Ethereum

株式会社Neo Breakthrough
山科 優希

~~Ethereum Whitepaperを読む~~

Ethereum Whitepaperについて

Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform.
By Vitalik Buterin (2014).

When Satoshi Nakamoto first set the Bitcoin blockchain into motion in January 2009, he was simultaneously introducing two radical and untested concepts. The first is the "bitcoin", a decentralized peer-to-peer online currency that maintains a value without any backing, intrinsic value or central issuer. So far, the "bitcoin" as a currency unit has taken up the bulk of the public attention, both in terms of the political aspects of a currency without a central bank and its extreme upward and downward volatility in price. However, there is also another, equally important, part to Satoshi's grand experiment: the concept of a proof of work-based blockchain to allow for public agreement on the order of transactions. Bitcoin as an application can be described as a first-to-file system: if one entity has 50 BTC, and simultaneously sends the same 50 BTC to A and to B, only the transaction that gets confirmed first will process. There is no intrinsic way of determining from two transactions which came earlier, and for decades this stymied the development of decentralized digital currency. Satoshi's blockchain was the first credible decentralized solution. And now, attention is rapidly starting to shift toward this second part of Bitcoin's technology, and how the blockchain concept can be used for more than just money.

Commonly cited applications include using on-blockchain digital assets to represent custom currencies and financial instruments ("colored coins"), the ownership of an underlying physical device ("smart property"), non-fungible assets such as domain names ("Namecoin") as well as more advanced applications such as decentralized exchange, financial derivatives, peer-to-peer gambling and on-blockchain identity and reputation systems. Another important area of inquiry is "smart contracts" - systems which automatically move digital assets according to arbitrary pre-specified rules. For example, one might have a treasury contract of the form "A can withdraw up to X currency units per day, B can withdraw up to Y per day, A and B together can withdraw anything, and A can shut off B's ability to withdraw". The logical extension of this is decentralized autonomous organizations (DAOs) - long-term smart contracts that contain the assets and encode the bylaws of an entire organization. What Ethereum intends to provide is a blockchain with a built-in fully fledged Turing-complete programming language that can be used to create "contracts" that can be used to encode arbitrary state transition functions, allowing users to create any of the systems described above, as well as many others that we have not yet imagined, simply by writing up the logic in a few lines of code.

*Ethereum Whitepaper*は*2014*年に*Vitalik Buterin*によって発表され、*2015*年に実装された。

*Ethereum*を一言で（個人的に）表すと、

『Public Blockchain上にチューリング完全なプログラムの実行環境を搭載したWorld Computer』

であると考えている。

*Bitcoin*と比べて*Whitepaper*が長いため、時間の都合上本セミナー中の全体説明は控える。

テキスト教材に原文・日本語訳リンクを紹介してあるので、より深く学ばれたい方はそちらを参照いただきたい。

*Whitepaper*の冒頭部にのみ触れ、以降は要点のみを掻い摘んで説明する。

冒頭部要

Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform.
By Vitalik Buterin (2014).

When Satoshi Nakamoto first set the Bitcoin blockchain into motion in January 2009, he was simultaneously introducing two radical and untested concepts. The first is the "bitcoin", a decentralized peer-to-peer online currency that maintains a value without any backing, intrinsic value or central issuer. So far, the "bitcoin" as a currency unit has taken up the bulk of the public attention, both in terms of the political aspects of a currency without a central bank and its extreme upward and downward volatility in price. However, there is also another, equally important, part to Satoshi's grand experiment: the concept of a proof of work-based blockchain to allow for public agreement on the order of transactions. Bitcoin as an application can be described as a first-to-file system: if one entity has 50 BTC, and simultaneously sends the same 50 BTC to A and to B, only the transaction that gets confirmed first will process. There is no intrinsic way of determining from two transactions which came earlier, and for decades this stymied the development of decentralized digital currency. Satoshi's blockchain was the first credible decentralized solution. And now, attention is rapidly starting to shift toward this second part of Bitcoin's technology, and how the blockchain concept can be used for more than just money.

