育目的でインターネット上のブロックチェーンを利用することに関しては、証明書の発行等での先行事例があるものの障壁が多い。

PoW を用いた公開ネットワークの場合には、小規模ネットワークでは 51% 攻撃による乗っ取りや悪意のあるブロックチェーン再編の可能性があるため、ビットコインやイーサリアムのような大規模で多様な参加者が利用しているネットワークを活用する必要がある。この場合には、マイニング費用とそれに伴うエネルギー消費、掲載するデータの選別と暗号化、ブロックチェーンへの攻撃対策、コントラクトへの攻撃対策、ブロック生成速度の限界、及び、ノードのデータ量等を考慮して、教育関係のデータやコントラクトがブロックチェーンに掲載して共同運営するべきかどうかを事前に検証する必要がある。

一方で、PoA を用いるプライベートネットワークの場合には、マイニング費用が不要となり、攻撃への対抗策として従来のファイアウォール等が利用できるので参入への敷居は低い。大学内で小規模に PoA を用いたネットワークを開始し、コントラクトの作成方法やそのセキュリティ対策に十分習熟した上で、学外の組織との連携を図る流れが妥当と思われる。特に、スマートコントラクトにおいては、インターネット上の共有領域にデータとプログラムを配置するという、従来のセキュリティ対策では否定されてきた行為を前提とするため、新しい視点からのセキュリティ対策が求められる。

PoA によるネットワークをアクセス制限のもとで限られた組織間で共同運用することは可能と思われる。本稿の試作システムで示したように、ブロックチェーンのノードはプライベートネットワークで稼働させて一般公開せず、アクセス制限を行った API 経由で既存の Web アプリケーションやスマートフォンから間接的に利用する運用形態が妥当と思われる。PoA によるネットワークをインターネット上で完全に公開して長期間安全に利用できる設計や運用方法を見いだすことは今後の課題である。

コンセンサス・アルゴリズムについては、本稿で扱った PoW と PoA 以外にも、Proof-of-Stake（Wood 2020）をはじめとする多数の選択肢が検討されている。教育目的のシステムでの利用に適したコンセンサス・アルゴリズムを明らかにすることも課題の一つである。特に、参加者が少ない場合でも攻撃が困難であり、かつ、低コストで長期間運用できる仕組みを見いだすことは未解決の問題である。

暗号資産としての機能を除外して考えると、ブロックチェーンは世界全体あるいは一部のグループが共同運用する低頻度時系列データの共有データベースであり、固定長のハッシュ値を掲載してタイムスタンプを押印し、参加者の誰もが改ざんされていないことを検証可能という特徴を持っている。このようなハッシュ値のタイムスタンプ機能をブロックチェーン外のデータベースやアプリケーション等において活用する方法やそのためのミドルウェアを開発することも今後の課題である。

互いに信頼していない者同士がブロックチェーンを共同運営するというトラストレスの概念を教育関係に適用するとすれば、資格試験・入学試験・定期試験の問題・解答・採点をブロックチェーンで共同運営することが考えられる。ステークホルダとしては、出題者・受験者・採点者・講師・試験運営者・ブロックチェーン運営者・アプリケーション開発者等が考えられる。試験をブロックチェーン上で実現するには、指定時刻になって初めて試験問題や回答が読み取れるような暗号化方式、参加者の属性による試験問題や解答へのアクセス制限、採点結果を暗号化してブロックチェーンへ登録する仕組み、及び、受験者の得点を証明する仕組みが必要となる。これらの課題を解決し、トラストレス方式でインターネット上での試験運営と資格認定を実現すれば、受験者はいつでも好きなタイミングで資格試験を受験することができ、しかも合格証を検証可能な形でブロックチェーン上に記録することができる環境を創出することができる。

参考文献

Protocol documentation.（n.d.）2021 年 1 月 3 日閲覧 https://en.bitcoin.it/wiki/Protocol_documentation

Enerdata.（2020）「グローバルエネルギー統計イヤーブック」2020 年 12 月 10 日閲覧 https://yearbook.enerdata.jp/electricity/electricity-domestic-consumption-data.html

経済産業省（2019）「平成 30 年度産業技術調査事業（国内外の人材流動化促進や研究成果の信頼性確保等に向けた大学・研究機関へのブロックチェーン技術の適用及びその標準獲得に関する調査）報告書」

小出俊夫（2020）「ブロックチェーンのデータ構造と動作原理」『通信ソサイエティマガジン』電子情報通信学会、14 巻 1 号、12-18 ページ

Addy Yeow Chin Heng（2020）Bitnodes. 2020 年 12 月 15 日閲覧 https://github.com/ayeowch/bitnodes

Bartoletti, M. and Pompianu, L.（2017）An analysis of Bitcoin OP_RETURN metadata.