Commonly cited applications include using on-blockchain digital assets to represent custom currencies and financial instruments ("colored coins"), the ownership of an underlying physical device ("smart property"), non-fungible assets such as domain names ("Namecoin") as well as more advanced applications such as decentralized exchange, financial derivatives, peer-to-peer gambling and on-blockchain identity and reputation systems. Another important area of inquiry is "smart contracts" - systems which automatically move digital assets according to arbitrary pre-specified rules. For example, one might have a treasury contract of the form "A can withdraw up to X currency units per day, B can withdraw up to Y per day, A and B together can withdraw anything, and A can shut off B's ability to withdraw". The logical extension of this is decentralized autonomous organizations (DAOs) - long-term smart contracts that contain the assets and encode the bylaws of an entire organization. What Ethereum intends to provide is a blockchain with a built-in fully fledged Turing-complete programming language that can be used to create "contracts" that can be used to encode arbitrary state transition functions, allowing users to create any of the systems described above, as well as many others that we have not yet imagined, simply by writing up the logic in a few lines of code.

”次世代*Smart contract*と分散型アプリケーション基盤”

*Bitcoin*は*TTP*を介さない分散型デジタル資産の例であるが、それだけでなく分散型大衆決定ツールとしてその基盤を成す*Blockchain*技術である。

*Bitcoin*に対して*Ethereum*は『チューリング完全なプログラミング言語の実行環境を*Blockchain*上に構築する』点を特徴としている。

この言語によって”contract”という関数をプログラムしたものが作成され、これによって以下のアプリケーションが実装可能になる。

- 一定取引量のある通貨や金融商品（colored coins）
- 基本的な物理デバイスの所有権（ownership）
- ドメインのような代替不可能の資産（Namecoin）
- あらかじめ決まっているルールでデジタル資産を自動交換（smart contract）
- 組織規則のコード化や資産を含有する長期スマートコントラクト（DAO）

Philosophy

- *Simplicity*（シンプル性）
- *Diversity*（多様性）
- *Modularity*（モジュール性）
- *Agility*（機敏性）
- *Non-discrimination/non-censorship*（無差別・無検閲）

Simplicity

*Ethereum*はプロトコルレベルでの
『民主化』を目指しており、
普遍的なプログラマーがコードを
記述できるようにシンプルであるべき。
たとえ効率性が上がったとしても
複雑なプロトコルになるのであれば
最適化の余地はない。

Diversity

*Ethereum*は基本的に機能を持たず、
チューリング完全なプログラミング言語
『Solidity』を用いることによって
数学に基づいたトランザクションや
コントラクトを利用可能。
例として、Derivative（金融派生商品）や
Tokenを作ることができる。

Modularity

*Ethereum*はプロトコルレベルでの
『分割可能性』を実現しており、
プロトコルの改修を一箇所行うことで
アプリケーションの修正は必要ない。
また、ライブラリを実装することで
*Ethereum*の特定の機能を必要とせずに
実行可能。
Composability（構成可能性）

Agility

*Ethereum*のプロトコルは不変でないと
いけない。
開発途中でプロトコル設計やEVM
（イーサリアム仮想マシン）等の修正で
大幅にスケーラビリティやセキュリティ
を向上できる可能性もあり、その際には
慎重に変更を行う。

Non-discrimination
Non-censorship

*Ethereum*はアプリケーションに対して
検閲を行なってはいけない。
規制するべきは直接的な害を
もたらすことであり、
望ましくないアプリケーションを
規制してはいけない。

Ethereumを理解する上での重要項目

- *Smart Contract*
- *Account Model*
- *ERC20/ERC721*
- 分散型金融（DeFi）
- 自立分散型組織（DAO）

Smart

nszabo@law.gwu.edu

Nick Szabo's Papers and Concise Tutorials

Contract

The Idea of Smart Contracts

What is the meaning and purpose of "security"? How does it relate the the relationships we have? I argue that the formalizations of our relationships -- especially contracts -- provide the blueprint for ideal security.

Many kinds of contractual clauses (such as collateral, bonding, delineation of property rights, etc.) can be embedded in the hardware and software we deal with, in such a way as to make breach of contract expensive (if desired, sometimes prohibitively so) for the breacher. A canonical real-life example, which we might consider to be the primitive ancestor of smart contracts, is the humble vending machine. Within a limited amount of potential loss (the amount in the till should be less than the cost of breaching the mechanism), the machine takes in coins, and via a simple mechanism, which makes a freshman computer science problem in design with finite automata, dispense change and product according to the displayed price. The vending machine is a contract with bearer: anybody with coins can participate in an exchange with the vendor. The lockbox and other security mechanisms protect the stored coins and contents from attackers, sufficiently to allow profitable deployment of vending machines in a wide variety of areas.

Smart contracts go beyond the vending machine in proposing to embed contracts in all sorts of property that is valuable and controlled by digital means. Smart contracts reference that property in a dynamic, often proactively enforced form, and provide much better observation and verification where proactive measures must fall short.

As another example, consider a hypothetical digital security system for automobiles. The smart contract design strategy suggests that we successively refine security protocols to more fully embed in a property the contractual terms which deal with it. These protocols would give control of the cryptographic keys for operating the property to the person who rightfully owns that property, based on the terms of the contract. In the most straightforward implementation, the car can be rendered inoperable unless the proper challenge-response protocol is completed with its rightful owner, preventing theft.

If the car is being used to secure credit, strong security implemented in this traditional way would create a headache for the creditor - the repo man would no longer be able to confiscate a deadbeat's car. To redress this problem, we can create a smart lien protocol: if the owner fails to make payments, the smart contract invokes the lien protocol, which returns control of the car keys to the bank. This protocol might be much cheaper and more effective than a repo man. A further reification would provably remove the lien when the loan has been paid off, as well as account for hardship and operational exceptions. For example, it would be rude to revoke operation of the car while it's doing 75 down the freeway.

In this process of successive refinement we've gone from a crude security system to a reified contract:

```
(1) A lock to selectively let in the owner and
    exlude third parties;
(2) A back door to let in the creditor;
(3a) Creditor back door switched on only upon nonpayment
     for a certain period of time; and
(3b) The final electronic payment permanently switches
     off the back door.
```

Mature security systems will be undertaking different behavior for different contracts. To continue with our example, if the automobile contract were a lease, the final payment would switch off leasee access; for purchase on credit, it would switch off creditor access. A security system, by successive redesign, increasingly approaches the logic of the contract which governs the rights and obligations covering the object, information, or computation being secured. Qualitatively different contractual terms, as well as technological differences in the property, give rise to the need for different protocols.

(Derived from "Formalizing and Securing Relationships on Public Networks" , by Nick Szabo)

A related article discusses a formal language for analyzing contracts and specifying smart contracts.

部分和訳

スマートコントラクトのアイディア

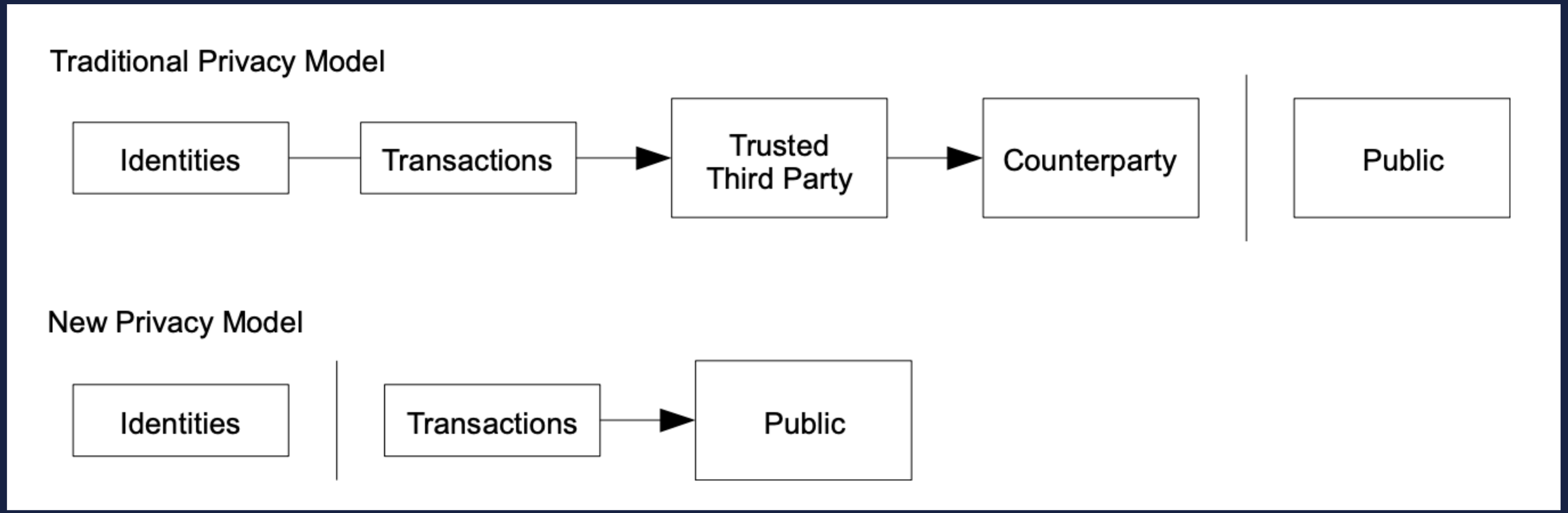
スマートコントラクトの原始的な祖先と考えられる典型的な現実の例は、粗末な自動販売機である。
この自動販売機は、限られた潜在的な損失額の範囲内でコインを取り込み、有限オートマトンによる設計でコンピューターサイエンスの1年生の問題になるような簡単なメカニズムによって、表示された価格に応じて釣り銭や製品を払い出す。
自動販売機は無記名契約であり、コインを持っている人なら誰でも業者との交換に参加することができる。
セキュリティ機構によって保管されたコインやコンテンツは攻撃者から保護され、さまざまな地域で自動販売機を収益的に展開することが十分に可能である。