Lecture Notes in Computer Science. doi:10.1007/978-3-319-70278-0_14

Bitquery（2020）Attacker Stole 807K ETC in Ethereum Classic 51% Attack. 2021 年 1 月 4 日閲覧 https://bitquery.io/blog/attacker-stole-807k-etc-in-ethereum-classic-51-attack

Fedorova, E. P. and Skobleva, E. I.（2020）Application of Blockchain Technology in Higher Education. European Journal of Contemporary Education.

Grech, A. and Camilleri, A. F.（2017）Blockchain in Education. Publications Office of the European Union.

Greenspan, G.（2016）Four genuine blockchain use cases. 2020 年 12 月 29 日 閲 覧 https://www.multichain.com/blog/2016/05/four-genuine-blockchain-use-cases/

Infante, R.（2019）Building Ethereum Dapps. Manning Publications.

Jirgensons, M. and Kapenieks, J.（2018）Blockchain and the future of digital learning credential assessment and management. Journal of Teacher Education for Sustainability, 20（1）, 145-156.

Kodate, A.（2019a）. Open Badges とは？その応用と e ラーニングへの実装について . 2021 年 1 月 4 日閲覧 . https://bit.ly/3hCVLIf

Kodate, A.（2019b）. Blockcerts（ブロックサーツ）の開発経緯とメリットについて . 2021 年 1 月 4 日閲覧 . https://bit.ly/2X6hmPF

Mikroyannidis, A.; J. Domingue, J.; Bachler, M. and Quick, K.（2018）Smart Blockchain Badges for Data Science Education. 2018 IEEE Frontiers in Education Conference（FIE）, San Jose, CA, USA, 2018, 1 -5. doi: 10.1109/FIE.2018.8659012.

Mikroyannidis, A.; Third, A.; J. Domingue, J.; Bachler, M. and Quick, K（2020）Blockchain Applications in Lifelong Learning and the Role of the Semantic Blockchain. In: Sharma, Ramesh Chander; Yildirim, Hakan and Kurubacak, Gulsun eds. Blockchain Technology Applications in Education. IGI Global, pp. 16–41.

Nakamoto, S.（2009）Bitcoin: A Peer-to-Peer Electronic Cash System. 2021 年 1 月 3 日閲覧 https://bitcoin.org/bitcoin.pdf

Nazaré, J. and Hamilton, K. Digital Certificates Project. 2021 年 1 月 4 日閲覧 https://certificates.media.mit.edu/

Original Bitcoin client.（2015）2020 年 12 月 15 日 閲 覧 https://en.bitcoin.it/wiki/Original_Bitcoin_client

Patricia Tree, Ethereum Wiki.（2020）2020 年 12 月 27 日閲覧 https://eth.wiki/fundamentals/

patricia-tree

Sony Global Education.（2016）Sony Global Education Develops Technology Using Blockchain for Open Sharing of Academic Proficiency and Progress Records. 2020年12月9日閲覧 https://www.sony.net/SonyInfo/News/Press/201602/16-0222E/

Szilágyi, P.（2017）EIP-225: Clique proof-of-authority consensus protocol. 2021年1月11日閲覧 https://eips.ethereum.org/EIPS/eip-225

Tapscott, D. and Tapscott, A.（2017）2020年12月9日閲覧 https://er.educause.edu/articles/2017/3/the-blockchain-revolution-and-higher-education

W3C. Verifiable Credentials Data Model 1.0. 2021年1月4日閲覧 https://www.w3.org/TR/vc-data-model/

Wood, G.（2020）Ethereum: A Secure Decentralized Generalized Transaction Ledger Petersburg Version 3e2c089-2020-09-05. 2020年12月26日閲覧 https://ethereum.github.io/yellowpaper/paper.pdf