スマートコントラクトは自動販売機を超えて、デジタル手段で管理される価値のあるあらゆる種類のアセットにコントラクトを組み込むことを提案している。

Smart Contract

- コントラクトは単なる『合意』
- トラストできる関係性である前提でないと契約は成立しない
  →相手が契約を履行しないかもしれない（Smartではない）
- *Smart Contract*のメリットは『契約条件を満たした時に自動執行される』点
  →契約執行をTrust-lessに行うことができる
- *Ethereum*は*Public Blockchain*であり、取引履歴や関連情報を誰もが閲覧可能
  →監査や追跡に有効
- 自己のアイデンティティがトランザクションと切り離されていることから、プライバシー向上に必然的に繋がる
- 契約時にはデジタル署名を行うが、署名前に契約内容の確認ができる
  →これは当事者に限らず、誰もが契約内容の精査が可能

Bitcoin Whitepaper 10章

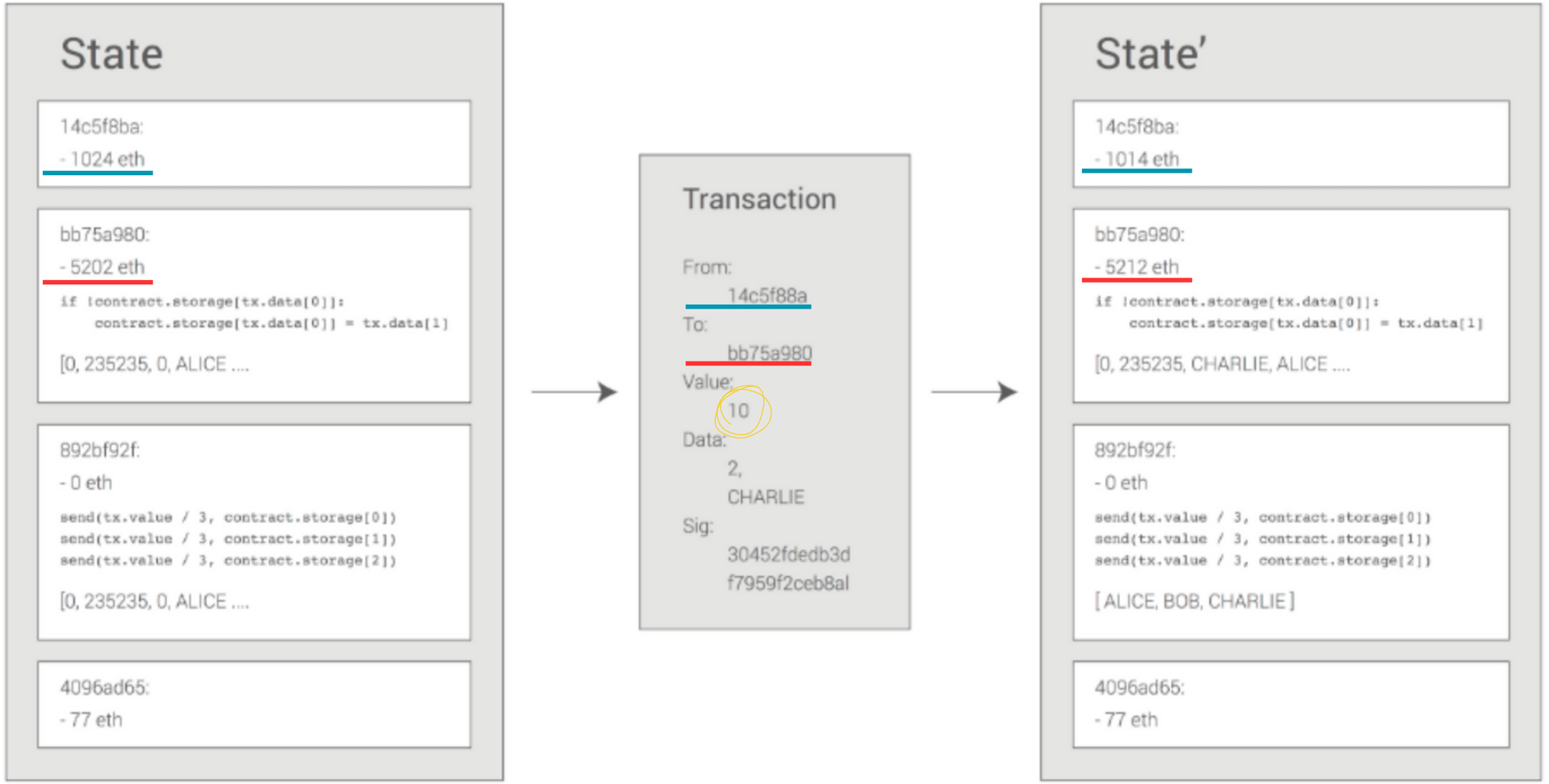


Account

Model

- *BitcoinはUTXO Model*であるが、*EthereumはAccount Model*
- *Ethereum*では*2*つの*Account*（*EOA/CA*）が登場する
- ***EOA***は*Externally Owned Account*（外部所有アカウント）の略称
- *EOA*はコードを持たず、トランザクションを生成して署名しメッセージ送信
- ***CA***は*Contract Account*（コントラクトアカウント）の略称
- *CA*はメッセージを受信した場合、コードを実行してトランザクションの送信や新しいコードの生成などを行う
- *CA*は*Ethereum*実行環境内の『自動金融エージェント』のような存在

Ethereum State Transition Function



ERC20/ERC721

*ERC*とは？

- ***ERC***は*Ethereum Request for Comment*の略称
- *ERC*は*Ethereum*の改善案のことで、後ろにつく番号はその序数
- ***EIP***というものも存在し、これは*Ethereum Improvement Proposals*の略称
- 直訳すると『Ethereum技術提案』
- 関係性としては、『EIP上にはERCが提出され、コミュニティ可決を得られると正式なERCとなり、次回アップデートで実装される』
- *ERC*は共通規格

ERC20/ERC721

- *ERC20*や*ERC721*は*Token*に関する*ERC*
- *Token*は『インターネット上で価値を持つもの』と捉えるとわかりやすい
  （個人的に）
- ***ERC20***は*Fungible Token*と呼ばれ、代替可能性を持つトークン
- *Fungibility*とは、どの100円であっても同じ価値で、交換可能であるということ
- ***ERC721***は*Non-Fungible Token*と呼ばれ、非代替可能性を持つトークン
- *Non-Fungiblity*とは、一点ものの絵画が他の絵画と交換不可能であるということ
- *ERC20*では各アドレスの保有量が記録され、*ERC721*では各トークン（*ID*）について
  保有者アドレスを記録する

＊*ERC20*と*ERC721*のコードレベルでの比較は*Hands-on*にて行う。

分散型金融
（Decentralized Finance）

イーサリアムを使う 学ぶ デベロッパー エンタープライズ コミュニティ 検索 言語

イーサリアムの用途は常に発展し、進化しています。このページに記載の内容をより分かりやすく、または関連する最新情報をご存知の場合は、ぜひ追加してください。 ページの編集

分散型金融(DeFi)

- 現在の金融システムに変わる、グローバルで開かれた分散型金融。
- 借りる、貯蓄、投資、取引などを実現する製品。
- オープンソース技術に基づき、誰でもプログラミング可能。

出典：https://ethereum.org/ja/defi/

比較

分散型金融(DeFi)	従来の金融システム
自分でお金を保有します。	企業があなたのお金を保有します。
自分のお金がどこに行くか、どのように使われるかをコントロールすることができます。	あなたのお金が不正に管理されたり、危険な借り手に貸しだされないなど、その企業を信用する他ありません。
資金の移動は数分で完了します。	支払いには手動の作業プロセスのために数日かかることがあります。
トランザクションは仮名で行われます。	金融活動は、利用者の個人情報と密接に結びついています。
分散型金融(DeFi)は誰でも利用できます。	金融サービスの利用申請が必要です。
市場は常にオープンしています。	従業員の休息のため、市場が閉じる時間があります。
誰もが製品データを見て、システムがどのように機能しているかを調べることができる、透明性の高いシステムです。	金融機関はクローズド・ブックであり、融資の履歴や運用資産の記録などを見せてもらうことはできません。

出典：https://ethereum.org/ja/defi/

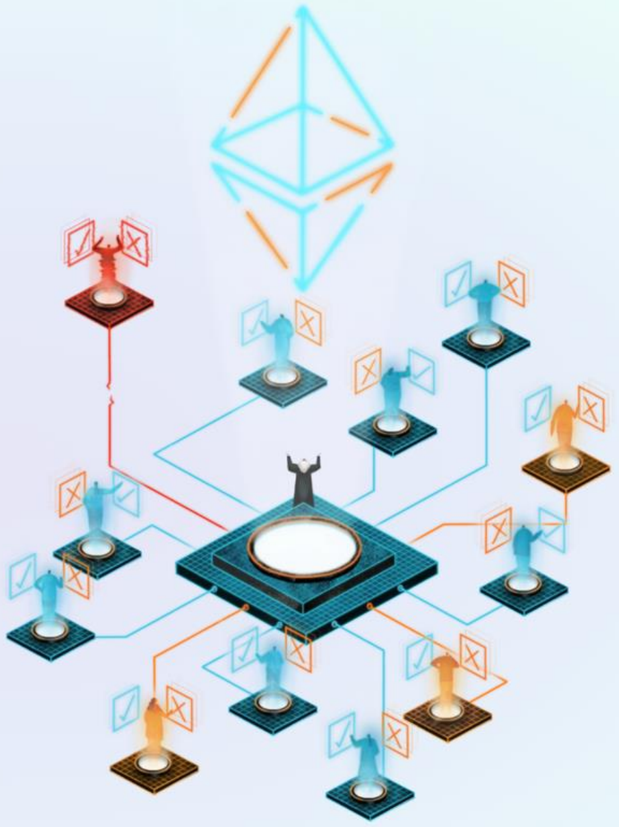
自立分散型組織
（Decentralized Autonomous Organization）

イーサリアムを使う　学ぶ　デベロッパー　エンタープライズ　コミュニティ　検索 ⌘ k　言語

イーサリアムの用途は常に発展し、進化しています。このページに記載の内容をより分かりやすく、または関連する最新情報をご存知の場合は、ぜひ追加してください。 ページの編集

分散型自律組織(DAO)

- 中央集権的な制御がない、メンバー所有のコミュニティ
- インターネットの見知らぬ人と協力する安全な方法
- 特定の目的に資金を委ねるのに安全な場所



出典：https://ethereum.org/ja/dao/

比較

分散型自律組織(DAO)	従来の組織
通常はフラットな組織で、完全に民主化	通常は階層的
変更を実行するには、メンバーによる投票が必要	組織構造によっては、単独の当事者から変更が要求されることがあり、または投票が行われる場合がある
投票は集計され、結果は信頼できる仲介者なしに自動的に実行される	投票が可能な場合、投票は内部で集計され、投票結果は手動での処理が必要
提供されるサービスは、自動的に分散化された方法で処理される(例えば慈善資金の分配)	人間による処理、または集中管理された自動化を必要とし、改ざんされるおそれがある
すべてのアクティビティは透明で完全に公開	通常、アクティビティは非公開で、一般には非公開

出典：https://ethereum.org/ja/dao/